ANADOLU ÜNİVERSİTESİ

anadolum eKampüs

anadolum eKampüs ve anadolu mobil

dilediğin yerden, dilediğin zaman, öğrenme fırsatı!

(ekampus.anadolu.edu.tr)

(mobil.anadolu.edu.tr)

ekampus.anadolu.edu.tr

- Takvim
- Duyurular
- Ders Kitabı (PDF)
- Epub
- Html5
- Video
- Canlı Ders
- Sesli Kitap
- Ünite Özeti
- Sesli Özet
- Sorularla Öğrenelim
- Alıştırma
- Deneme Sınavı
- İnfografik
- Etkileşimli İçerik
- Bilgilendirme Panosu
- Çıkmış Sınav Soruları
- Sınav Giriş Bilgisi
- Sınav Sonuçları
- Öğrenci Topluluklari

AOSDESTEK
AÇIKÖĞRETİM DESTEK SİSTEMİ

Açıköğretim Sistemi ile ilgili merak ettiğiniz her şey AOS Destek Sisteminde...

- Kolay Soru Sorma ve Soru-Yanıt Takibi
- Sıkça Sorulan Sorular ve Yanıtları
- Canlı Destek (Hafta İçi Her Gün)
- Telefonla Destek

aosdestek.anadolu.edu.tr

AOS DESTEK Sistemi İletişim ve Çözüm Masası

0850 200 46 10

www.anadolu.edu.tr

/AOFAnadolum /Anadolu_Univ instagram.com/anadoluuniv

T.C. ANADOLU ÜNİVERSİTESİ YAYINI NO: 3577
AÇIKÖĞRETİM FAKÜLTESİ YAYINI NO: 2410

İŞLEM TABLOSU PROGRAMLAMA

Yazarlar
Öğr.Gör. Hüseyin ÖZKAYA (Ünite 1, 2)
Dr.Öğr.Üyesi Savaş Selahattin ATEŞ (Ünite 5, 6, 7)
Öğr.Gör. Emre KAÇMAZ (Ünite 3, 4, 8)

Editör
Doç.Dr. Cihan KALELİ

ANADOLU ÜNİVERSİTESİ

Kapak Düzeni
Prof.Dr. Halit Turgay Ünalan

Dizgi
Kitap Hazırlama Grubu

İŞLEM TABLOSU PROGRAMLAMA

E-ISBN
978-975-06-2595-4

ESKİŞEHİR, Ağustos 2018

3110-0-0-0-2209-V01

İçindekiler

Önsöz

İşlem tablosu programlama yazılımları geliştirildikleri ilk günden günümüze kadar ciddi bir gelişim göstermiştir. Bu tür yazılımların sağlamış olduğu programlama kabiliyeti ile birçok kişi, profesyonel yazılımcı olmadığı halde kendi hesaplama modelini oluşturarak gereksinimlerini karşılayabilmiştir. Bu nedenle, bu tür yazılımlara olan ilgi giderek artmış, günlük hayatımızın birçok alanında işlem tablosu programları kullanılır hale gelmiştir. Ayrıca, birçok iş ilanında işlem tablosu programlama yazılımlarında tecrübe arayışı neredeyse standartlaşmıştır. Bu kitap ile birlikte, siz değerli öğrencilerimizin işlem tablosu programlama konusunda tecrübe kazanmaları hedeflenmektedir. Bu doğrultuda, kitap içeriği sizlere en çok fayda sağlayacak konulardan seçilirken, işlem tablosu programlama yazılımı olarak da bu tip yazılımların dünyada en yaygın kullanılanlarından biri olan Microsoft Excel seçilmiştir.

Bu kitabın size sadece belirli bir eğitim - öğretim dönemi süresince fayda sağlamasından ziyade tüm hayatınız boyunca bir başvuru kaynağı olması hedeflenmiştir. Bu doğrultuda, kitabın sizlere ulaşmasında emeği geçen herkese, göstermiş oldukları çabadan dolayı teşekkür ederim.

Editör
Doç.Dr. Cihan KALELİ

İŞLEM TABLOSU PROGRAMLAMA

1

Amaçlarımız

Bu üniteyi tamamladıktan sonra;

- İşlem tablosu uygulamalarının işlevleri, özellikleri ve tarihsel gelişimini hakkında bilgi sahibi olacak,
- İşlem tablosu programlamasının temel kavramlarını tanımlayabilecek,
- Excel işlem tablosu programında formül oluşturmayı ve ön tanımlı fonksiyonları kullanmayı öğrenebileceksiniz.

Anahtar Kavramlar

- İşlem Tablosu Programlama
- İşlem Tablosu Avantajları
- İşlem Tablosu Tarihçe
- İşlem Tablosu Özellikleri
- Programlama Nesneleri
- Adresleme Modları
- Formüller
- Referans Modları
- Özel İsim Kullanımı
- Formül Operatörleri
- Formül Hataları
- Ön Tanımlı Fonksiyonlar

İçindekiler

İşlem Tablosu Programlama → İşlem Tablosu ve Programlama

- GİRİŞ
- TARİHÇE
- İŞLEM TABLOSU PROGRAMLAMA KAVRAMLARI
- ÖN TANIMLI FONKSİYONLAR

İşlem Tablosu ve Programlama

GİRİŞ

İşlem tablosu programları; matematiksel ve mantıksal işlemlerin, formüller veya veriler içeren hücreler üzerinde, belirli bir yapı içerisinde gerçekleştirilmesini sağlayan programlardır. Bu programlarla profesyonel programcı olmayan kullanıcılar da hesaplama modelleri oluşturabilirler. Günlük hayattaki pek çok faaliyetin hem bilgisayar ortamına aktarılması hem de bu faaliyetlerdeki veriler arasında ilişkilerin kurulması işlem tablosu programları ile kolayca yapılabilmektedir. Öğrenci başarı notlarının hesaplanması, stok takiplerin yapılması, arşiv kayıtlarının oluşturulması, mali raporların hazırlanması gibi hemen her alanda kullanılabiliyor olmaları işlem tablosu programlarına olan ilgiyi artırmıştır.

İşlem tablosu programlarının kullanımı kişisel bilgisayar sahibi olma isteğini de artırmış ve bu da kişisel bilgisayarların yaygınlaşmasında önemli bir etken olmuştur. Kullanımının kolaylığı ve sağladığı avantajlar nedeniyle neredeyse her kişisel bilgisayarda işlem tablosu programı bulunmaktadır. Arşiv kayıtlarının tutulması, muhasebe hesaplamalarının yapılabilmesi, puantajların hazırlanması gibi pek çok matematiksel ve mantıksal işlemin uzmanlık gerektirmeden yapılabiliyor olması günümüzde iş dünyasında da işlem tablosu programlarına olan profesyonel ilgiyi beraberinde getirmiştir.

Aslında günlük hayatımızdaki hemen her faaliyet sayı, metin, tarih gibi değerlere dönüştürülerek işlem tablosu programları aracılığıyla bilgisayar ortamına aktarılabilmektedir. Böylece bu değerler veya veriler arasında ilişkiler kurulabilmekte, hesaplamalar yapılabilmekte ve yeni veriler elde edilebilmektedir. Alışveriş mağazaları hesap işlemlerini basit formüller kullanarak yapabilmektedirler. Tıp, mühendislik, finans sektörü gibi karmaşık fonksiyonel işlemlerin kullanıldığı alanlarda bile işlem tablosu programları kullanılabilmektedir.

İşlem tablosu programlarının en büyük avantajı profesyonel yazılımcı veya programcı olmayan insanlar tarafından da rahatlıkla kullanılabiliyor olmasıdır, fakat ileri seviye fonksiyonel programlamalar için yazılım bilgisi gerekebilmektedir. Aslında işlem tablosu programlarını kullanan çoğu kullanıcı aynı zamanda programcıdır denilebilir, çünkü bilmeden ya da farkında olmadan basit formüller kullanılarak programlama yapılmakta ve bu da işlem tablosu programlarını en çok kullanılan programlama ortamları haline getirmektedir.

İşlem tablolarının kullanımlarının kolay olması yaygınlaşmalarındaki en önemli faktörlerdendir. Kullanıcılar kendileri fonksiyonlar tanımlayabilecekleri gibi önceden hazırlanmış ön tanımlı fonksiyonları da kullanabilirler. Örneğin "TOPLA", "ÇARPIM"," ORTALAMA" gibi fonksiyonlar ön tanımlı fonksiyonlardır ve kullanıcılar formül yazmaya gerek kalmadan bu fonksiyonları kullanarak rahatlıkla işlem yapabilmektedirler.

İşlem tabloları programlama ortamlarıdır fakat profesyonel programlama ortamlarına göre farklılık gösterirler. İşlem tablosu programlarında önceden hazırlanmış fonksiyonlar kullanılır. Kullanılan fonksiyonların karmaşıklığı ve yapılan işlemlerin büyüklüğü kullanıcıların bilgi, beceri ve tecrübesine göre değişkenlik gösterir. İşlem tablolarının profesyonel programlama ortamlarından farklılıkları genelde eksiklikleridir. Örneğin, hata ayıklama, gelişmiş yazılım kütüphanelerinin eklenmesi gibi özellikler işlem tablosu programlarında bulunmaz. Genellikle ön tanımlı olan ya da sonradan tanımlanan fonksiyonların kullanılması ile programlama yapılır. Teknik yeterlilikler anlamında eksikleri bulunmasına rağmen oldukça yaygın kullanılan işlem tablosu programlarının en önemli avantajı ise kullanıcılarının programlama bilgisine ihtiyaç duymamasıdır. Bunun yanı sıra kopyala ve yapıştır, çıktı alma, verileri kolayca düzenleme ve işleme gibi özellikleri de işlem tablosu programlarının avantajları arasında sayılabilir. Kullanım kolaylığı ve programlama bilgisi gerektirmeyişi profesyonel programlama ortamlarına göre kullanım oranlarının bir hayli yüksek olmasını sağlamaktadır.

TARİHÇE

İlk işlem tablosu programı 1978 yılında VisiCalc ismi verilen programdır. Bu program Apple II bilgisayarlar için geliştirilmiştir. VisiCalc kendinden sonra gelecek işlem tablosu programları için temel oluşturmuştur.

VisiCalc programının geliştirildiği yıllarda SuperCalc ismiyle yeni bir işlem tablosu programı da farklı bir firma tarafından geliştirilmiştir. Her iki program da 1981 yılında IBM firmasının kişisel bilgisayarları geliştirilmesiyle PC'ler için sürüm yayınlamışlardır.

VisiCalc işlem tablosu programı SuperCalc programına göre daha fazla başarı elde etmiştir. Kişisel bilgisayarların kullanımıyla birlikte işlem tablosu programlarına artan talep ve bu programların başarısı yeni işlem tablosu programlarının geliştirilmesini sağlamıştır. 1983 yılında Lotus 1-2-3 ismi verilen işlem tablosu programı, geliştirilen en başarılı programlardandır. Lotus Geliştirme Firması tarafından üretilen Lotus 1-2-3 kullanıcılar tarafından daha çok tercih edilmiş ve kısa zamanda işlem tablosu programları arasında en çok kullanılan ürün olmuştur. Lotus 1-2-3, VisiCalc isimli işlem tablosu programını da geride bırakarak satışlarda üst sırada kalmayı uzun bir süre devam ettirmiştir. 1990 yılına kadar bu hakimiyetini sürdüren Lotus 1-2-3, bu hakimiyetini, 16 bitlik IBM PC'nin sağlayabileceği tüm avantajları kullanarak kullanıcı etkileşimini hızlandırmasına borçludur.

Lotus 1-2-3 aynı zamanda makroları desteklemekteydi. Klavye kayıtlarını tutması ve bu kayıtları tekrar edebilmesi sayesinde kullanıcıların ilgisini çekmiş ve bu da üst sıralarda kalmasındaki diğer önemli etkenlerden biri olmuştur. Bunların yanı sıra Lotus 1-2-3 sınırlı veri tabanı desteği sunuyor ve grafik desteği de veriyordu. Bu artılarıyla diğer rakiplerinden sıyrılmış ve işlem tablosu programları piyasasında hakim durumda kalabilmiştir.

İşlem tablosu programlarına 1982 yılında Microsoft firması, metin tabanlı MultiPlan isimli programı ile girmiş fakat bu program yaygınlık kazanamamıştır. Kullanımının zorluğu ve öğrenilmesinin güçlüğü yaygınlık kazanamamasının temel sebeplerindendir. 1985 yılında Microsoft firması Excel isimli işlem tablosu programını geliştirmiştir. Macintosh bilgisayarlar için geliştirilen bu program Multiplan programından farklı olarak grafik tabanlı tasarlanmış ve geliştirilmiştir. Microsoft firması, 1985 yılında Macintosh bilgisayarlar için geliştirdiği Excel programını, Windows bilgisayarlar için 1987 yılında piyasaya sürmüştür. 1987 yılında Windows'un yaygın olmaması nedeniyle Microsoft firması basitleştirilmiş bir Windows işletim sistemini de Excel programı ile dağıtmaya başlamıştır. Excel programı bu sürümüyle XLM ismindeki makro dilini kullanmıştır.

Windows için Excel 3 sürümü 1990 yılında çıkarılmıştır. Microsoft firmasının pek çok yeni özellik ve eklentilerle çıkardığı bu sürüm daha kullanıcı dostu ve işlevsel açıdan da daha kullanışlıydı. Özellikle 3 boyutlu grafikler, makrolar, veri tabanları bağlantıları, görev çubuğu gibi yeni özellik ve eklentiler Excel 3 ü diğer işlem tablosu programları arasında öne çıkarmaktaydı. Excel 3 ten sonra 1992 yılında Excel 4 yeni özellik ve eklentilerle

çıkarılmıştır. Microsoft çıkardığı her Excel sürümüyle biraz daha kullanıcı dostu arayüz geliştirmeyi başarmış ve sadece görsel olarak değil özellik olarak da pek çok yenilik eklemiştir. Bu nedenledir ki Excel 4 teknoloji dergilerinin yaptığı işlem tabloları karşılaştırmalarında en yüksek notları almıştır.

Excelin güncel sürümlerinde VBA makro dili kullanılmaktadır fakat Excel 4 ve öncesinde hazırlanmış dosyaların da açılabilmesi için Microsoft XLM diline desteğini sürdürmektedir.

Microsoft 1994'te Excel 5'i piyasaya sürmüştür. Excel 5'le gelen en önemli yenilik XLM makro dili yerine VBA dilinin kullanılmaya başlanmasıdır. Günümüzde hala VBA dili kullanılmaktadır. XLM diline göre daha kolay öğrenilebilmesi ve kullanışlı oluşu Excel 5'te VBA makro diline geçilmesinin temel gerekçeleridir. Microsoft firması Office programlarında standart bir isimlendirme kullanabilmek için 1995 yılında Office 1995 ile birlikte Excel 7'yi çıkarmıştır. Excel 7'ye bakıldığında yeni sürüm için fazla değişiklik olmadığı görülebilmektedir, Microsoft Firması'nın isimlendirme standardı nedeniyle erken çıkarılan bu sürümde önemli performans geliştirmeleri bulunmaktadır. 1997 yılında Office 97 programlar grubu içerisinde Excel 8 piyasaya sürülmüştür. Microsoft'un isimlendirme standardı nedeniyle Excel 97 ismini de taşıyan bu sürümde pek çok iyileştirme mevcuttur. VBA tabanlı uygulamalar geliştirmek için yeni bir kullanıcı arayüzüyle beraber özel diyalog kutuları geliştirmek için de yeni yaklaşımlar eklenmiştir.

2000 yılında Office 2000 programlar grubu içerisinde Excel 2000 sürümü çıkarılmıştır. Performans güncellemelerinin yanı sıra internet bağlantılarında da bazı yeniliklerle gelen bu sürümün ardından Excel 2002 piyasaya sürülmüştür. Excel 2002'de bozuk dosyaları tamir etme özelliği eklenmiştir. Ayrıca Excel programı çalışmayı durdurduğunda çalışma dosyasını kaydedebilme özelliği eklenmiştir. Bu özellik kullanıcılar tarafından çok beğenilen özelliklerden biri olmuştur. Excel 2002'den sonra çıkarılan Excel 2003 beklentilerin altında yenilikle piyasaya sürülmüştür. Yapılan küçük iyileştirmeler kullanıcılar tarafından yeterli görünmese de az sayıda kullanıcıyı etkileyen yeni bir özellik de eklenmiştir. XML dosyalarının çalışma sayfasındaki hücrelerle ilişkilendirilebilmesini sağlayan bu özellik herkes için olmasa da bazı kullanıcılar için işe yarar bir iyileştirme olarak görülmüştür. Bu sürümün görsel arayüzü kullanıcılar tarafından çok beğenilmiş ve Excel 2003 ün görsel arayüzü kendinden sonra çıkarılan sürümlerde de eklentiler aracılığıyla kullanılmaya devam etmiştir. Excel 2007'de yeni bir ara yüz ile sürüm çıkaran Microsoft Excel çalışma sayfasının üst kısmında bulunan menüleri şerit biçimindeki kullanıcı arayüzü ile güncellemiştir. Özellikle alışkanlıklardan dolayı Excel 2007'deki bu güncelleme bazı kullanıcılar tarafından olumlu karşılanmamış ve eski sürüm görüntüsünü veren eklentiler ile beraber kullanılmıştır.

Karmaşık ve büyük veri setleri ile çalışmak ve analizler yapmak pivot tabloları ve grafikler ile daha kolay bir hal almıştır.

Excel 2007'den sonra piyasaya sürülen Excel 2010 pivot tabloları, koşullu formatlanabilmesi, resim dosyaları ile çalışabilmesi, hücre içi grafiklerin kullanımı gibi pek çok yeni özelliğinden dolayı çok beğenilmiştir. Yeni eklenen özellik ve fonksiyonların olduğu bu sürüm, kullanıcılar için pek çok zorlu işi özellikle büyük veri setleri ile çalışmayı kolay hale getirmiştir. 2013 yılında Excel 2013 sürümü yayınlanmıştır. Excel 2013 Microsoft firmasının sürüm numaralandırmasına göre Excel 15 olarak da adlandırılır.

Günümüzde işlem tablosu programları arasında en çok kullanılan ve bilinen Microsoft firmasının geliştirmiş olduğu Excel işlem tablosu programıdır. Piyasada çok fazla rakibi olmayan Excel programının en önemli rakipleri arasında OpenOffice ismiyle piyasada varlığını sürdüren açık kaynak kodlu yazılım ve Google Spreadsheets isimli internet tabanlı yazılım gösterilebilir. Yine de bu programların kullanım oranı Excel'e göre oldukça düşüktür. Piyasadaki hakimiyetini kararlıkla sürdüren Excel işlem tablosu programı sadece kendisi ile yarışır durumdadır, yani sadece bir sonraki sürümünün satılması için kullanıcılarının ilgisini çekecek yeni geliştirmeler yapmak ve kullanıcılarının eski sürüm alışkanlıklarını kırarak yeni sürüme geçmelerini sağlamak için çalışma yapmaktadır.

Microsoft firmasının geliştirmiş olduğu Excel programı en çok kullanılan işlem tablosu programı olduğundan bu kitapta işlem tablosu ile alakalı tanımlamalar ve özellikler Excel programı baz alınarak hazırlanmıştır. Aşağıda Excel programına ait temel özellikler listelenmiştir. Listelenmiş özelliklerden bazılarına kitabın ilerleyen bölümlerinde ayrıntılı

bir şekilde yer verilecektir.

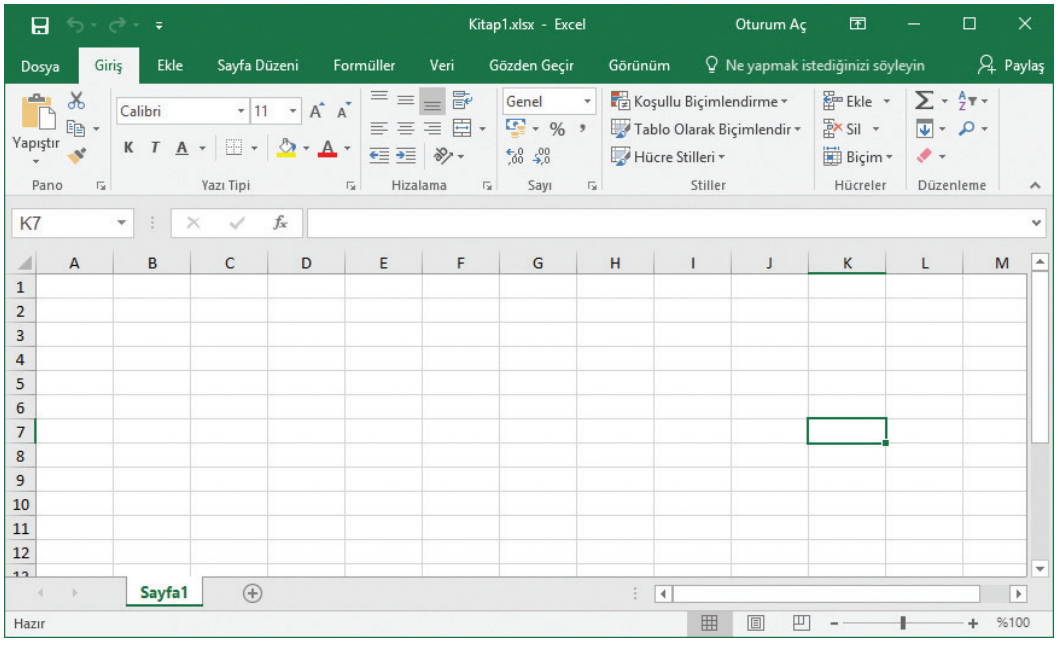
- Excel ile ileri seviye fonksiyonel programlama yapılabilmektedir.
- İleri seviye olmayan bir yazılım bilgisiyle bile buton gibi kontrol elemanları sayfalara eklenebilmektedir.
- Excel dosyasında VBA programlama dili ile yapısal programlar oluşturulabilmektedir.
- Excel çalışma kitabında birden fazla form, grafik ve makro saklanabilmektedir, böylece tüm işlemler tek dosyada yapılabilmektedir.
- Kısayol tanımlayabilme özelliğiyle sıklıkla kullanılan makro ve fonksiyonların üst şeritte kısayolları tanımlanabilmektedir.
- VBA programlama diliyle farenin sağ tıklama özelliğine kısayollar tanımlanabilmektedir.
- İç ve dış kaynaklardan veri aktarabilme özelliği sayesinde web sayfalarından ve veri tabanı dosyaları, yazılı dosyalar gibi yerel kaynaklardan doğrudan veri alınabilmektedir.
- Pivot tablo kullanılarak karışık ve büyük veri setleri kolaylıkla analiz edilebilmektedir.
- Excel programı HTML dosyalar oluşturabilmektedir.
- VBA kullanımıyla diğer programlar ile de etkileşim sağlanabilmektedir.

İŞLEM TABLOSU PROGRAMLAMA KAVRAMLARI

İşlem tablosu programları satır ve sütunlardaki verilerle matematiksel ve mantıksal işlemlerin yapılabilmesini sağlayan programlardır. Satır ve sütunlar veriler veya formüller içerebilir. İşlem tablosu programlarının arayüzü Word benzeri programlardan farklı olarak düz bir sayfa değil içerisinde hücreler barındıran çalışma sayfası şeklindedir. Excel çalışma kitabında ise birden fazla çalışma sayfası bulunabilir. Excel programına ait ekran görüntüsü Resim 1.1 de verilmiştir.

Resim 1.1

Excel İşlem Tablosu Programı Genel Görünümü



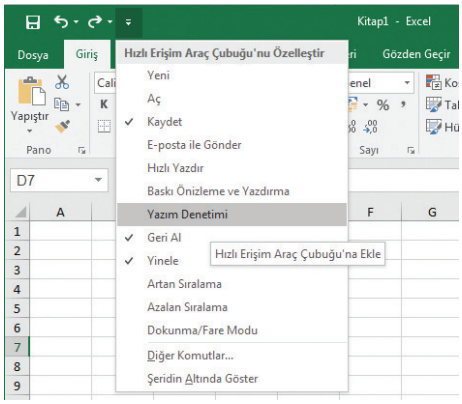
Excel'de çoklu seçme işlemi yapılırken aralarında mesafe bulunan hücreler CTRL tuşuna basılarak, ardışık hücreler ise SHIFT tuşuna basılarak seçilir.

Çalışma Kitabı Excel'de üzerinde çalışılan belgeye verilen isimdir. Çalışma kitabı çalışma sayfalarından oluşur. *Çalışma Sayfası* Excel açıldığında üzerinde çalışılan sayfaya verilen isimdir. Çalışma sayfaları çalışma kitabının sol alt kısmında gösterilir ve ilk açılışta Sayfa1 isimli sayfa açılır. Çalışma sayfaları *Hücre* ismi verilen satır ve sütunların kesişti-

ği kutulardan oluşur. Çalışma sayfasında seçili olan hücreye *Aktif Hücre* ismi verilir ve diğer hücrelere kıyasla daha koyu bir çerçeve içerisine alınır.

Hızlı Erişim Araç Çubuğunda Excel belgesinin ismi yer alır. Ayrıca Kaydet, Geri Al, Yinele fonksiyonları da varsayılan olarak burada yer alır. Bu fonksiyonlar dışında Resim 1.2 de belirtilen fonksiyonların kısa yolları da hızlı erişim araç çubuğuna butonlar şeklinde eklenebilmektedir.

Resim 1.2

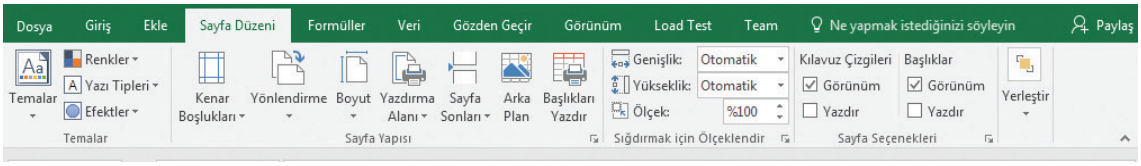


Hızlı Erişim Araç Çubuğu Özelleştirme

Hızlı erişim araç çubuğunun altında *Şerit* bulunur. Şeritte yapılması istenen işlerin yerine getirilebilmesini sağlayacak menüler, alt menüler, butonlar vb. bulunmaktadır. Resim 1.3'de Şerit teki Sayfa Düzeni menüsünü açılmış halde görebilirsiniz.

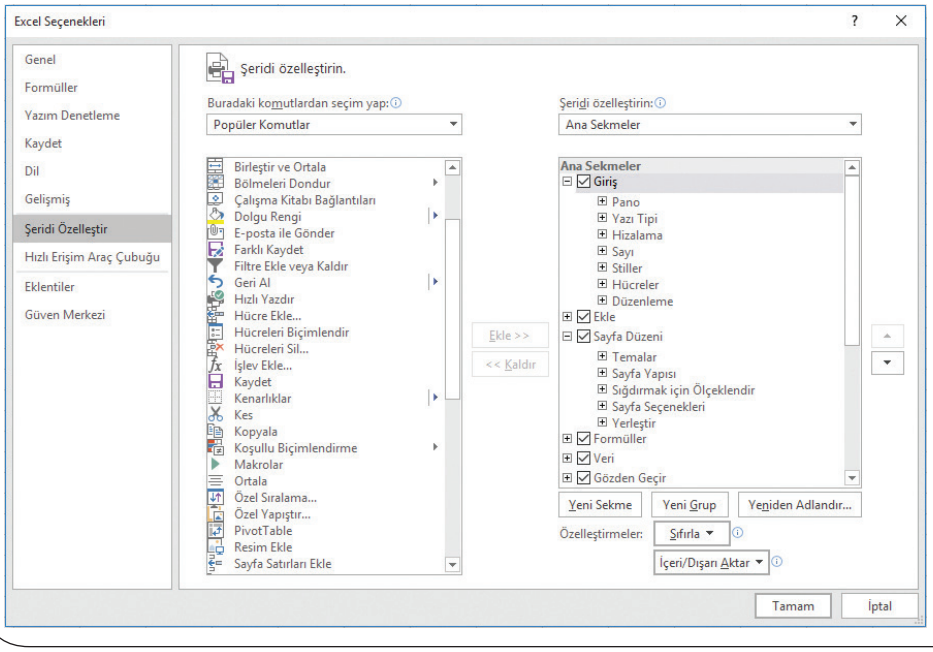
Resim 1.3

Şeritteki Safa Düzeni Menüsü



Şeritte farenin sağ tuşuyla tıklayarak *Şeriti Özelleştir* diyebilir ve özelleştirme seçeneğiyle şerite yeni sekme ekleyebilir ya da var olan sekmeleri silebilirsiniz. Şeriti Özelleştirmek için *Excel Seçenekleri* arayüzüne Dosya menüsünden Seçenekler i tıklayarak da erişebilirsiniz.

Resim 1.4



Excel Seçenekleri Penceresi

Formül Çubuğu şeritin altında yer alır ve aktif hücreye veri girişi için kullanılabilir. Hangi hücrenin aktif olduğu bilgisi ise *Aktif Hücre Referansında* gösterilir. Ayrıca aktif hücre diğer hücrelerden daha koyu bir çerçeve içine alınırken aktif hücrenin bulunduğu satır ve sütun başlıkları da diğerlerinden farklı gösterilir.

DİKKAT

Excel'de birden fazla hücre seçildiğinde hücrelerin çerçeve rengi, satır ve sütun renkleri aktif hücre ile aynı olsa da sadece ilk tıklanan hücre aktif hücre kabul edilir.

Excel'de birden fazla hücreye aynı veri girilmek istenirse hücreler CTRL tuşuna basılarak seçilir, değer girilerek CTRL+ENTER tuşuna basılır.

Şeritteki menülerden herhangi birine tıkladığında şerit açılır ve şeritteki menülerden herhangi birine çift tıklandığında şerit gizlenir. Excel'de bir işlemi yapmak için birden fazla yöntem bulunabilir. Örneğin şeriti gizlemek için çift tıklama haricinde; şerite sağ tıklayarak "*Şeridi Daralt*" diyebilir ya da şeritin sağ altında bulunan yukarı yönlü oku tıklayarak şeriti daraltabilirsiniz.

Çalışma kitabının en alt kısmında *Durum Çubuğu* yer alır. Durum çubuğunda görüntüleme modu ve ölçeği bulunur.

Çalışma sayfasında ekran boyutundan kaynaklı olarak gösterilemeyen hücreleri görüntüleyebilmek için *"Kaydırma Çubukları"* kullanılabilir ya da "*Durum Çubuğu"* kullanılarak çalışma alanını küçültülüp büyültülebilir.

SIRA SİZDE 1

Microsoft Excel programını açarak çalışma sayfası ve bu sayfadaki alanları inceleyiniz. Şeritteki menüleri inceleyerek şeriti özelleştiriniz.

Excel Nesneleri

Excel nesneleri birer sınıf değildir. Yazılımda sınıf (class) kavramı planlamayı ifade ederken nesne (object) kavramı ise bu planın uygulanmasını ifade eder.

Excel işlem tablosu programında nesneler kullanılarak uygulama geliştirilir. Excel programında yüzlerce nesne bulunmaktadır. Excel çalışma kitabı tek başına nesne olduğu gibi çalışma kitabındaki bir çalışma sayfası da bu sayfadaki bir hücre de Excel nesnesidir.

Excelde nesnelerin kontrol edilmesi, güncellenmesi ve programlanması işlem tablosu programlamanın temelini oluşturur. Bu nesnelerin kontrol edilmesi kullanıcı tarafından elle yapılabilirken VBA makro programlama diliyle de yapılabilmektedir. Kitabın ilerleyen bölümlerinde VBA dili ile programlama hakkında detaylı bilgi verilecektir.

Çalışma sayfasına sağ tıklandığında sayfa ile ilgili özelleştirmeler yapılabilir. Sayfanın yeniden adlandırılması, sayfa renginin değiştirilmesi, sayfanın korumaya alınması gibi özellikler bu özelleştirmeler arasındadır. Ayrıca aynı menüden sayfanın silinmesi işlemi de yapılabilir.

Excel programında temel nesne çalışma kitabıdır. Fakat aynı anda sadece bir tane çalışma kitabı aktif olur. Çalışma kitabı çalışma sayfalarından oluşur ve aynı şekilde herhangi bir zamanda sadece bir çalışma sayfası aktif olur. Excelde çalışma sayfaları çalışma kitabının sol alt tarafında listelenir. İstenilen çalışma sayfasına tıklanarak sayfalar arası geçişler kolaylıkla yapılabilir. Sayfaların isimleri sayfanın ismine sağ tıklanarak "*Yeniden Adlandır*" komutu ile veya sayfa ismine çift tıklanarak değiştirilebilir.

DİKKAT

Herhangi bir zamanda bir tane çalışma kitabı aktif olsa da çalışırken birden fazla çalışma kitabı açılabilir.

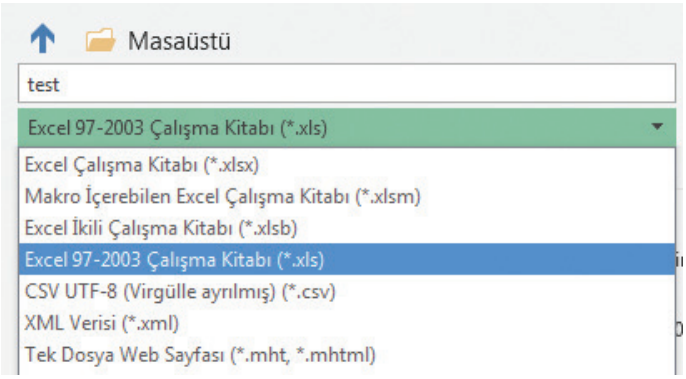
DİKKAT

Excel programında pek çok işlem geri alınabilse de çalışma sayfasının silinmesi geri alınamaz bir işlemdir.

Excel 2007 ve sonrası sürümlerde hazırlanan dosyalar Excel 97-2003 sürümlerinde kullanılmak istenirse, çalışma kitabının .xls uzantılı olarak kaydedilmesi gerekir. Bunun için "Dosya" menüsünden farklı kaydet diyerek "Excel 97-2003 Çalışma Kitabı" seçeneği seçilmelidir.

Excel'de çalışma sayfası satır ve sütunların kesiştiği hücrelerden oluşur. Çalışma sayfasına bunların çok küçük bir kısmı sığar, ihtiyaç duyulması halinde kullanılan diğer satır ve sütunlara kaydırma çubukları ile erişilebileceği gibi, ihtiyaç duyulmadığında da çalışılmayan satır ve sütunlar gizlenebilmektedir. Excel 97-2003 sürümlerinde 65.536 satır ve 256 sütun bulunurken sonraki sürümlerde bu sayılar oldukça arttırılarak 16.384 sütun ve 1.048.576 satır olarak belirlenmiştir. Excel 97-2003 sürümlerinde .xls olan dosya uzantısı da .xlsx ve .xlsm olarak değiştirilmiştir. Fakat Excel 'in sonradan çıkarılan sürümleri .xls uzantılı dosyaları da açabilmektedir.

Resim 1.5



Excel 97-2003 Formatında Dosya Kaydetme

DİKKAT

Farklı kaydet seçeneği ile Excel 97-2003 sürümlerinden sonraki bir sürümden Excel 97-2003 formatında kaydetme işlemi yapıldığında kullanılabilir satır ve sütun sayısı azalacağından 65.536 satır ve 256 sütundan sonraki veriler kaybolur.

Çalışma sayfasından bir satırı veya sütunu silmek için ilgili satır başlığına veya sütun başlığına sağ tıklanarak "Sil" demek yeterlidir. Aynı şekilde bir satırı veya sütunu sağ tıklayarak kesip taşıyabilir ya da kopyalayıp yapıştırabilirsiniz. Çalışma kitabına yeni bir sayfa daha eklemek içinse aktif çalışma sayfasının adının yanında yer alan artı işaretli butona tıklamak yeterlidir.

İşlem tablosu programlarında en temel yapı taşı hücrelerdir. Hücreler başlangıçta eşit önem derecesine sahiptirler, hücrelere doldurulan veriler yani hücrelerin içerikleri hücreleri başkalaştırır. Hücreler doğrudan bir veriyi barındırabileceği gibi formüller ve aritmetiksel işlemleri de barındırabilirler. Hücreleri içerikleri bakımından sınıflandırmak gerekirse aşağıdaki gibi bir sınıflandırma yapılabilir.

- *Sayısal bir değer içeren hücreler:* Rakamlarla ifade edilen her türlü veriyi içerebilen hücrelerdir. Örneğin doğal sayılar, rasyonel sayılar, tarihler, saat gibi veriler bu hücrelerde bulunur.
- *Metin içeren hücreler:* Bu hücreler içerisinde sözel veriler taşırlar, sütunlara verilen isimlendirmeler, kişi adları, hatırlatma notları vb. veriler metin içeren hücrelerin içeriğini oluşturur.
- *Formül içeren hücreler:* Bu hücrelerdeki formüller diğer hücrelere verilen referansları içerir. Formül içerikli bir hücre diğer formül içeren hücrelere de referans olarak verilebilir. Bu hücrelerdeki formüller ile hesaplanan sonuçlar aynı zamanda Excel çıktısı olarak da verilebilir.
- *Mantıksal doğru veya yanlış içeren hücreler:* Mantıksal koşullu karşılaştırma yapmak için mantıksal sınama değerlerini içeren hücrelerdir. Bu hücrelerdeki değerler formül içeren hücrelere referans olarak verilebilir.

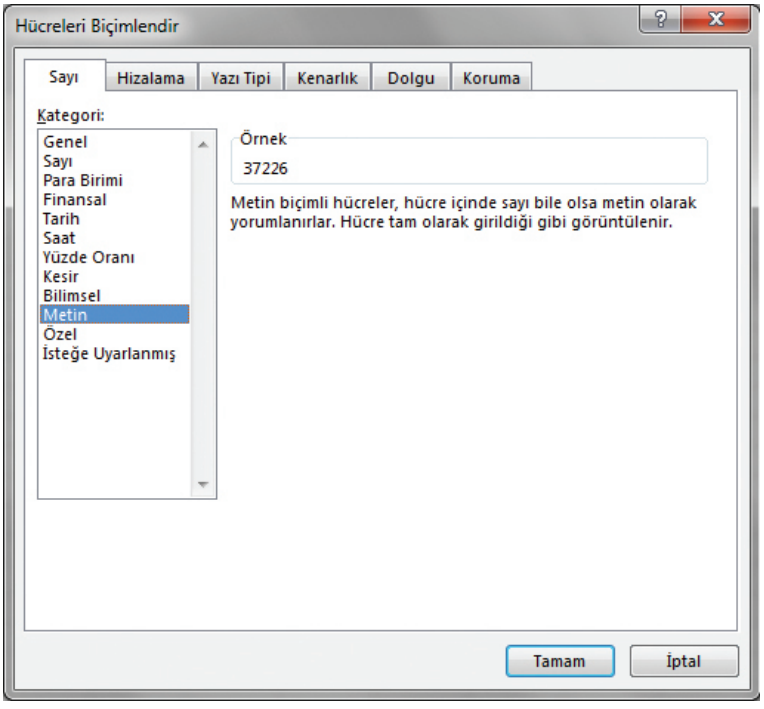
Excelde hücreye sağ tıklayarak açılan menüden "Hücreleri Biçimlendir" komutu ile hücrelerin içerikleri bakımından sınıflandırması yapılabilir.

DİKKAT

Hücre biçimlendirmesi kullanıcı tarafından yapılmazsa Excel hücrelerde bulunan değerlere göre otomatik biçimlendirme yapar. Otomatik biçimlendirme bazen hatalı gösterimlere neden olabilir. Örneğin ARA2001 yazılı bir hücre metin olarak biçimlendirilmezse Excel otomatik biçimlendirme yaparak bu veriyi tarihe dönüştürür ve 01.12.2001 tarihini hücreye yazar.

Resim 1.6

Excelde Hücre Biçimlendirme



İşlem tablosu programlarında program girdisi, program çıktısı ve program tek tabloda tanımlanabilir. Excel gibi işlem tablosu programlarında ayrıca programlama için makro tanımlama ortamları da yer alır. Excel programında VBA makro programlama dili kullanılarak kullanıcı tanımlı makrolar programlanabilir. Makro programlama ile birden fazla kez tekrarlanacak işlemlerin kolayca yapılması sağlanır.

SIRA SİZDE 2

Excel programında yeni bir çalışma sayfasını açınız ve farklı hücre çeşitleri ile hücreler oluşturunuz. Formül içeren hücreler için toplama formülünü, mantıksal doğru veya yanlış içeren hücreler için büyüktür/küçüktür operatörünü kullanabilirsiniz.

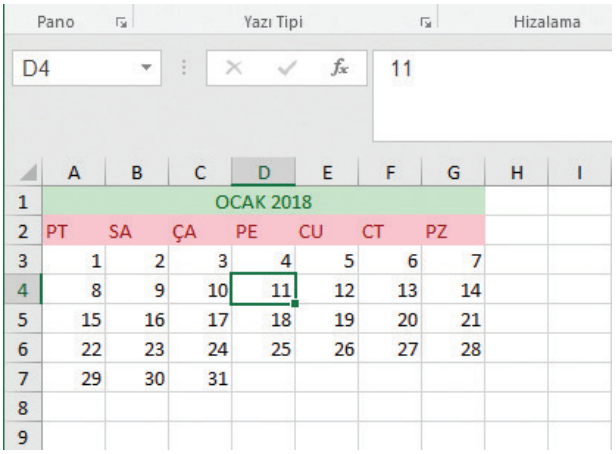
Hücre Adresleme Modları

Excel programındaki çalışma sayfalarının temel yapı taşları hücre nesneleridir. Bu hücrelerden her birinin çalışma sayfasında bir ismi bulunmaktadır. İsimlendirmede genel olarak satır ve sütun bilgisi kullanılır. Hücreler A1 stili, R1C1 stili ya da özel isimlendirmeyle adreslenebilir. Aktif hücre referansına bakılarak adresleme modu anlaşılabilir.

- **A1 Stili**
  Çalışma sayfasındaki hücreler satır ve sütunların kesişim yerlerindeki kutucuklardır. A1 stili adreslemede de bu bilgi doğrudan kullanılarak sütun ismi ve satır ismi birleştirilerek hücrenin ismi oluşur. Örneğin E sütunu 26 numaralı satırdaki hücrenin adresi E26 olarak tanımlanır.

Resim 1.7

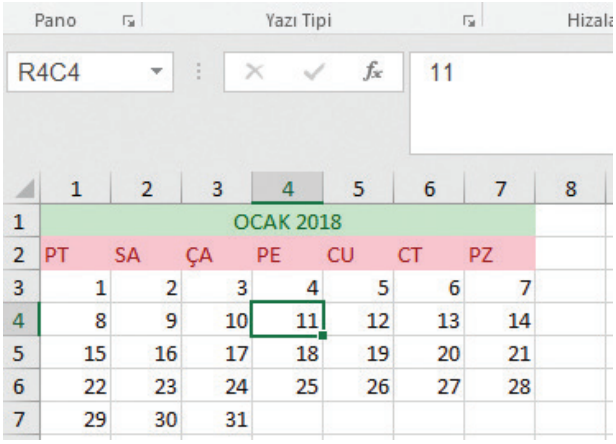
A1 Stilinde Çalışma Sayfası Görünümü



- **R1C1 Stili**
  R harfi satır (**R**ow) C harfi sütun (**C**olumn) için kullanılır. R harfinden sonra satır numarası C harfinden sonra ise sütun numarası yazılır. Örneğin 5 numaralı satır 26 numaralı sütunun kesişim yerindeki hücre için R5C26 olarak adresleme tanımlaması yapılır.

Resim 1.8

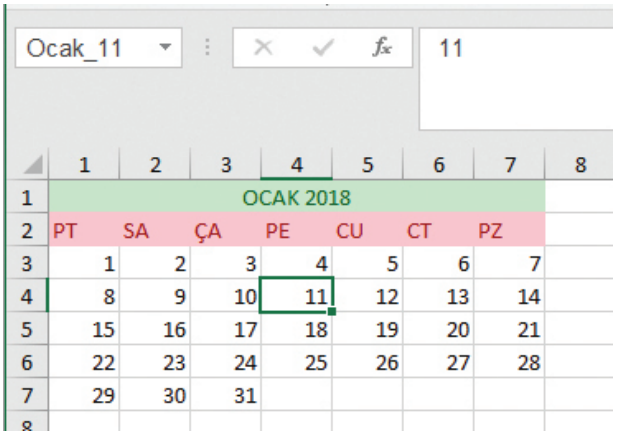
R1C1 Stilinde Çalışma Sayfası Görünümü



- **Özel isimli adresleme**
  Excelde herhangi bir hücreye özel isim verilebilmektedir. Hücrelere özel isim tanımlayabilmek için hücreye sağ tıklanarak *"Ad Tanımla"* komutu verilir. Açılan pencereden hücreye verilecek isim girilir. Özel isimlendirme sadece bir hücreye yapılabildiği gibi birden fazla hücre seçilerek de yapılabilir. Özel isimlendirme kullanılan hücreler ve formüller diğer adreslemelere göre daha kolay okunmaktadır. Ayrıca özel isimlendirme verilen hücrelere, aktif hücre referansı alanından da kolayca ulaşılabilmektedir.

Resim 1.9

Özel İsimlendirme Yapılmış Hücre ve Çalışma Sayfası Görünümü



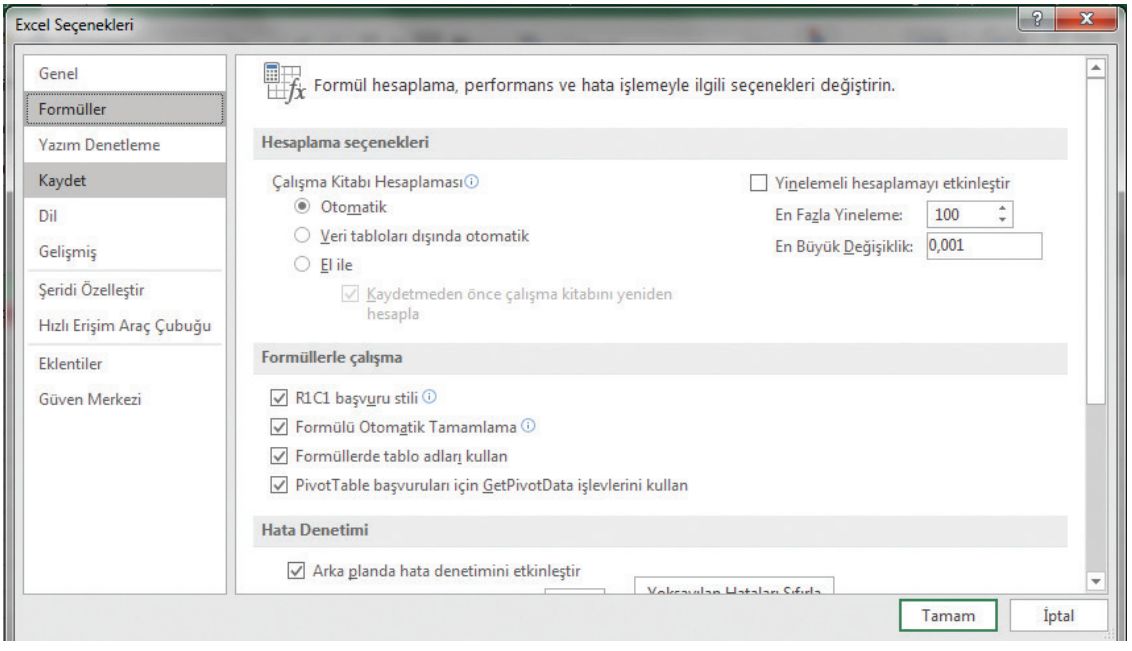
SIRA SİZDE 3

Excelde bir hücre için ve hücre grubu için özel isimlendirme yaparak, aktif hücre referansı alanını gözlemleyiniz. Ayrıca aktif hücre referansından seçim yaparak çalışma sayfasını gözlemleyiniz.

Excel programı varsayılan olarak A1 stilini kullanır. A1 stili ile R1C1 stili arasında ise kolayca geçiş yapılabilmektedir. Stiller arası geçiş yapmak için *Dosya* menüsünden *Seçenekler* penceresine tıklanarak açılır ve bu penceredeki *Formüller* sekmesinden stiller arası geçiş yapılabilir.

Resim 1.10

Formüller Sekmesi ve Stil Değiştirme Alanı



Excelde çalışma sayfasını R1C1 stiline dönüştürünüz ve çalışma sayfasını gözlemleyiniz. Çalışma sayfasında A1 stili ile R1C1 stili arasındaki farklılıkları bulmaya çalışınız.

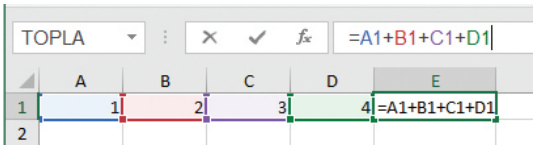
SIRA SİZDE 4

Fonksiyonlar

Fonksiyon ya da formüller işlem tablolarında sıklıkla kullanılırlar. Excel'deki her fonksiyon "="(eşittir) ile başlar, aksi halde Excel fonksiyonu hücre içeriğini düz metin olarak algılar. Fonksiyonların eşittir ile başlaması kuralı hem önceden tanımlanmış fonksiyonlar için hem de kullanıcının kendi tanımladığı fonksiyonlar için geçerlidir.

Excel'de formüllerin kullanım şartları ve standartları vardır. Bu şartlar dışına çıkılmadığı sürece birden fazla formül iç içe ve aynı anda kullanılabilir. Formüller; rakamlar gibi sabit değerler, diğer hücrelere verilmiş referanslar, başka formüller ve "+, -, /, *" gibi operatörleri içerebilir.

Resim 1.11



Excelde Formül Tanımlama

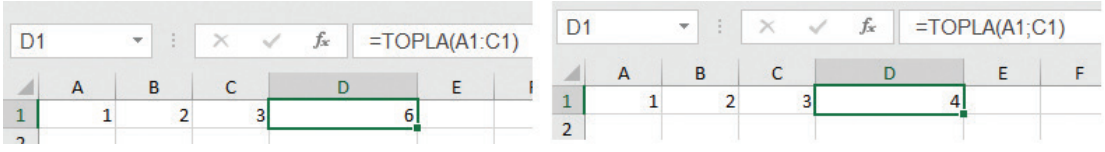
Operatörlerle oluşturulmuş formüllerde matematiksel işlem önceliği geçerlidir. Yani önce çarpma ve bölme sonra toplama ve çıkarma işlemi yapılır. Bu nedenle işlem önceliğine göre gerekli olan yerlerde "( )" (parantez) kullanılması gerekebilir. Örneğin "=(A1+B1)*(C1+D1)".

DİKKAT

Excel'de fonksiyonlar oluşturulurken matematiksel ve mantıksal operatörler kullanılmaktadır. Bunlarla beraber iki nokta ":", noktalı virgül ";", virgül "," gibi operatörlerde kullanılmaktadır. Mantıksal operatörler büyüktür ">", küçüktür "<", büyük eşittir ">=", küçük eşittir "<=", eşit değildir "<>" gibi operatörlerdir. Matematiksel operatörler sonuç olarak işlem sonucu döndürürken mantıksal operatörler doğru veya yanlış değerlerini döndürür. Noktalama işaretlerinden oluşan operatörler ise seçimin nasıl olacağı bilgisini verir, yani hücre adresleri arasına iki nokta ":" gelirse yazılan adreslerle beraber bu hücreler arasında kalan diğer hücreleri de formüle dahil eder, noktalı virgül ";" operatörü ise sadece yazılı olan hücreleri kabul eder. Örneğin; =A1:C1 işlemi A1+B1+C1 anlamına gelirken =A1;C1 formülü A1+C1 anlamına gelir. Formülde birden fazla hücre dizisi kullanmak gerektiğinde hem ":" hem de ";" kullanılabilir. Örneğin =TOPLA(A1:A3;A6:A9) formülü A1 ve A3 aralığındaki tüm hücrelerin toplamı ile A6 ve A9 aralığındaki hücrelerin toplamını toplayarak sonuç üretir.

Resim 1.12

TOPLA Fonksiyonunda İki Nokta ":" ve Noktalı Virgül ";" Kullanımı



Formül kullanarak Excel'de çarpım tablosu oluşturunuz.

SIRA SİZDE 5

Excel'de pek çok fonksiyon ön tanımlı olarak gelir. Kullanıcı isterse bu fonksiyonları kendi de oluşturabilir. TOPLA, ORTALAMA, ÇARPIM gibi fonksiyonlar kullanıcı tarafından kolaylıkla oluşturulabilecek fonksiyonlardandır fakat Excel'deki ön tanımlı fonksiyonların büyük çoğunluğu karmaşık formüller içerir ve kullanıcı tarafından oluşturulması oldukça zordur. İlerleyen bölümlerde Excel'deki ön tanımlı fonksiyonlara yer verilecektir.

DİKKAT

Excel'de formüller hücre içeriğine bağlıdır ve referans verilen hücredeki içerik değişirse formülün ürettiği çıktılar da değişecektir.

DİKKAT

Excel'de sonsuz döngü ya da döngüsel başvuru oluşturacak formüller mantık hatası olarak algılanır ve hesaplanamaz. Örneğin A1 hücresi üzerinde tanımlanmış bir formül yine A1 hücresindeki değeri kullanacaksa sürekli A1 hücresinin değeri güncellenmek zorunda kalacağından işlem otomatik olarak sonlandırılır.

Excel işlemleri bağımlılık sırasına göre yapılır. Örneğin A3 hücresinde A2 hücresine, A2 hücresinde A1 hücresine referans verilmişse önce A1 hücresi sonra A2 hücresi ve son olarak da A3 hücresi hesaplanır.

Excel karmaşık ve uzun süren hesaplamaları yaparken başka işlem daha yapması gerektiğinde hesaplamalara ara vererek önce diğer işlemi yapar sonra hesaplamalara kaldığı yerden devam eder.

Excel programında formüller çalıştırılırken standart olarak bağımlılık sırasına göre ve otomatik olarak çalıştırılır. Bazı durumlarda çok fazla ve karışık formüller varsa kullanıcı formüllerin çalıştırılmasını manuel kontrol edebilir. Excel formüllerin çalıştırılma şekline hesaplama moduna bakarak karar verir. Üç farklı mod seçeneği bulunmaktadır. Bunlar; otomatik, veri tabloları dışında otomatik ve El ile modlarıdır. Hesaplama Modu "El ile" ayarındaysa kullanıcı formüllerin çalışma sürecini kendi kontrol eder.

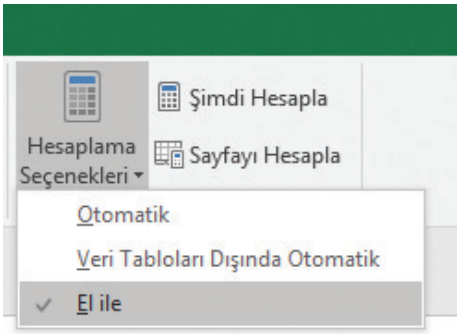
Hesaplama modu çalışma sayfası ya da çalışma kitabıyla ilişkili olmayıp doğrudan Excel programı ile ilişkilidir. Hesaplama modundaki değişiklik o anda açık tüm çalışma kitapları için geçerli olur. Sonradan açılan çalışma kitaplarında geçerli olmaz ve Excel varsayılan olarak çalışma modunu "Otomatik" olarak günceller.

DİKKAT

Hesaplamalar yapılırken referans verilmiş bir hücrenin değeri değiştirilirse hesaplamaların içeriği değişen hücrenin kullanıldığı yerden itibaren ya da baştan itibaren yeninden yapılması gerekir.

Resim 1.13

Excel Programında Hesaplama Seçenekleri



Excelde Referans Modları

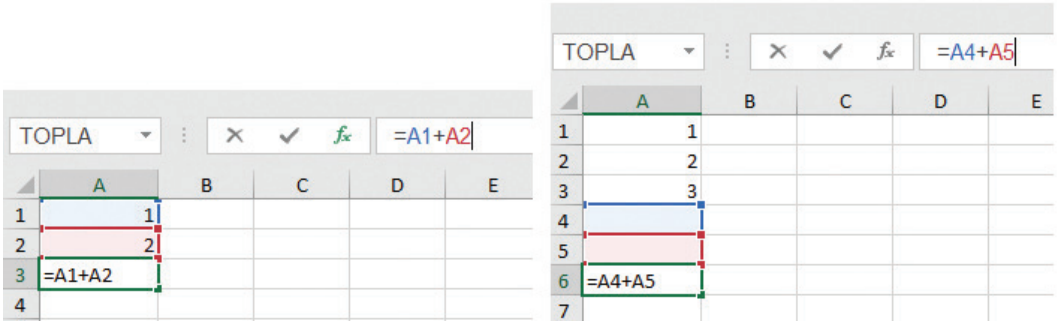
İşlem tablosu programlarında formüllerde diğer hücrelerin içerikleri kullanılabilir. Formüller kullanılırken referans verilen hücreye bağımlı çalışır. Formüllerin kopyalanması durumunda ise referansın nasıl olduğuna bağlı olarak referans verilen hücre sabit kalabilir ya da referans moduna göre güncellenebilir. Excel'de kullanılan referans modları A1 stiline göre aşağıdaki gibidir:

Göreli Referans: Göreli referansta formül yazılan hücre ile referans verilen hücreler arasında bağıl ilişki söz konusudur, yani formül bulunan hücre ile referans verilen hücre arasındaki hücre sayısı değişmeden formül kopyalanır. Aradaki hücre sayısının değişmemesini Excelin kendisi otomatik olarak sağlar. Örneğin; A1 hücresi ile A2 hücresindeki değerlerin toplamını verecek fonksiyon A3 hücresinde = A1+A2 şeklinde oluşturulmuş

olsun. Eğer A3 hücresindeki formül kopyalanıp A6 hücresine yapıştırılırsa A6 hücresindeki formülün açık hali =A4+A5 şeklinde olur.

Resim 1.14

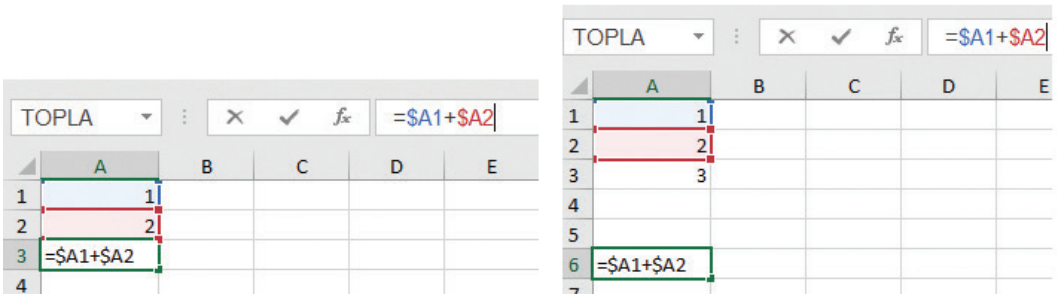
Göreli Referans Modunda Formül Kopyalama



Mutlak Referans: Oluşturulan formüller başka hücrelere kopyalansa bile formüldeki hücre adreslerinin değişmeden aktarıldığı referanslama şeklidir. Referanslama yapılırken satır ve sütun adreslerinin başına $ işareti konularak adresleme yapılır. Bu şekilde formül kopyalansa bile referans verilen hücre adresi değişmez. Örneğin A1 ve A2 hücrelerindeki değerlerin toplamını A3 hücresine =$A$1+$A$2 şeklinde bir formülle yazdırılıyorken A3 hücresinde tanımlanmış bu formül A6 hücresine kopyalanıp yapıştırılırsa A6 hücresindeki formül yine =$A$1+$A$2 şeklinde olacaktır.

Resim 1.15

Mutlak Referans Modunda Formül Kopyalama



Satır Mutlak Referans: Mutlak referansta hem satır hem sütun için göreli olma durumu engellenmekteydi. Satır mutlakta ise bir hücredeki formül kopyalanıp başka bir hücreye yapıştırıldığında sadece satır numarası aynı kalırken sütun numarası göreli olarak değişmektedir. Örneğin A1 ve A2 hücrelerindeki değerlerin toplamını A3 hücresine =A$1+A$2 şeklinde bir formülle yazdırılıyorken A3 hücresinde tanımlanmış bu formül B6 hücresine kopyalanıp yapıştırılırsa B6 hücresindeki formül =B$1+B$2 şeklinde olacaktır.

Sütun Mutlak Referans: Satır mutlak referanslama ile benzer şekilde işlem yapılır fakat satır mutlaktan farklı olarak, isminden de anlaşılacağı üzere bir hücredeki formül kopyalanıp başka bir hücreye yapıştırıldığında sadece sütun numarası aynı kalırken satır numarası göreli olarak değişir. Örneğin A1 ve A2 hücrelerindeki değerlerin toplamını A3 hücresine =$A1+$A2 şeklinde bir formülle yazdırılıyorken A3 hücresinde tanımlanmış bu formül B6 hücresine kopyalanıp yapıştırılırsa B6 hücresindeki formül =$A4+$A5 şeklinde olacaktır.

DİKKAT

A1 stiline göre yapılan referanslama ile R1C1 stiline göre yapılan referanslama tanımlamaları farklıdır. Ayrıca A1 stilinde göreli referans varsayılan referanslama modu olarak otomatik gelir. R1C1 stlinde ise mutlak referans varsayılan olarak tanımlanmıştır. Örneğin A1 stiline göre $A$6 şeklinde tanımlanan hücre R1C1 stilinde R6C1 olarak gösterilir.

Excel programında varsayılan A1 stili olduğundan R1C1 stilinin kullanımı standart kullanıcılar arasında pek yaygın değildir. Genellikle VBA makro programlama diliyle programlama yapılırken R1C1 stilinin kullanımı daha kolay olduğundan tercih edilmektedir.

Referans verilen çalışma kitabının ismi boşluk içeriyorsa yani "kitap 2" şeklinde bir isimlendirmesi varsa referans verilirken dosya adı tırnak içinde yazılır. Tırnak içerisinde yazılmadığında Excel dosya adının sadece ilk kelimesini okur ve ilk kelimeden sonra boşluk karakteri ile karşılaşınca formül yazımı standartlara uymadığından hata verir. Bu örnek için referans şu şekilde verilmedir; '[kitap 2.xlsx] Sayfa1!A1'.

R1C1 stilinde göreli referans verilmek istenirse satır ve sütun için ayrı ayrı görelilik tanımlanabilmektedir. R1C1 stilinde göreli tanımlaması köşeli parantez "[ ]" ile sağlanır. Örneğin R[3]C[5] ifadesinde 3 satır aşağı ve 5 sütun sağdaki hücreye referans verilmiş olur. Yukarı yönlü veya sola yönlü referanslama için eksi "-" işareti kullanılır. Örneğin R[-3]C[-5] ifadesinde 3 satır yukarıda ve 5 sütun soldaki hücreye referans verilmiş olur.

Excelde referans verme işlemi sadece aynı çalışma sayfasındaki hücrelere özgü değildir. Aynı çalışma kitabındaki farklı çalışma sayfalarındaki ve farklı çalışma kitaplarındaki çalışma sayfalarında yer alan hücrelere de referans verilebilir. Aynı çalışma kitabındaki farklı bir çalışma sayfasına referans ünlem işareti "!" ile sağlanır. Farklı bir çalışma kitabına ise dosya ismi köşeli parantez içerisine alınarak ulaşılabilir. Örneğin aynı çalışma kitabındaki Sayfa2 isimli farklı bir çalışma sayfasında yer alan A1 hücresine *=Sayfa2!A1* şeklinde bir kullanımla, Kitap2 isimli farklı bir çalışma kitabının Sayfa1 isimli sayfasındaki A1 hücresine ise *=[Kitap2.xlsx]Sayfa1!A1* şeklinde bir kullanımla erişilebilir.

SIRA SİZDE 6

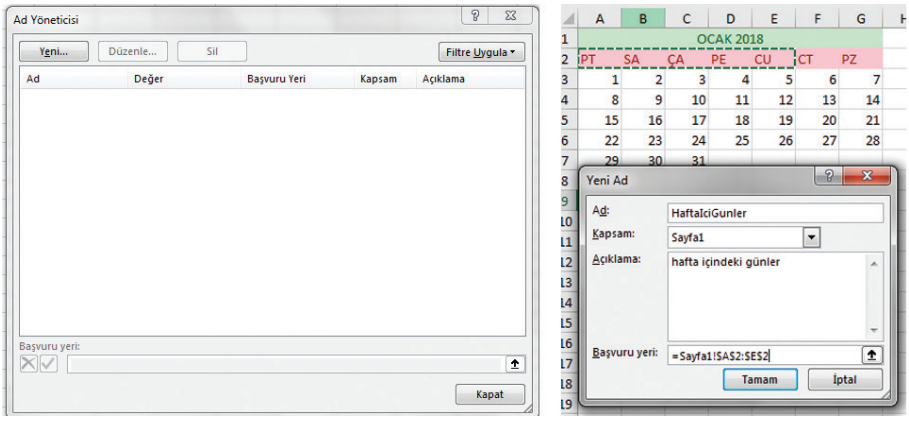
Sıra Sizde 5'te oluşturduğunuz çarpım tablosunu mutlak referans ve göreli referans kullanarak formülleri bir kez oluşturup daha sonra kopyalayıp yapıştırarak yapmaya çalışınız. Hangi satır ve sütunlar için hangi referans modunu kullandığınıza dikkat ediniz.

Excelde Özel İsimlendirme

Excel programında hücreler için nasıl özel isimlendirme yapıldığı konusuna değinilmişti. Excelde sadece hücreler için değil hücre dizileri, hücre grupları, satır ve sütunlar, grafikler gibi bileşenler içinde özel isimlendirme yapılabilmektedir. Özel isimlendirmeler Ad Yöneticisi arayüzünden yapılmaktadır. Bu arayüze Formüller menüsünden erişilebilir. Excelde özel isimlendirmeler mutlak olarak tanımlanır ve özel isimlendirilmiş yapılara ulaşmak daha kolaydır. Özel isimlendirilmiş yapılarla çalışmak programlama yaparken daha okunabilir ve anlaşılabilir programlama yapılmasını sağlar.

Resim 1.16

Özel İsimlendirme Kullanımı



Özel isimlendirme kullanılarak aylık gelir giderler arasındaki farkın hesaplanacağı bir senaryoda A1 hücresinden A4 hücresine kadar aylık gelirler, B1 hücresinden B8 hücresine kadar da aylık giderlerin yazılı olduğunu varsayalım. Standart kullanımda =TOPLA(A1:A4)-TOPLA(B1:B8) yazılması gerekirdi. Gelirlerin yer aldığı hücre dizisi AylikGelirler ve giderlerin olduğu hücre dizisi de AylikGiderler olarak isimlendirilse formül =TOPLA(AylikGelirler)-TOPLA(AylikGiderler) olarak oluşturulur. Örnekten de anlaşılabileceği gibi özel isimlendirme kullanıldığı karmaşık bir program yazıldığında formül okunduğu anda anlaşılabilir bir duruma gelecek ve kodlaması daha kolay olacaktır.

Özel isimlendirme mutlak referans verir. Yani formül kopyalanıp başka bir yere yapıştırılsa ya da formülde referans verilen hücrelerin soluna veya üstüne yeni bir hücre eklendiğinde formül bu değişikliği otomatik olarak algılayacak ve kendini güncelleyerek olası hataları engellemiş olacaktır.

DİKKAT

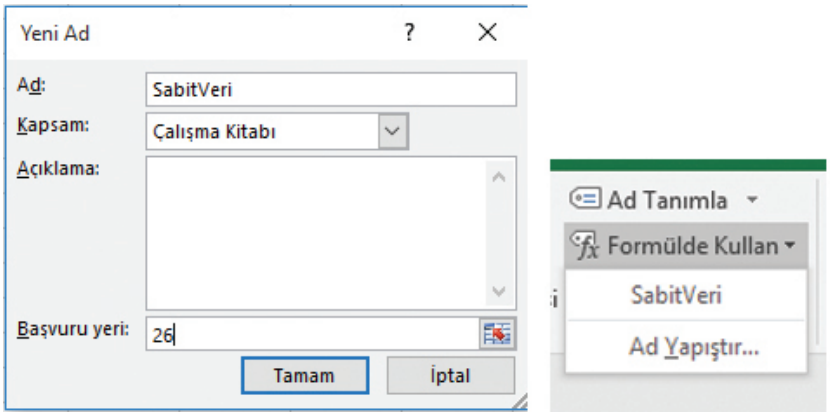
Özel isimlendirmeler çalışma sayfasında istenilen bir anda tanımlanabilir. Fakat Excel bu isimlendirmeyi otomatik olarak uygulamaz. Kullanıcının sonradan yaptığı özel isimlendirmelerin formüller içerisinde uygulanması istenirse Formüller menüsünden *"Adları Uygula"* butonuna basılması gerekir.

Excelde özel isimlendirme ile beraber kesiştirme işlemleri de yapılabilmektedir. Kesiştirme işlemleri boşluk karakteri ile tanımlanır. Kullanımı =birincihücredizisi ikincihücredizisi şeklindedir. Örneğin Güney ve Doğu olarak tanımlanmış iki hücre dizisinin kesişimi = Güney Doğu şeklinde tanımlanır.

Excel programında formüller her zaman hücrelere referans verilerek oluşturulmaz. Formüller hücrelerden bağımsız olarak sabit sayılarla ya da sabit içerikli formüllerle de oluşturulabilir. Bunun için formüller menüsünden Ad Tanımla diyerek açılan pencerede sabitin ismi tanımlanır ve Başvuru Yeri alanına istenilen sabit bilgisi girilir. Bu sabite istenildiği zaman Formüller menüsünden Formülde Kullan butonuyla ulaşılır.

Resim 1.17

Sabit Veri Tanımlaması



Sabit tanımlaması aynı zamanda özel isimlendirmedir ve özel isimlendirme sadece sabitler tanımlanmaz. Aynı arayüzünden yani Ad Tanımlama penceresinden farklı fonksiyonlarda tanımlanabilir. Bu fonksiyonlar hücre bağımsız işlemler olabileceği gibi hücrelerin toplamı, çarpımı, ortalaması gibi formüller de olabilir.

DİKKAT

Ad tanımlaması yapılırken başvuru yeri olarak hücre ya da hücre dizisi tanımlanıyorsa referans moduna dikkat edilmelidir. Mutlak referans vermek için A1 stilinde $ işareti satır ve sütun numaralarının önüne yazılmalıdır.

DİKKAT

Ad tanımlanırken Kapsam çalışma kitabı olarak seçilirse oluşturulan formülde çalışma sayfasının ismi de yer almalıdır. Aksi halde Excel formülü düz metin olarak algılayabilecektir.

Formül Denetleme

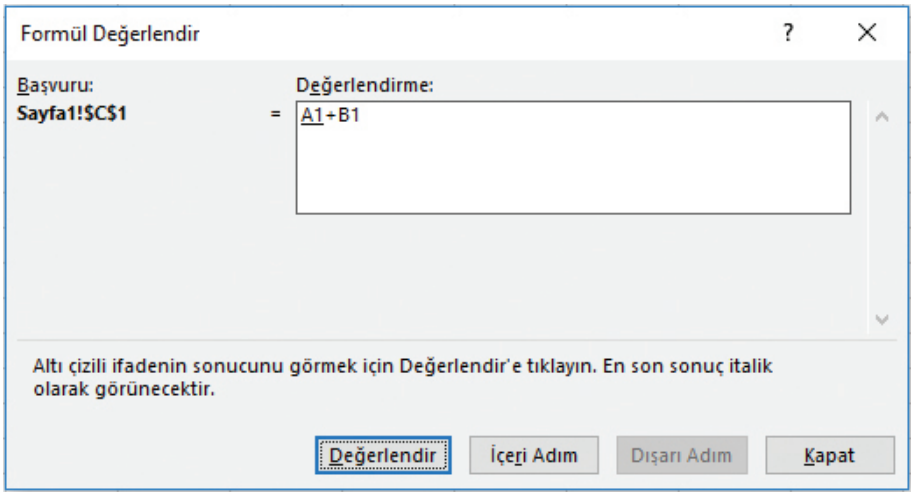
Excel programında formüller yazılırken dikkatli olmak gerekir. Çünkü formüller oluşturulurken standartlara uygun olup olmadıkları kontrol edilir fakat çalıştırılma aşamasında hata verebilirler. Örneğin kullanıcı formül tanımlaması yaparken boş bir hücreye bölme işlemi ya da değeri sıfır olan bir hücreye bölme işlemi yapmaya çalışıyor olabilir. Excel formül oluşturulurken hücrenin değeri ile ilgilenmez ve formülü oluşturur. Excel programında oluşturulan formüller denetlenebilmekte ve Excel'in verdiği hata kodları ve açıklamasına bakılarak çözüm üretilebilmektedir. Excel programında formül denetleme için Formüler menüsünden Formül denetleme butonuna tıklanarak hatalar araştırılabilir ve hatalara çözüm üretilebilir.

Formüllerde hata olması halinde bu hatanın kaynağını ve bu kaynak hücreyi kullanan diğer hücreleri görebilmek mümkündür. Hata izleme sadece kaynak hücre üzerinden yürütülmez, hatanın alındığı herhangi bir hücre üzerinden de hata takip edilebilir. Excel programı hatalı hücreleri oklar yardımıyla göstermektedir. Bunun için ilk olarak hatalı hücre seçilir ve sonra farenin sağ tuşuna tıklanarak Etkileyenleri izle komutu verilir. Bu komutu aynı zamanda Formüller menüsünden de çalıştırmak mümkündür. Etkileyenleri izle komutu verildikten sonra Excel hücrede hata varsa bu hücredeki hataya sebep olan diğer hücreleri oklar yardımıyla gösterir, bu oklar Okları Kaldır seçeneği ile kaldırılabilir.

Formülleri Göster, çalışma sayfasındaki tüm formülleri göstermek için kullanılır. *Hata denetimi* yapılarak çalışma sayfasındaki tüm hatalı formüller tespit edilir. Excel'de formül çalıştırılınca hücrenin değerinin ne olacağını görebilmek için *Formülü Değerlendir* kullanılır. Bir veya birden fazla hücrenin değerinin ne olduğunu görebilmek için *Gözcü Penceresi* kullanılabilir.

Resim 1.18

Formül Değerlendir Penceresi



Excel'de çalışırken en önemli konulardan biride hataları ayıklamaktır. Excel programı formüllerde oluşan her hata için bir hata kodu üretir ve bunu kullanıcı ile paylaşır. Bu hata kodlarının doğru yorumlanabilmesi hatanın çözümü önemlidir. Excel programında sıklıkla alınan hata kodlarının bir kısmı aşağıda verilmiştir:

#SAYI/0! Boş bir hücre içeriğine bölme yapılmaya çalışılması veya sayının sıfıra bölünmeye çalışılması durumunda alınır.

#YOK Aranan değerin bulunamadığı durumlarda alınır.

#BOŞ! Excelde kesişmeyen iki alanın kesişim sonucu istendiği zaman bu hata alınır.

#AD? Hatalı formül tanımlarından bu hata kodu alınır. Örneğin =TOPLA(A1:A5) yerine =TOPLAMI(A1:A5) formülü yazılırsa Excel bu hata kodunu oluşturur.

#SAYI! Matematiksel ifadeler ve bunların sonucunda mantıksal hatalar oluşursa bu hata kodu oluşturulur. Negatif parametre almayan =KAREKÖK fonksiyonu içerisine negatif bir sayı alırsa bu hata oluşur.

#BAŞV! Formülde kullanılan bir hücreye ulaşılamaması durumunda bu hata alınır. Sıklıkla göreli referans verilen hücrelerin kopyalanıp yapıştırılması sonucu bu hata alınır. Örneğin; =TOPLA(A1:A5) formülü D1 hücresinden C1 hücresine kopyalanırsa A1 den önceki hücreye referans verilmiş olacağından Excel formülde doğru tanımlanmamış hücre olduğunu bildirmek için #BAŞV! Hata kodunu oluşturur.

#DEĞER! Formülün yapısına uymayacak şekilde farklı türdeki verilerin olduğunu ifade eder. Örneğin iki metin ya da bir sayı bir metin toplanmaya çalışılırsa bu hata kodu oluşur.

Formüllerde alınan hataların ne anlama geldiğini bilmek çok önemlidir. Bu sayede hatalara hızlı çözüm üretilebilir.

DİKKAT

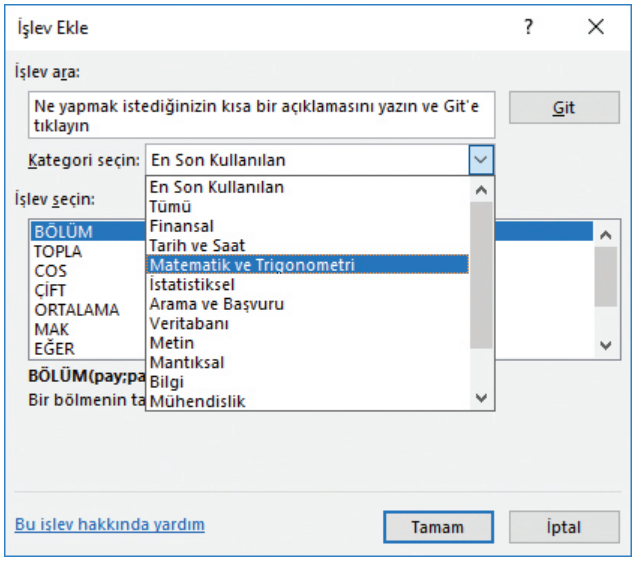
Excelde bazen ##### şeklinde bir değer görünebilir. Bu hata değildir. Sadece formül ya da hücrenin değerinin hücreye sığmayacak kadar büyük olduğunu belirtir. Bu durumda hücrelerin genişliği artırılarak hücre içeriği görüntülenebilir.

SIRA SİZDE 7

Hatalı formül oluşturup formülün bulunduğu hücreyi peş peşe en az iki hücrede daha kullanınız. Örneğin A1 hücresine 5 değerini giriniz sonra A3 hücresine =A1/(A1-5) yazınız. Daha sonra C2 hücresine =A3+2 yazınız ve sonra D5 hücresine =C2/2 yazınız. Çalışma sayfasındaki hücrelerin değerlerini gözlemleyiniz. Sonra hata denetimi yaparak etkileyenleri izleyiniz.

ÖN TANIMLI FONKSİYONLAR

Excelde kullanıcılar formül oluşturup bunları kullanabilirler fakat her formülün oluşturulması çoğu zaman mümkün olamamaktadır. Excel en basitinden en karmaşığına kadar pek çok formülü önceden oluşturmuş ve kullanıcılara sunmuştur. Bu formüller çok fazla olduğundan birden fazla kategoride toplanmışlardır. Kullanıcı kategori seçerek istediği formülü kullanabilir. Bunlar Matematiksel, Mühendislik, Tarih-Saat gibi kategorilerdir. Bu fonksiyonların genel kullanımı =FONKSIYON (parametre1,parametre2, ...) şeklindedir. Excel her fonksiyonun nasıl kullanılacağını fonksiyon çalıştırma penceresinde göstermektedir. Ayrıca fonksiyon oluşturmak için ise kullanıcıyı yönlendirecek arayüzler oluşturmuştur.



Resim 1.19

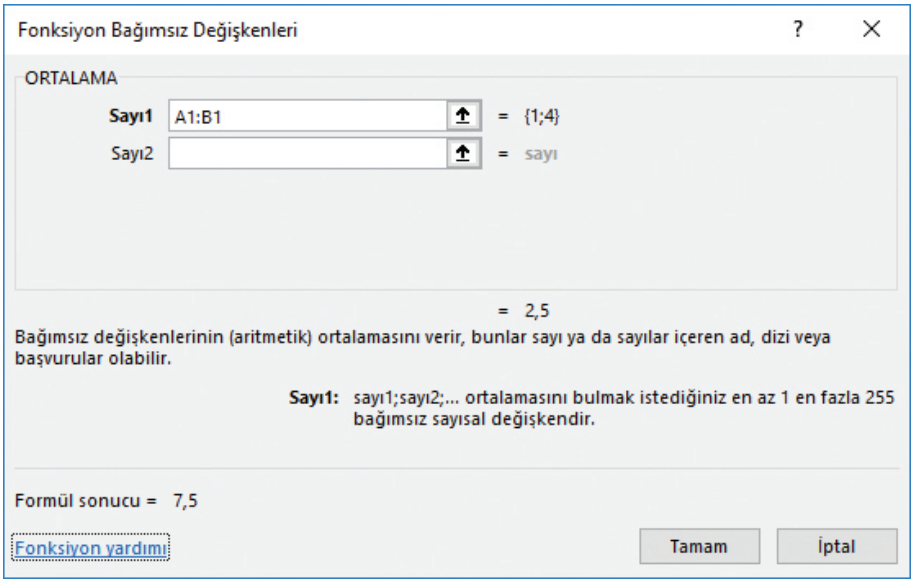
Excelde Ön Tanımlı Fonksiyonlar ve Fonksiyon Kategorileri

Seçilen fonksiyonun nasıl kullanılacağı bilgisi İşlev Ekle arayüzünde kısaca anlatılmaktadır.

Excel programında ön tanımlı fonksiyonlara erişmek için *fx* butonu kullanılabilir. Bu butona basıldığında fonksiyonlar penceresi açılır ve bu pencereden kullanıcı kullanmak istediği fonksiyonu seçer. Bu pencerede son kullanılan fonksiyonlar üstte yer alır. Excel ilk kez çalışıyorsa yada fonksiyon penceresi ilk kez kullanılıyorsa bu pencerede kullanıcıların sıklıkla kullandığı fonksiyonlar yer alır. Bu fonksiyonlar =TOPLA, =ORTALAMA gibi fonksiyonlardır. Kullanıcı kullanmak istediği fonksiyonu ister kategori seçerek isterse tüm kategorileri görüntüleyerek seçebilir.

Resim 1.20

Fonksiyon Oluşturma Penceresi



Kullanılacak olan fonksiyon seçildikten sonra fonksiyon girdilerinin tanımlanacağı bir pencere açılır. Bu pencerede fonksiyon girdileri sabit değerler, hücreler veya hücre dizileri olabilir. Açılan bu pencere fonksiyona özeldir ve fonksiyon hakkında tanımlama içerir. Ayrıca fonksiyonun doğru oluşturulması içinde kullanıcıyı yönlendirir.

Excelde sıklıkla kullanılan fonksiyonlar kategorileriyle birlikte aşağıda verilmiştir.

Matematik ve Trigonometri

TOPLA fonksiyonu iki veya daha fazla hücredeki sayıların toplamını bulmak için kullanılır.

ÇARPIM fonksiyonu iki veya daha fazla hücredeki sayıların çarpımı bulmak için kullanılır.

ORTALAMA fonksiyonu iki veya daha fazla hücredeki sayıların aritmetik ortalamasını bulmak için kullanılır.

MAK ve MİN fonksiyonları hücre dizisi veya seçili hücrelerdeki en büyük ve en küçük değerleri bulmak için kullanılır.

YUVARLA fonksiyonu girdi olarak hücre değerine ilaveten ondalık sayısını alır. Girilen ondalık basamak sayısına göre verilen sayıyı yuvarlar.

TAMSAYI fonksiyonu verilen değeri en yakın alt tam sayıya yuvarlar.

OBEB ve OKEK fonksiyonları girilen sayıların ortak bölenlerinin en büyüğünü ve ortak katlarının en küçüğünü veren fonksiyonlardır.

KAREKÖK fonksiyonu girilen sayının karekökünü verir.

KUVVET fonksiyonu girdi olarak üs değeri alır ve girdi olarak verilen değerin üs değerini hesaplar.

MUTLAK fonksiyonu bir sayının mutlak değerini verir.

Mantıksal

Excelde mantıksal işlemler için kullanılan fonksiyonlar mantıksal kategorisinde toplanmışlardır. EĞER fonksiyonu Excel'de çok sık kullanılan mantıksal operasyondur. Kullanım mantığı eğer şu ifade doğruysa bu sonucu döndür değilse şu sonucu döndür şeklindedir. Excel'de yazımı EĞER(Koşul, doğruysa üretilecek sonuç, yanlışsa üretilecek sonuç) şeklindedir. Örneğin EĞER (A1>=0; Pozitif;Negatif) kullanımının tanımlı olduğu hücreye eğer A1 hücresindeki değer sıfırdan büyük ise Pozitif değilse Negatif değeri yazılacaktır.

Eğer fonksiyonu mantıksal sınama yapar ve bu sınamayı yaparken (<, >, <=, >=, =, <>) operatörlerinden birini kullanabilir.

DOĞRU ve YANLIŞ fonksiyonları mantıksal doğru veya mantıksal yanlış sonuçlarını döndürür. Birden fazla sınamada VE kullanımı sınanan tüm değerler doğru ise doğru en az biri yanlış ise yanlış değerini döndürür. YADA kullanımı ise yapılan sınamalardan en az biri doğru ise doğru, hiç doğru yok ise yanlış sonucunu döndürür.

Metin

Metinlerle çalışılırken kullanılacak fonksiyonlar bu kategoride toplanmıştır. Sıklıkla kullanılan fonksiyonlar; verilen metnin tamamını büyük harflere dönüştüren BÜYÜKHARF, verilen metni küçük harflere dönüştüren KÜÇÜKHARF fonksiyonları, girilen değeri metin haline dönüştüren METNEÇEVİR fonksiyonu ve birden fazla metni tek metin haline getiren BİRLEŞTİR fonksiyonudur. BİRLEŞTİR fonksiyonu noktalı virgüllerle ayrılmış metinleri girdi olarak alarak onları birleştirir.

Tarih ve Saat

Tarih ve saatle ilgili fonksiyonlar bu kategoride tanımlanmıştır. Tarih fonksiyonu yıl, ay ve gün girdilerini alarak tarih sonucunu üretir. Otomatik olarak tarih düzeltmesini yapar. Örneğin =TARİH(2017,08,40) değeri girildiğinde fonksiyon otomatik olarak gün parametresindeki 40 değerini Ağustos ayı 31 gün olduğu için bir sonraki aya ilave ederek 9 Eylül 2017 tarihini döndürür.

Tarih fonksiyonu sistem saatini kullanarak değer döndürür. Bu nedenle doğru sonuçlar elde edilebilmesi için sistem saatinin doğru çalışıp çalışmadığına bakılması önemlidir.

GÜNSAY fonksiyonu iki tarihi girdi olarak alır. Tarihler TARİH fonksiyonunu kullanılarak da parametre olarak verilebilir. GÜNSAY fonksiyonu ile başlangıç ve bitiş tarihi arasındaki toplam gün sayısına ulaşılır.

Excelde tarih ve saatler birer sayıyı ifade eder. Saat kavramı Excelde 0 ile 1 arasında bir sayıya karşılık gelir. Örneğin saat 00:00:00 Excel programında 0 ile ifade edilirken 23:59:59 ise 0,999988426 ile ifade edilir.

ZAMANSAYISI fonksiyonu saat değerini girdi olarak alarak Excelin o saat için kullandığı sayıyı çıktı olarak verir. Örneğin Türkçe 13:36:26 için =ZAMANSAYISI("13:36:26") fonksiyonu Türkçe 0,566967593 değerini döndürür.

TARIHSAYISI fonksiyonu tarih değerini alarak Excel programında tarih yerine kullanılan sayıyı çıktı olarak verir. Örneğin, 13 Mayıs 2016 tarihi =TARIHSAYISI("2016/05/13") şeklinde tanımlandığında çıktı olarak 42503 değerini verir.

SAAT, DAKİKA, SANİYE fonksiyonları girdi olarak zaman metnini veya Excel programının ürettiği zaman sayısını alır ve saat, dakika ve saniyeyi geri döndürürler.

GÜN, AY, YIL fonksiyonları girdi olarak tarih metnini veya Excel programının ürettiği tarih sayısını alır ve gün, ay ve yılı geri döndürürler.

BUGÜN fonksiyonu girdi almaz ve içinde bulunulan günün tarih bilgisini çıktı olarak verir.

ŞİMDİ fonksiyonu girdi almaz ve içinde bulunulan tarihi ve saati çıktı olarak verir.

DİKKAT

BUGÜN ve ŞİMDİ fonksiyonları sistem saatini kullanılırlar, sistem saatinin hatalı olması bu fonksiyonların hatalı sonuç üretmesine sebep olur.

Finansal

Finansal hesaplamalarda kullanılan fonksiyonlar bu kategoride toplanmıştır. Excel programıyla çok fazla finansal hesaplama yapılabilmektedir. Finansal hesaplamalar için 50 den fazla fonksiyon bulunmaktadır. Örneğin FAİZ_ORANI fonksiyonu finansal kategorisinde bulunan bir fonksiyondur ve kullanımı *=FAİZ_ORANI(dönemsayısı; devreselödeme; bd; gd; tür; tahmin) şeklindedir.*

Özet

İşlem tablosu uygulamalarının işlevleri, özellikleri ve tarihsel gelişimini hakkında bilgi sahibi olacak

İşlem tablosu programları matematiksel ve mantıksal işlemlerin belirli bir yapı içerisinde, satır ve sütunlar biçiminde tasarlanmış hücreler kullanılarak kolay, hızlı ve fonksiyonel bir şekilde yapılması sağlar. İlk işlem tablosu programları 1970li yılların sonlarında VisiCalc ve SuperCalc programları ile başlamıştır. Daha sonra Lotus 1-2-3 ile işlem tablosu programları yaygınlaşmaya başlamıştır. Microsoft firması da 1985 yılından itibaren Excel ile işlem tablosu programlarına giriş yapmıştır. Özellikle 1990 yılından sonra Excel işlem tablosu programları arasında hakimiyet kazanmıştır.

Günümüzde işlem tabloları pek çok işlemi yerine getirebilmektedir. Veriler üzerinde mantıksal ve matematiksel işlemler yapılması, verilerle temel hesaplamaların yapılması, verilerin tablolar halinde saklanması, makroların oluşturulabilmesi, grafiklerin hazırlanabilmesi, otomatik rapor üretilebilmesi, veri tabanı ve internet bağlantıları kurulabilmesi gibi pek çok farklı işlem yapılabilmektedir.

İşlem tablosu programlamasının temel kavramlarını tanımlayabilecek

İşlem tablosu programlarının temelinde nesne yapısı vardır. İşlem tabloları nesnelerle çalışmaktadır. Bu nesneler üzerinde elle ya da otomatik olarak değişiklikler yapılabilmektedir. İşlem tablosu programları satır ve sütunlardan oluşan hücre yapısı kullanırlar. Hücreler veriler veya formüller içerebilir. Hücreler içerikleri bakımından sayısal değer içeren, metin içeren, mantıksal sınama içeren ve formül içeren hücreler olarak gruplandırılabilirler. Formül içeren hücrelerde sabit verilerle beraber matematiksel, mantıksal, metin ve aralık operatörleri de bulunabilir. Formül girişi yapılırken hücrelere referans verilebilmektedir, bir başka deyişle formüllerde farklı hücrelerin içeriği kullanılabilmektedir. Referans verilirken mutlak, satır mutlak, sütun mutlak ve göreli referans kullanılabilmektedir.

Çalışma sayfasındaki hücreler farklı stillerle kullanılabilmektedir. Bunlar A1 stili, R1C1 stili veya özel isimlendirme kullanımıdır. A1 stilinde satır ve sütun ismi birleştirilerek, R1C1 stilinde satır ve sütun numarası birleştirilerek hücre isimlendirmesi yapılır. Hücre veya hücre gruplarına özel isimlendirme de yapılabilmektedir.

Excel işlem tablosu programında formül oluşturmayı ve ön tanımlı fonksiyonları kullanmayı öğrenebilecek

Excelde hücreleri kullanarak formüller oluşturulabilmektedir. Formül oluşturulurken hücre seçilerek eşittir karakteri yazılıp yapılması istenen işlem girilmektedir. Örneğin toplama işlemi yapılacaksa =A1+A2 yazılırsa bu formülün yazıldığı hücrenin değeri A1 ve A2 hücresindeki değerlerin toplamı olacaktır.

Formülleri elle oluşturmak yerine ön tanımlı fonksiyonlar da kullanılabilmektedir. Excelde farklı kategorilerde ön tanımlı fonksiyonlar oluşturulmuştur. Bu fonksiyonlar genel olarak =FONKSİYONADI(parametre1,parametre2,..) şeklindedir. Örneğin toplama işlemi için TOPLA fonksiyonu ön tanımlı fonksiyondur ve =TOPLA(A1:A2) şeklinde kullanılmaktadır.

Kendimizi Sınayalım

1. Microsoft firması, Office programları isimlendirme standardı nedeniyle hangi Excel sürümünü yayınlamamıştır?

a. Excel 4
b. Excel 5
c. Excel 6
d. Excel 7
e. Excel 8

2. Microsoft Office ilk Excel sürümlerinde hangi makro yazılım dilini kullanıyordu?

a. XML
b. XLM
c. VBA
d. Java
e. C

3. Çalışma sayfasında yapılan aşağıdaki işlemlerden hangisi geri alınamaz işlemlerdendir?

a. Sayfanın silinmesi
b. Satırın silinmesi
c. Sütünün silinmesi
d. Sayfanın korumaya alınması
e. Satır renginin değiştirilmesi

4. Hücreler içerikleri bakımından aşağıdakilerden hangisi gibi sınıflandırılmaz?

a. Sayısal bir değer içeren hücreler
b. Metin içeren hücreler
c. Formül içeren hücreler
d. Mantıksal sınama içeren hücreler
e. Renk içeren hücreler

5. R1C1 stilinde tanımlanmış olan R[4]C1 hücresi hangi hücreyi referans vermektedir?

a. 4 numaralı satır 1 numaralı sütunda bulunan hücreyi
b. 4 numaralı satır 1 sütun sağda bulunan hücreyi
c. 4 satır yukarıda 1 numaralı sütunda bulunan hücreyi
d. 4 numaralı satır 1 sütun solda bulunan hücreyi
e. 4 satır aşağıda 1 numaralı sütunda bulunan hücreyi

6. Aşağıdaki fonksiyonlardan hangisi doğru yazılmıştır?

a. = A0:A1+B0:B1
b. = EĞER(0, DOĞRU, YANLIŞ)
c. = TOPLA(A$1;20)
d. = ÇARP(A$1:$C20)
e. = A$1(B1+$C1D$1)

7. İşlem tablosu çalışma sayfasında A2 hücresindeki *=MAK(B1:B10)* formülü C5 hücresine taşındığında, formül aşağıdakilerden hangisine dönüşür?

a. Formül *=MAK(B1:B10)* olarak kalır.
b. Formül =MAK(D4:D13) olarak değişir.
c. Formül =MAK(B4:D13) olarak değişir.
d. Formül =MAK(B4:B13) olarak değişir.
e. Formül =MAK(D1:D10) olarak değişir.

8. =KAREKÖK(-1) fonksiyonu aşağıdaki hatalardan hangisini döndürür?

a. #SAYI!
b. #SAYI/0!
c. #AD?
d. #BAŞV!
e. #YOK

9. =TOPLAM(A1:A3) formülü yazılırsa Excel hangi hata kodunu oluşturur?

a. #SAYI!
b. #SAYI/0!
c. #AD?
d. #BAŞV!
e. #YOK

10. Excel programında bir hücreye *=EĞER(3<5;"3+5";"5-3")* formülü yazılırsa, hangi çıktı üretilir?

a. 8
b. 2
c. 3+5
d. 5-3
e. #AD?

Kendimizi Sınayalım Yanıt Anahtarı

1. c	Yanıtınız yanlış ise "Tarihçe" konusunu yeniden gözden geçiriniz.
2. b	Yanıtınız yanlış ise "Tarihçe" konusunu yeniden gözden geçiriniz.
3. a	Yanıtınız yanlış ise "Excel Nesneleri" konusunu yeniden gözden geçiriniz.
4. e	Yanıtınız yanlış ise "Excel Nesneleri" konusunu yeniden gözden geçiriniz.
5. e	Yanıtınız yanlış ise "Hücre Adresleme Modları" konusunu yeniden gözden geçiriniz.
6. b	Yanıtınız yanlış ise "Fonksiyonlar" konusunu yeniden gözden geçiriniz.
7. b	Yanıtınız yanlış ise "Excelde Referans Modları" konusunu yeniden gözden geçiriniz.
8. a	Yanıtınız yanlış ise "Formül Denetleme" konusunu yeniden gözden geçiriniz.
9. c	Yanıtınız yanlış ise "Formül Denetleme" konusunu yeniden gözden geçiriniz.
10. c	Yanıtınız yanlış ise "Ön Tanımlı Fonksiyonlar-Mantıksal" konusunu yeniden gözden geçiriniz.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1

Excel programında Şerit, formül çubuğu, satır ve sütun başlıkları, hücreler, durum çubuğu, aktif hücre referans alanı gibi farklı alanlar mevcuttur. Şeritte kategorilere ayrılmış menüler bulunmaktadır. Bu menüler artırılıp azaltılabilir. Ayrıca yeni grup ve menüler oluşturulabilir. Bu işlem şeriti özelleştirmektir. Şerit, Excel seçenekleri penceresinden şeriti Özelleştir sekmesi kullanılarak özelleştirilir.

Sıra Sizde 2

Excelde sayısal veri içeren, metin içeren, formül içeren ve mantıksal sınama içeren hücreler bulunur. Sayısal veri içeren hücreye örnek olarak A1 hücresine 1982 ve A2 hücresine 2017, metin içeren hücreye örnek olarak B1 hücresine Eskişehir, formül içeren hücreye örnek olarak B2 hücresine =A2-A1 ve mantıksal sınama içeren hücreye örnek olarak B3 hücresine =B2>0 yazılabilir.

Sıra Sizde 3

Excel'de hücrelere özel isimlendirme yapmak için önce hücre veya hücre grubu seçilir. Sonrasında farenin sağ tuşuna tıklanarak açılan pencereden Ad Tanımla seçilerek özel isim verilir. Özel isimli hücre seçildiğinde Resim 1.9 daki gibi aktif hücre referansı alanında verilen özel isim yer alır. Farklı bir hücre seçili iken de Aktif Hücre Referansı alanından özel isimli hücre seçildiğinde çalışma sayfasında özel isimlendirilmiş hücreler seçili konuma gelir.

Sıra Sizde 4

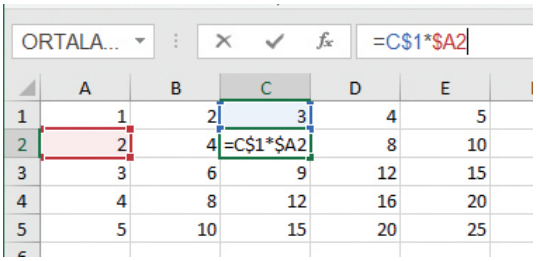
Excel'de adresleme modu varsayılan olarak A1 stilidir. A1 stili R1C1 stiline dönüştürülmek istenirse Excel Seçenekleri penceresindeki Formüller sekmesinde yer alan Formüllerle çalışma alanından R1C1 başvuru stili kutusu seçilir. İki stil arasında gözle görülen farklardan birincisi sütunlarda harf yerine rakam kullanılmasıdır. Diğer fark ise Aktif hücre referansı alanındaki aktif hücre adreslerinin gösterimidir.

Sıra Sizde 5

Excel de çarpım tablosu yapmak için A1 den A5 e kadar ve A1 den E1 e kadar olan hücrelere 1,2,3,4,5 rakamları girilir. Sonra B2 hücresi B1 hücresi ve A2 hücresinin çarpımı olduğundan bu hücreye = B1*A2 yazılır. Benzer şekilde tüm hücreler doldurulur.

Sıra Sizde 6

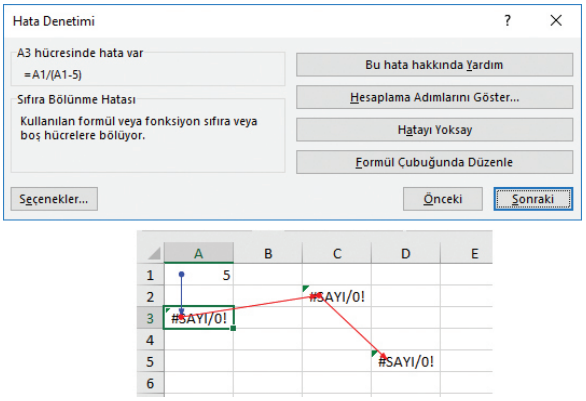
Sıra Sizde 5'te yapılan çarpım tablosunda satır mutlak ve sütun mutlak referans modu aynı anda kullanılarak formül oluşturulduğunda bu formül kopyalanıp tüm hücrelere yapıştırılabilir ve çarpım tablosu tamamlanmış olur. Resim de formül ve çarpım tablosu yer almaktadır.



Resim 1.21: *Satır mutlak ve sütun mutlak referans kullanarak çarpım tablosu oluşturma*

Sıra Sizde 7

İstenilen hücrelere istenilen değerler girildiğinde #SAYI/0! Hatası alınacaktır. Hata denetimi yapılıp etkileyenler izlendiğinde aşağıdaki görüntü oluşacaktır. Hatanın giderilmesi için A1 hücresinin değerinin değiştirilmesi yeterli olacaktır.



Resim 1.22: *Hata denetimiyle etkileyenlerin izlenmesi*

Yararlanılan ve Başvurulabilecek Kaynaklar

https://msdn.microsoft.com/

2

Amaçlarımız

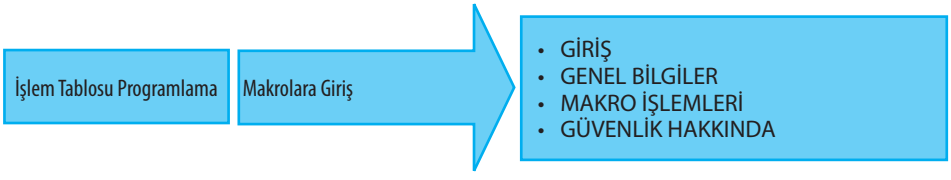
Bu üniteyi tamamladıktan sonra;

- Makrolar hakkında temel bilgi sahibi olabilecek
- Temel makro işlemlerini tanımlayabilecek
- Makrolarda güvenlik kavramını açıklayabileceksiniz.

Anahtar Kavramlar

- Makro
- Makro Temelleri
- Makroların Avantaj ve Dezavantajları
- Makro Oluşturma
- Makro Görüntüleme
- Makro Güncelleme
- Kişisel Makro Çalışma Kitabı
- Makro Referans Modları
- Makro Kısayol Tuşu
- Makro Hızlı Erişim Butonu
- Makro Gizleme
- Vba Makro Yazılımı
- Makrolarda Güvenlik
- Excel Dosya Çeşitleri

İçindekiler



Makrolara Giriş

GİRİŞ

İşlem tablosu programlarında işlemler genel olarak bir kez yapılır fakat bazı durumlarda aynı işlemin birden fazla tekrarlanması gerekebilir. Tekrarlanması gereken işlem tek bir işlem olabileceği gibi birbirini takip eden işlemler bütünü de olabilir. Örneğin aylık gelir gider hesaplaması yaparken, gelir sütundaki tüm hücrelere bakıp değer boş değilse bunları toplayarak toplam gelir hücresine yazan, benzer şekilde giderleri toplayıp toplam gider hücresine yazan, sonrasında gelir-gider dengesini hesaplayıp bunları grafiklere döken ve raporlamaya hazır hale getiren bir işlemler bütününü her ay tekrar etmek gerekebilir. Aynı işlemleri sürekli tekrar etmek yerine bu işlemleri bir kez yaptıktan sonra yapılan işlemleri kaydedip her ay bu kaydı çalıştırmak daha kolay olacak ve daha az hata ile işlemlerin bitirilmesini sağlayacaktır. Bahsedilen kaydın alınması ve tekrar tekrar çalıştırılabilmesini makrolar sağlamaktadır.

Makrolar bir fonksiyon ya da formül değildir. Önceden kaydedilmiş işlemleri istenildiği zaman tekrar eden komut setleridir. Makrolar, genel olarak işlem tablosu programında klavye hareketleri ve komutları işlem sırasına göre kaydeder. Daha sonra bu sırada aynı klavye hareketlerini yapar ve komutları çalıştırır.

Makrolar kayıtlı işlemleri tekrar ettiğinden ilk oluşturulduğunda işlemlerin doğru bir şekilde yapıldığından emin olunması gerekir. Hata barındıran işlemler makrodan her yürütüldüğünde aynı hatalar tekrarlanmış olacaktır.

DİKKAT

Makrolar hem basit işlemler için hem de karmaşık işlem setleri için pek çok avantaj sağlar. Bu avantajları aşağıdaki şekilde sıralamak mümkündür:

1- Makro kullanımı ile her seferinde tekrar tekrar çalıştırılan programlarda kullanıcı kaynaklı hata olmasının önüne geçilir. Makro kullanılmadığı durumlarda kullanıcılar yanlış hücre seçimleri, yanlış formül kullanımı gibi birtakım hatalar yapabilmektedir. Özellikle karmaşık işlem setlerinin tekrarlanmasında kullanıcının dikkatinden kaçan hatalar olabilmektedir. Kurallara uygun hazırlanmış bir makro hatasız çalışacaktır.

2- Makrolar zamandan kazanç sağlarlar. Kullanıcının işlem setlerini her seferinde aynı dikkatle yapabilmesi için uzun süreler harcaması gerekir. Makro kullanımında sadece makro ilk oluşturulurken süre harcanır. Harcanan süre işlem süresinden çok az daha fazladır. Sonrasında makro çalıştırılırken işlemlerin yapılma süresinden tasarruf edilmiş olur. Ayrıca uzun zamanlar alabilecek kopyala yapıştır gibi işlemler de otomatik olarak yapılacağından işlerin bitme süresi oldukça kısalacaktır.

3- Makrolar çalıştırıldığında öngörülür sonuçlar verirler, çünkü her makronun hangi işlemleri yapacağı önceden belirlenmiştir ve makro oluşturulurken, makro kaydedilmesi için en az bir kez çalıştırılmıştır.
4- Makrolar taşınabilirler, bu nedenle çalışma kitabından bağımsız olarak bir kere yazıldıktan sonra diğer çalışma kitaplarında da kullanılabilirler. Makroların gelişip yaygınlaşmasındaki en önemli etkenlerden biri bu özelliği olmuştur. Bir kez hazırlanan makro farklı çalışma kitaplarında kullanılabildiğinden benzer işleri yapan farklı kurumlarda bile aynı makrolar çalıştırılabilmektedir.
5- Makrolar ileri seviye programlama bilgisi gerektirmez. Genellikle makro yazabilmek için giriş seviyesinde programlama bilgisi yeterli olmaktadır. Excel programında yeterli tecrübesi olmayan kullanıcılar bile makro oluşturabilirler. Tabi gelişmiş makro yazımları için ileri seviye programlama bilgisi gerekmektedir.

Makrolar işlem tablosu programlarında pek çok avantajı beraberinde getirse de bazı dezavantajları da vardır. Makroların sunduğu işlem kolaylığı ve hızlı sonuç üretme, tekrarlanacak işler için zamandan tasarruf gibi avantajlar dikkatli kullanılmazsa ya da makrolar yanlış oluşturulursa tekrarlanacak hatalar da beraberinde gelecektir. Aşağıda makroların dezavantajları listelenmiştir:

Excel programında temel düzeyde makro yazmak çok zor olmamasına ve yazılım bilgisi gerektirmemesine rağmen karmaşık makrolar yazabilmek için ileri seviye programlama bilgi ve becerisi gerekir. Özellikle karmaşık makroların oluşturulabilmesi için kullanıcının VBA dilinde programlama tecrübesinin olması gerekmektedir.

1- Makroların çalıştırılması kayıtlı işlemlerin tekrar edilmesi demektir. Bu nedenle makrolar oluşturulurken yapılan bir hata makronun her çalıştırılmasında tekrar edecektir. Mantıksal bir tutarsızlık, hatalı seçilmiş bir hücre, göreli referanslarla çalışma gibi basit hatalar bile makrolarda çok büyük olumsuz sonuçlar olarak geri dönebilmektedir. Özellikle makrolar taşınabildiğinden taşındığı ve çalıştığı her çalışma kitabında aynı hata tekrarlanacaktır.
2- Makrolar değişen sürümlerde doğru çalışmayabilir. Özellikle VBA makro programlama diliyle oluşturulan makrolar güncellenen sürümlerde hiç çalışmayabilir ya da eksik veya hatalı çalışabilir. VBA dili Excel tarafından desteklenen bir dil olmakla beraber Excel programı güncel sürümlerinde VBA dilinin bazı sürümlerini desteklemeyi bırakabilmektedir. Bu nedenle makrolar taşınırken ya da kopyalanırken sürümlerin aynı olmasına veya kopyalandığı sürümde çalışıp çalışmadığına bakılması gerekmektedir.

Makrolar çalıştırılabilir programlardır. VBA diliyle hazırlanmış bu programlar zararlı yazılımlar içerebilir. Çünkü VBA dili bir yazılım dilidir ve bu dille virüs veya zararlı yazılım geliştirilebilir. Bu nedenle bilinmeyen kaynaklardan gelen makroların çalıştırılmasında dikkatli olunmalıdır. Virüs ve zararlı yazılım tarama programlarının kullanımı, Excelin kendi güvenlik duvarının kullanımı virüsler gibi sisteme zarar verebilecek yazılımların engellenmesinde yardımcı olabilir.

Makrolar Şerit üzerindeki Görünüm menüsünde bulunurlar. Bu menü aracılığıyla makro kaydetme, makro çalıştırma, makrolarda düzenleme yapma gibi işlemler yapılabilir. Excelde Şerit özelleştirilebilmektedir. Makrolar için de hızlı erişim butonları ve kısayol tuşları tanımlanabilir. Kısayollar kullanılarak makro çalıştırmak daha hızlı olacaktır.

Excel programı tüm ortamlarda birebir aynı fonksiyonları çalıştırmayabilir veya desteklemeyebilir. Özellikle MAC, Linux gibi farklı işletim sistemleri ile bulut sisteminde çalışan Excel programları Microsoft Windows işletim sistemleri için geliştirilen Excel programıyla aynı fonksiyonları kullanmayabileceğinden makrolar kopyalanırken ve oluşturulurken dikkatli olunması gerekir.

Bu bölümde makro oluşturma, kaydetme, güncelleme, çalıştırma, hızlı erişim tuşu ve kısayol tuşu oluşturma konularına yer verilecektir. Ayrıca makro güvenlik seçenekleri ve otomatik makro çalıştırma konularına yer verilmiştir.

GENEL BİLGİLER

İşlem tablosu programlarında makro kullanımı yaygındır. Makrolar sadece Excel programında değil diğer işlem tablosu programlarında kullanılır fakat her işlem tablosu programı aynı makroları çalıştıramaz. Bunun en önemli sebebi kullanılan makro yazılım dilidir. İlk makro oluşturma işlemi Lotus 1-2-3 programı ile başlamış ve günümüzde kullanılan makrolara kıyasla oldukça basit makrolar oluşturulabilmiştir. Lotus 1-2-3 te kullanılan makrolar tuş hareketlerinin kaydedilip sonradan çalıştırılması esasına dayanmaktaydı. Bu yapı günümüzde kullanılan makro yapısının temellerini oluşturmaktadır.

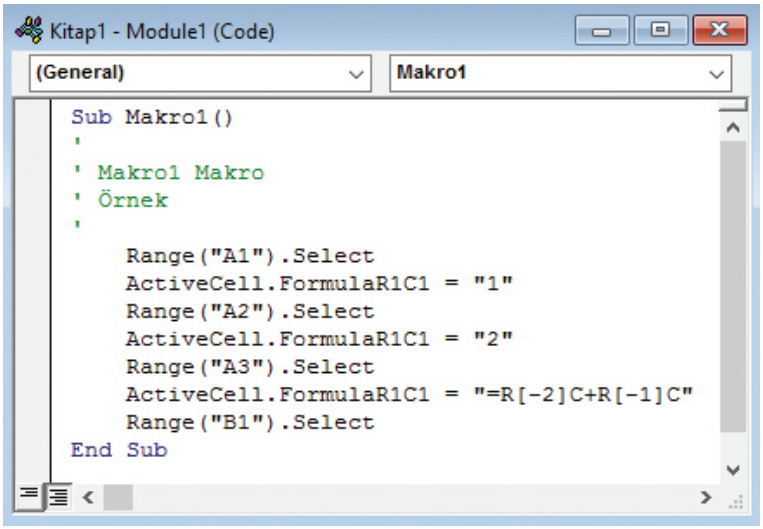
Microsoft firması Excel programında makro kullanımına XLM makrolarıyla başlamıştır. XLM makroları kullanımı zor olmasına rağmen oldukça güçlüdür. Microsoft kullanıcıların makro oluşturma konusunda zorlanması nedeniyle daha kolay makro oluşturabilecek VBA dilini Excel in sonraki sürümlerinde kullanmaya başlamıştır.

Microsoft Excelin yeni sürümlerinde XLM makrolar oluşturulamaz, bunun için VBA makro dili kullanılır fakat Excel programının yeni sürümleri XLM ile oluşturulmuş makroları çalıştırabilir.

Makro oluşturmak için kullanılan VBA dili Excelin piyasada üstün duruma gelmesini sağlayan önemli özelliklerden biridir. VBA makro dili ile makrolar oluşturulabilir, programlanabilir, kaydedilebilir ve çalıştırılabilirler.

Resim 2.1

Örnek VBA Kodu



Makro oluşturmak için VBA makro dili genellikle programcılar tarafından kullanılırlar. VBA ile makro oluşturmak çoğunlukla karmaşık makrolar için kullanılır ve kimi zaman ileri seviye programlama bilgisi gerektirir. VBA ile makro oluşturmak için VBA editörü kullanılır. VBA editörünün kullanımına ilişkin bilgiler ilerleyen bölümlerde verilecektir.

Makro oluşturmak için ikinci yöntem ise Excel programının sunduğu makro kaydedici kullanılarak makro oluşturulmasıdır. Makro kaydedici ile VBA kodu kullanılmasına gerek kalmadan makrolar oluşturulabilmektedir. VBA makro diline ihtiyaç duyulmadan

makro hazırlanabiliyor oluşu kullanıcılar tarafından makro kaydedicinin daha çok tercih edilmesinin önemli nedenlerindendir. Makro kaydedici başlatıldığı zaman Excel de yapılan işlemler kaydedilmeye başlanır. Makro kaydetme sonlandırıldığı zaman Excel kaydedilmiş işlemleri VBA diline çevirerek makroyu kaydeder. Kaydedilen makro sonradan değiştirilebilir.

Excelde her iki yöntem de kullanılarak hem makro kaydedici hem de VBA dili kullanılarak makrolar hazırlanabilir. Bu birleştirilmiş yöntem ile makrolar daha kısa sürede ve daha kolay bir şekilde hazırlanabilmektedir. Makro kaydedici ile kodlanması uzun sürecek işlemler kaydedildikten sonra oluşan VBA dilindeki koda VBA editörü ile müdahale edilerek gerekli eklemeler ve düzeltmeler yapılabilir. Çoğu kullanıcı için sadece makro kaydedici yeterli olsa da bazı durumlarda bu hibrit yöntem kullanılmaktadır. Makro kodu yazmak uzun ve zahmetli bir iştir. Özellikle karışık makroların yazılımı oldukça zaman alabilmektedir. Bunun yerine kullanılan bu ikili çözüm zamandan tasarruf edilmesini sağlar.

Her zaman makro kaydedici yeterli olmayabilir. Örneğin kullanıcı tanımlı diyalog kutularının oluşturulması gibi işlemler VBA diliyle yazılmak zorundadır. Fakat makro kaydedici ile kullanıcı tanımlı diyalog kutusu olmadan makro oluşturulup daha sonra VBA editörü ile makro kodu açılarak içine ekleme yapılabilir. Bu sayede makro kaydedici ile kolaylıkla yapılan işlemler için kodlama yapılmadan kodların otomatik olarak üretilmesi sağlanmış olur. Her iki yöntemle oluşturulan makrolar da Excelde VBA diliyle kaydedildiğinden istenildiği zaman makro koduna müdahale edilerek güncelleme ve ekleme yapılabilmektedir.

MAKRO İŞLEMLERİ

Makro kaydedici kullanıcının yaptığı her hareketi kaydetmez. Sadece işlem içeren hareketler kaydedilir. Örneğin bir hücrenin içeriğinin değiştirilmesi kaydedilirken menüler arası gezinme veya bekleme kaydedilmez.

Makrolarla alakalı yapılacak işlemler yeni bir makro oluşturma, oluşturulan makroyu kaydetme, kaydedilmiş makroyu yeniden düzenleme, makrolarla ilgili kısayol tuşları ve hızlı erişim çubuğu ayarlamaları yapma, çalışma kitabının açıldığı anda makroların çalışmasını sağlayacak ayarları yapma gibi işlemlerdir. Sayılan işlemlerin tamamı Excel programı içerisinden yapılabilmektedir. Bu bölümde makro kaydedici kullanılarak makrolarla çalışma konusu işlenecektir.

Makro kaydedici ile karmaşık makroların oluşturulması çok zordur ve kullanıcı diyalog kutusu oluşturma gibi işlemlerin yapılması ise mümkün değildir. Makro kaydedici karmaşık makrolarda küçük bölümler için kullanılabilir. Makro kaydedici VBA dili ile kodlama oluşturduğu için karmaşık sorguları bölümleyip küçük bölümleri makro kaydedici ile oluşturarak daha hızlı kodlama yapılabilir.

Makro kaydedici ile oluşturulan VBA kodu her zaman temiz kod olmayabilir, bir başka deyişle içerisinde gereksiz kod parçacıkları olabilir. Bu gereksiz kod parçacıkları VBA editörü ile makro kodu açılarak elle temizlenebilir.

Makro Kaydetme

Excelde *Görünüm* menüsü altında *Makrolar* alt menüsü bulunmaktadır. Bu menüden *Makro Kaydet* e tıklanarak yeni bir makro kaydı başlatılabilir.

Makro kaydedici fonksiyon tanımlamaları içeren Function oluşturamayıp Sub prosedürleri oluşturur. Sub prosedürleri dışarıdan parametre almazlar, tanımlı komut setlerini çağrıldıkları yere yansıtırlar. Function'lar ise parametre alıp bu değeri işleyerek sonuç döndürebilirler.

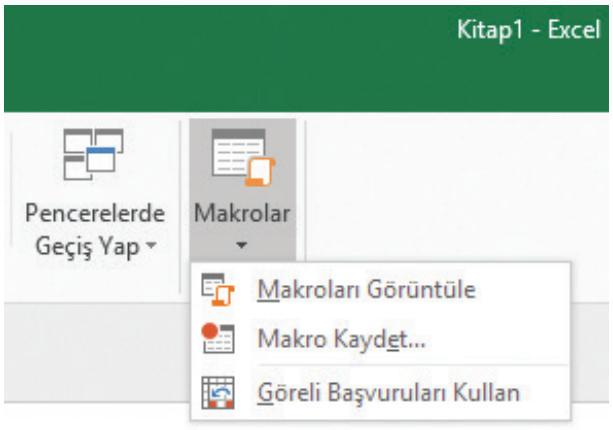
Makro kaydetme ara yüzünde oluşturulacak makroya ilişkin bilgiler istenir. Bu bilgiler makro adı, makroya erişim için kısayol tuşu, makronun kaydedileceği yer ve makro hakkında kısa bir açıklama bilgisidir.

Makro isimlendirmesi yapılırken boşluk karakteri kullanılamaz. Birden fazla kelime kullanılacaksa bu kelimeler bitişik yazılmalı veya aralarına alt çizgi "_" gibi karakterler konulmalıdır. Makrolarda benzer isimler ya da yanlış isimlendirmeler hatalı kullanımlara sebep olabilir. Bu nedenle makro isimlendirmesi yapılırken verilen ismin makroyu tanımlar şekilde olmasına dikkat edilmelidir.

Makrolar oluşturulurken kısayol tuşu tanımlaması istenmektedir ama kısayol tuşu zorunlu alan değildir. Eğer kısayol tuşu tanımlanırsa Excelde çalışırken ctrl tuşu ile birlikte tanımlanan kısayol tuşuna basıldığında kaydedilen makro çalıştırılır.

Resim 2.2

Makrolar Menüsü



Kısayol tuşu atanırken daha önceden atanmış kısayollar kullanılamaz. Bununla birlikte işletim sisteminin kullandığı kısayol tuşları da makro kısayolu olarak atanamaz. Örneğin Windows kopyala komutu için Ctrl+c, yapıştır için Ctrl+v, kes için Ctrl+x, tümünü seç için Ctrl+a tuşlarını kullanır. Bu tuş kombinasyonları Windows işletim sisteminin kendi kısayol komutları ile çakışacağından makro kısayol tuşu olarak kullanılamaz. Excel önceden tanımlı olan kısayol tuşlarından biri kullanılmaya çalışıldığında kullanıcıya uyarı verir. Kullanıcı isterse Windows işletim sisteminin kısayol tuşları dahil önceden kullanılan kısayol tuşlarını devre dışı bırakarak istediği tuş kombinasyonunu kısayol tuşu olarak makroya atayabilir.

DİKKAT

Windowsun kullandığı kısayol tuşları genellikle küçük harflerden oluşur. Bu tuşlar kısayol tuşu olarak atanmak istenirse shift tuşu ile birlikte kullanılarak atama yapılabilir. Örneğin Ctrl+c tuşu kopyala işlemi için ayrılan kısayol tuşudur. Makro için c tuşu Ctrl+Shift+c şeklinde kısayol tuşu olarak atanabilir.

Makro kaydedilirken sorulan sorulardan biride makronun nereye kaydedileceğidir. Excel makro saklama yeri olarak kullanıcıya üç seçenek sunmaktadır. Bunlar:

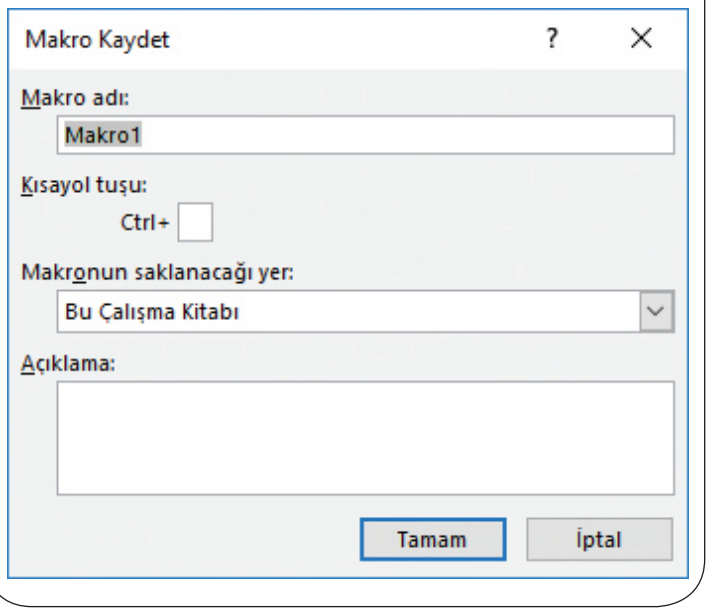
Bu Çalışma Kitabı: Oluşturulacak makro sadece ilgili çalışma kitabında kullanılacaksa bu seçenek seçilebilir. Çalışma kitabına özel olarak tanımlanmış makrolar o çalışma kitabı açıkken diğer çalışma kitapları tarafından kullanılabilirler.

Yeni Çalışma Kitabı: Bu seçenek seçildiğinde yeni bir çalışma kitabı otomatik olarak açılır ve makro yeni açılan çalışma kitabında kaydedilir.

Kişisel Makro Çalışma Kitabı: Excel tarafından özel olarak makro kaydedilebilmesi için oluşturulan çalışma kitabıdır. Bu çalışma kitabı ilk oluştuğu andan itibaren varsayılan olarak kullanıcıdan gizlenir. Kullanıcı isterse Görünüm menüsündeki Pencere alt menüsünde bulunan Göster sekmesine tıklayarak kişisel makro çalışma kitabını görüntüleyebilir. Bu çalışma

Resim 2.3

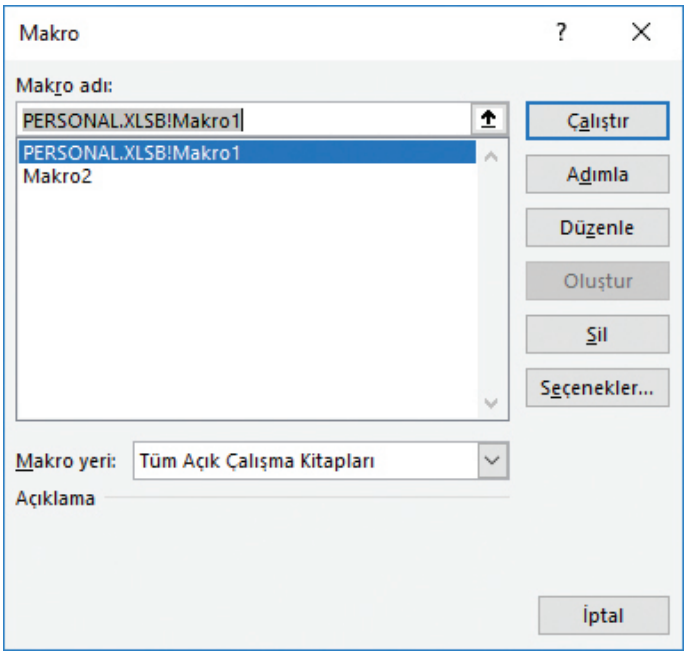
Makro Kaydetme Ekran Görünümü



kitabı Excel programının kurulumu yapılırken varsayılan olarak *C:\Users\KullaniciAdi\AppData\Roaming\Microsoft\Excel\Xlstart* dizinine kurulur. Kullanıcı isterse kurulum sırasında kurulum dizinini değiştirme hakkına sahiptir. *KullaniciAdi* alanı bilgisayar kullanıcısını ifade eder ve Excel her bilgisayar kullanıcısı için yeni bir kişisel makro çalışma kitabı oluşturur.

Resim 2.4

Çalışma kitaplarının görüntülenmesi



Kişisel makro çalışma kitabı *Personal.xlsb* ismiyle kaydedilir. Bu çalışma kitabına hiç makro kaydedilmişse bu dosya oluşmaz.

DİKKAT

Farklı bir çalışma kitabındaki makro kullanılmak istenirse makronun kayıtlı olduğu çalışma kitabının da açık olması gerekmektedir. Bu nedenle ortak kullanılacak makroların kişisel çalışma kitabında tanımlanması daha kolay bir kullanım sağlayacaktır.

Excel, makroların çalışabilmesi için ön koşul olarak makronun tanımlı olduğu çalışma kitabının açık olmasını ister. Kişisel makro çalışma kitapları bu amaçla oluşturulmuştur. Excel açıldığında ilk olarak açılan çalışma kitabı kişisel makro çalışma kitabıdır. Bu çalışma kitabı excel ilk kurulduğu zaman değil bu çalışma kitabına ilk makro kaydedileceği zaman otomatik olarak oluşturulur. Bu çalışma kitabı sürekli açık olacağından tüm çalışma kitapları bu çalışma kitabındaki makroları kullanabilir.

Makrolar oluşturulurken makro hakkında açıklama yazılması için de bir alan tanımlanmıştır. Bu alanı doldurmak zorunlu değildir fakat makro adında yer almayan detay bilgilerin bu alana yazılmasında fayda vardır. Açıklama alanına yazılan bilgiler makro kodunun başlangıcında yorum satırları olarak yer alır.

Makro oluşturma penceresindeki istenen bilgiler girilip makro oluşturulduğu andan itibaren kayıt başlar. Kaydın başlaması kullanıcının yaptığı tüm işlem ve hareketlerin kaydedilmeye başlaması anlamına gelir. Kullanıcı kaydı durduruncaya kadar kayıt devam eder. Kaydı durdurmak için Görünüm menüsündeki Makrolar sekmesinden *Kaydı Durdur* denmesi yeterlidir.

DİKKAT

Makro kaydı oluşturulurken yapılan hatalarla bu hatalara ilişkin düzeltme hareketleri de kaydedilir. Bu nedenle makro kaydı oluşturmadan önce yapılacak işlemlerin planlanması önemlidir. Yine de hatalar oluşur ve makro kaydı içerisinde hatalar giderilirse makro kaydedildikten sonra VBA kodu VBA editörü aracılığıyla açılıp gereksiz kod parçacıkları silinebilir.

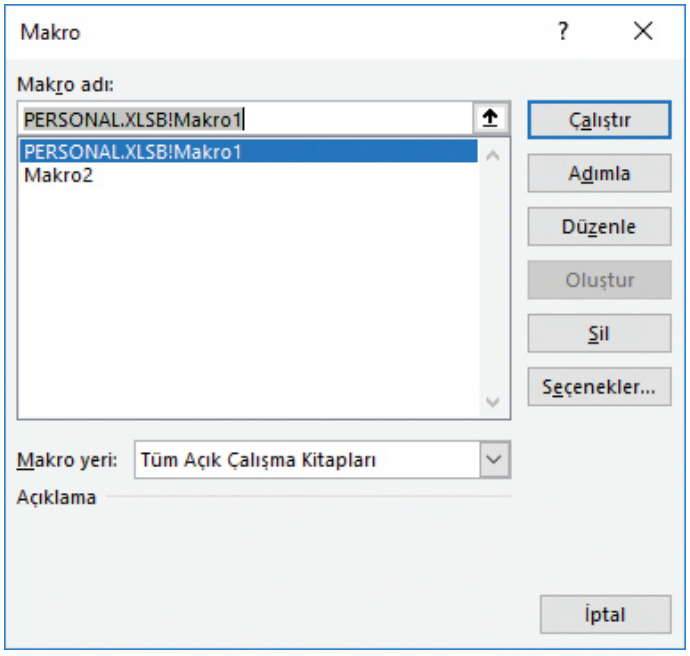
Makroların Çalıştırılması

Makrolar çalışmak için kayıtlı oldukları çalışma kitabının açık olmasına ihtiyaç duyarlar. Farklı bir çalışma kitabındaki makro kullanılmak istenirse öncelikle o çalışma kitabının açılması gerekir. Çalıştırılabilecek makrolar Görünüm menüsündeki Makrolar sekmesi altında *Makroları Görüntüle* tuşuna basılarak listelenebilirler.

Excel 2010 sürümü ve öncesinde makrolar kaydedilirken kullanıcı adı ve makronun oluşturulma tarihi otomatik olarak kaydedilmekteydi. Excel 2010 sürümünden sonra bu bilgileri otomatik olarak kaydedilmemektedir.

Resim 2.5

Makroların Çalıştırılması



Açılan diyalog kutusundan çalıştırılacak makro seçildikten sonra *Çalıştır* tuşuna basıldığında makro çalışmaya başlar. Makro çalışırken içerisine kaydedilmiş komutları sırası ile tekrarlar.

Makro kodunda Şeritteki gezinmeler, menüler arası geçişler, bekleme süreleri ve işlem süreleri kaydedilmez.

Makro çalıştırmak için tanımlı bir kısayol tuşu varsa bu kısayol tuşu kullanılarak da makro çalıştırılabilir. Fakat bu kısayol tuşu sadece makronun çalışabileceği çalışma kitabı açıksa aktif olur, çalışma kitabı dışında bu kısayol tuşu makroyu çalıştıramaz.

SIRA SİZDE 1

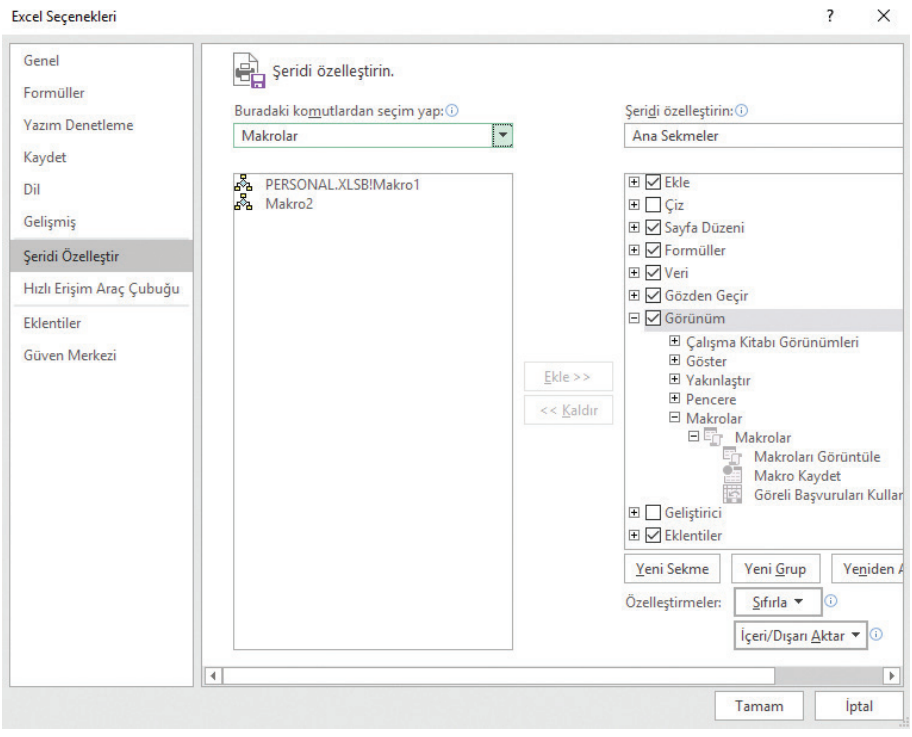
Excel programında bir makro oluşturun. Makronuz B1 hücresine Gelirler, C1 hücresine Giderler yazsın. Sonra B1 hücresinin arka planını yeşile C1 hücresinin arka planını kırmızıya boyasın. Sonra B2 hücresinden B5 hücresine kadar 4,5,6,7 sayılarını girsin. C2 hücresinden C5 hücresine kadar 1,2,3,4 sayılarını girsin. A6 hücresine Toplam: yazsın. B6 hücresine B2:B5 hücrelerinin toplamını, C6 hücresine C2:C5 hücrelerinin toplamını girsin. A8 hücresine gelir gider farkı yazarak B8 hücresine de B6-C6 değerini yazsın. Son olarak A8 ve B8 hücrelerinin arka planını griye boyayarak kaydı bitirsin. Oluşturduktan sonra makronuzu çalıştırın.

Makrolar İçin Hızlı Erişim Butonları Tanımlama

Makrolara Görünüm menüsü altındaki Makrolar sekmesinden ulaşılmaktadır. Makrolara daha hızlı ve kolay erişim sağlamak için Şerit özelleştirilerek makrolara Şeritte hızlı erişim butonları tanımlanabilir. Şerit, Excel Seçenekleri menüsündeki *Şeridi Özelleştir* alanından özelleştirilebilmektedir.

Resim 2.6

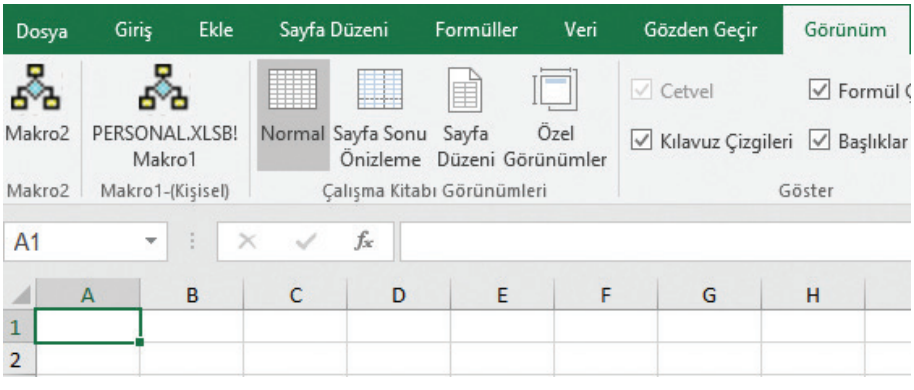
Şeridi Özelleştir alanı



Excel Seçenekler penceresinde özelleştirilecek alanın seçiminin yapılması gerekir. Bu pencerede *buradaki komutlardan seçim yap* açılır menüsünden makrolar başlığı seçilmesi gerekmektedir. Seçim yapıldığında kullanılabilecek makrolar görüntülenecektir. Bu makroların şeritte yer alabilmesi için bir gruba ekli olması gerekir. Gruba eklemek için ilk olarak *Şeridi Özelleştir* alanından yeni grup oluşturulur, grup oluşturulduktan sonra makrolar bu gruba eklenebilir.

Resim 2.7

Makrolara ilişkin kısayol butonlarının eklenmiş hali



SIRA SİZDE 2

Sıra Sizde 1 de oluşturduğunuz makro için Şeritte hızlı erişim butonu tanımlaması yapınız.

Makroların İncelenmesi ve Hata Ayıklanması

Makro oluşturmanın birden fazla yolu olmasına rağmen tüm makrolar VBA dili ile kodlanırlar. Makrolar VBA diliyle yazılmış olan kodlarına bakarak incelenebilirler. Kaydedilen tüm makroların VBA kodlarını görüntüleyip bu kodlar arasında hata ayıklaması yapmak, gereksiz kod parçacıklarını silmek ve gerekiyorsa koda ilaveler yapmak mümkündür.

Excel programında makroların VBA kodunu incelemek için Görünüm menüsünden Makrolar sekmesi ve ardından Makroları Görüntüle butonuna basılarak diyalog kutusu açılır. Bu diyalog kutusunda kayıtlı tüm makrolar görüntülenir.

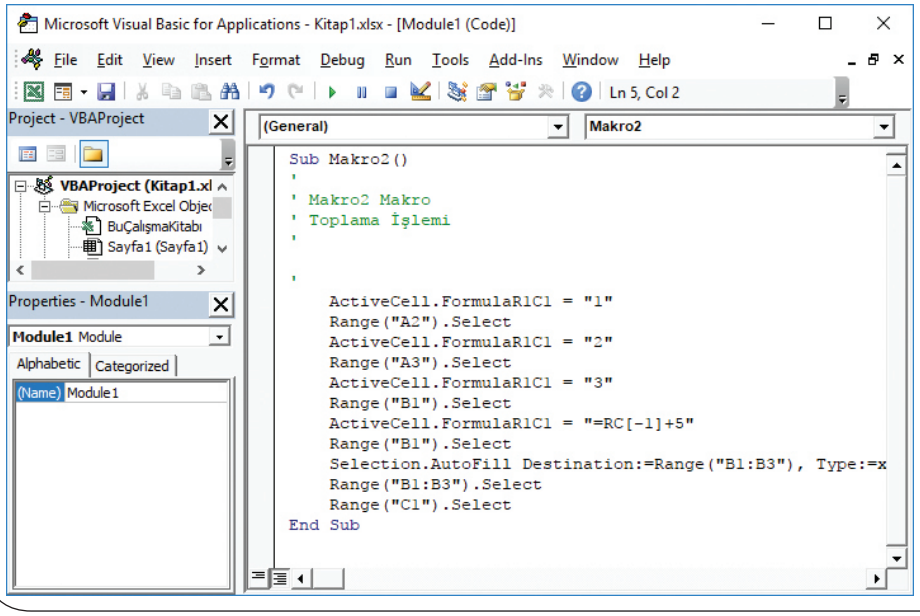
DİKKAT

Makroları görüntüle diyalog kutusunda sadece açık çalışma kitabının erişebildiği makrolar görüntülenir. Farklı bir çalışma kitabındaki makroların kodları incelenmek istenirse o çalışma kitabının da açılması gerekir.

Diyalog kutusundan kod incelenmesi yapılmak istenen makro seçilip Düzenle butonuna basılarak seçilen makronun VBA makro dili ile yazılmış koduna erişmek mümkündür. Düzenle butonuna basıldığında açılan pencere VBA editörüdür. Bu editör kullanılarak kod içerisinde hata ayıklama, düzenleme ve silme işlemleri yapılabilir.

Excel programında Alt+F11 tuş kombinasyonu VBA editörünün kısayol tuş kombinasyonudur.

Resim 2.8

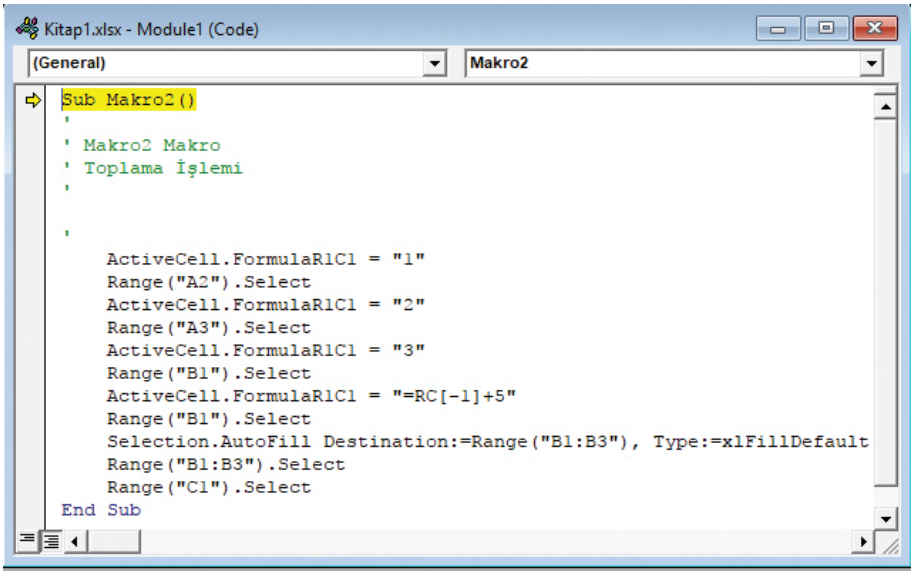


VBA editörü arayüz görünümü

VBA editörü ile hem kod düzenlemesi hem de hata ayıklaması yapılabilir. Makroları görüntüle diyalog kutusunda Düzenle tuşuna basılırsa VBA kodunu yeniden düzenlenebilir, Adımla tuşuna basıldığında ise editör hata ayıklama moduna geçer ve program hata ayıklama modunda adım adım çalıştırılır. Adım adım çalışma seçildiğinde makroda kaydedilen işlem adımları sırasıyla ve kullanıcıya bağlı olarak çalıştırılır.

Resim 2.9

VBA editörü adımla modunda çalışması



VBA makro dili bir yazılım dilidir. Bu nedenle hata ayıklama yapabilmek için programlama bilgisi gerekir. Makro kaydedici ile oluşturulan ve karmaşık olmayan makrolarda hata ayıklama için temel düzeyde programlama bilgisi yeterli olsa da, karmaşık makrolar için ileri seviye yazılım bilgisi gerekebilir. VBA makro diliyle alakalı daha detaylı bilgilere kitabın ilerleyen bölümlerinde yer verilecektir.

SIRA SİZDE 3

Sıra Sizde 1 de oluşturduğunuz makronun kodlarını VBA editöründe görüntüleyerek inceleyiniz.

Referans Modu

Excelde makrolar oluşturulurken varsayılan olarak mutlak referans kullanılır. Makro program kodlarının mutlak referanslı olarak kaydediliyor olması makrolar çalıştırılırken aktif hücreye bakılmayacağı anlamına gelir. Bu nedenle makro çalıştırılacağı zaman kullanıcı hangi hücreyi seçerse seçsin sonuç değişmeyecektir.

Aşağıda örnek bir senaryo kullanılarak makro kaydedici ile oluşturulan VBA makro kodu yer almaktadır. Bu senaryoda her şey Excelin varsayılan ayarları ile yapılacaktır.

Örnek senaryo:

- A1 hücresi seçilir.
- A1 hücresi seçili iken makro kaydedici ile yeni makro kaydedilmeye başlanır.
- A2 hücresi seçilerek içeriği 2 yapılır.
- A3 hücresi seçilerek içeriği 3 yapılır.
- A4 hücresi seçilerek içeriği =TOPLA(A2:A3) yapılır.
- B1 hücresi seçilir.
- Makro kaydı durdurulur.

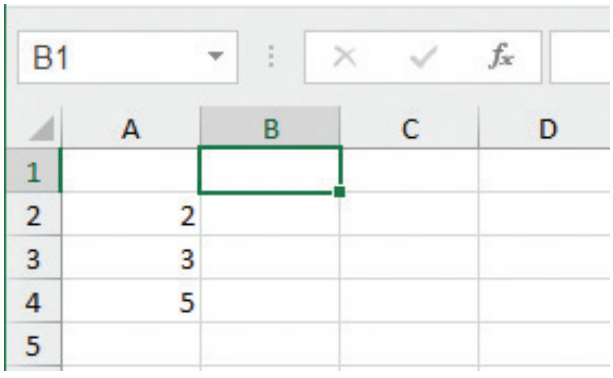
Bu örnek senaryodaki makroya ait VBA kodu açıldığında aşağıdaki şekilde bir makro kodu görüntülenir:

```
Sub Macro1()
'
' Macro1 Macro
' Örnek Senaryo
'
 Application.Goto Reference:="Macro1"
 Range("A2").Select
 ActiveCell.FormulaR1C1 = "2"
 Range("A3").Select
 ActiveCell.FormulaR1C1 = "3"
 Range("A4").Select
 ActiveCell.FormulaR1C1 = "=TOPLA(R[-2]C:R[-1]C)"
 Range("B1").Select
End Sub
```

Yukarıdaki senaryoya göre hazırlanan bir makro çalışma sayfasının herhangi bir hücresi seçili iken çalıştırılabilir. Fakat her çalışmasında mutlak referans kullanıldığı için aynı hücreler etkilenecektir. Örneğin bu makro, çalışma sayfasında hangi hücre seçili olursa olsun, hep aynı sonucu verecektir. Bu makronun çalıştırılması sonrasında çalışma sayfası aşağıdaki gibi olacaktır:

Resim 2.10

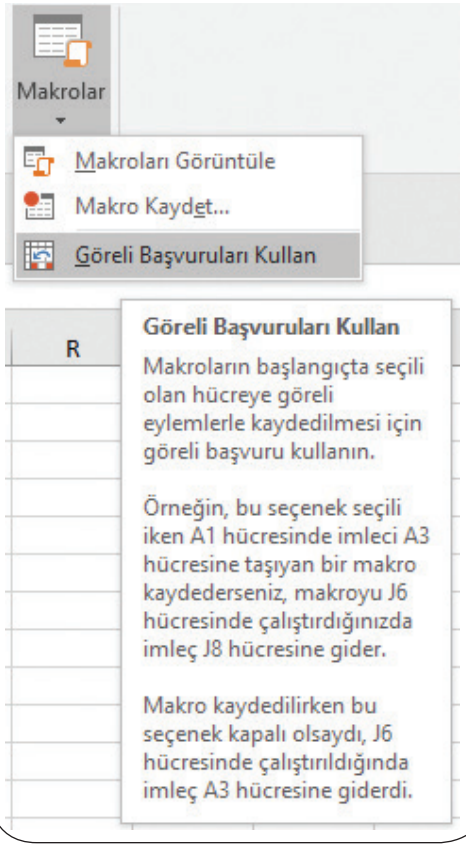
Örnek Makronun çalışma sonucu



Excelde makrolar oluşturulurken mutlak referans verme varsayılan olarak tanımlanmıştır. Kullanıcı bu ayarı değiştirerek makroların göreli referans ile oluşmasını da sağlayabilmektedir. Bunun için Görünüm Menüsündeki Makrolar sekmesinden *Göreli Başvuruları Kullan* seçeneği seçilmelidir.

Resim 2.11

Göreli Başvuruları Kullan Seçeneği



Göreli referans kullanarak oluşturulacak bir makro örneği aşağıdaki senaryoda gösterilmiştir.

Örnek Senaryo:

- A1 hücresi seçilir.
- Görünüm menüsündeki Makrolar sekmesinden *Göreli Başvuruları Kullan* seçeneği seçilir.
- A1 hücresi seçili iken makro kaydedici ile yeni makro kaydedilmeye başlanır.
- A2 hücresi seçilerek içeriği 2 yapılır.
- A3 hücresi seçilerek içeriği 3 yapılır.
- A4 hücresi seçilerek içeriği =TOPLA(A2:A3) yapılır.
- B1 hücresi seçilir.
- Makro kaydı durdurulur.

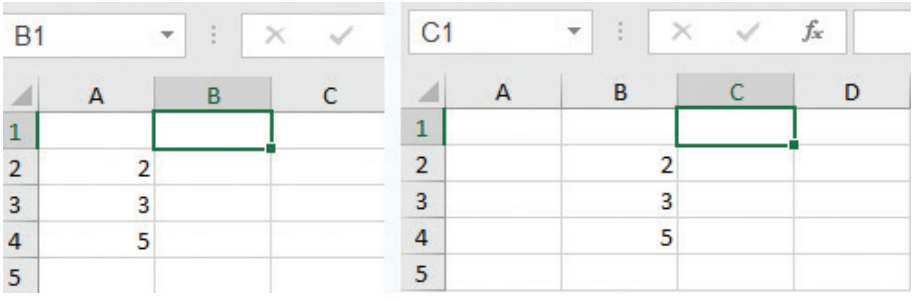
Bu örnek senaryodaki makroya ait VBA kodu açıldığında aşağıdaki şekilde bir makro kodu görüntülenir:

```
Sub Makro2()
'
' Makro2 Makro
' Örnek Senaryo
'
    ActiveCell.Offset(1,  0).Range("A1").
Select
   ActiveCell.FormulaR1C1 = "2"
    ActiveCell.Offset(1,  0).Range("A1").
Select
   ActiveCell.FormulaR1C1 = "3"
    ActiveCell.Offset(1,  0).Range("A1").
Select
 ActiveCell.FormulaR1C1 = "=SUM(R[-2]C:R[-1]C)"
 ActiveCell.Offset(-3, 1).Range("A1").Select
End Sub
```

Yukarıdaki senaryoya göre hazırlanan bir makro, çalışma sayfasının hangi hücresi seçili ise o hücreye göreli referans kullanarak makroyu çalıştırır. Örneğin bu makro A1 hücresi seçili iken çalıştırıldığında ve B1 hücresi seçili iken çalıştırıldığında aşağıdaki gibi olacaktır.

Resim 2.12

Sırasıyla A1 ve B1 hücresi seçili iken makronun çalışma sonucu



Sıra Sizde 1 de oluşturduğunuz makroyu göreli referans kullanarak yeniden oluşturunuz. Oluşan VBA kodunu inceleyiniz. Sonra C1 hücresini seçerek makroyu çalıştırınız ve sonucu gözlemleyiniz.

SIRA SİZDE 4

Çalışma Kitabı Açıldığında Otomatik Olarak Çalışacak Makroların Oluşturulması

Excelde yazılan makrolar istenildiği zaman kullanıcı tarafından çalıştırılabilmektedir. Aynı zamanda yazılan makroların çalışma kitabının ilk açılışında çalıştırılması da sağlanabilmektedir. Yeni bir çalışma kitabı açıldığında otomatik olarak çalışıp açılan çalışma kitabında istenilen alanları dolduran bir makro, test senaryoları hazırlama, sunum hazırlama gibi zamanlarda kullanışlı olmaktadır. Örneğin test verileri ile dolu bir çalışma kitabı üzerinden öğrencilere ders anlatımı yapılacaksa bu yöntem kullanılabilir. Bir başka deyişle test verilerini otomatik hazırlayacak bir makro yazılabilir. Bu makro, çalışma kitabının ilk açılışında çalışacak şekilde düzenlenir ve çalışma kitabı açıldığında önceden hazırlanmış makro çalışarak test verilerini otomatik olarak açılan sayfaya yazar.

Excelde bir çalışma kitabı açıldığında otomatik olarak çalışacak makro yapmak için sadece makro adının değiştirilmesi yeterlidir. Excel açılışında ilk olarak otomatik çalıştırılacak makro var mı diye bakar, varsa çalıştırır. Bu aramayı da sadece makro adına bakarak yapar. Excelin aradığı makro ismi "Auto_Open" ismidir. Eğer Excelde makro oluşturulurken *Auto_Open* ismi makroya verilirse o makro çalışma kitabı ilk açıldığında çalışacaktır.

Makroların Gizlenmesi

Excelde yazılan makroların sayısı arttıkça bazı makroların gizlenmesi istenebilir. Özellikle tek başına kullanılmayan, sadece diğer makrolar tarafından kullanılan makroların diyalog kutusunda görünmesi veya çağrılması istenmeyebilir.

Makroların görüntülendiği diyalog kutusundan herhangi bir makro gizlenmek istenirse VBA editörü kullanılarak makronun VBA kodu açılır. Makronun VBA kodunun başında *Sub MacroAdı()* ifadesi yer alır. Bu satırda *Sub* kelimesinin önüne *Private* yazıldığında, *Private Sub MacroAdı()* şeklinde satır güncellendiğinde makro gizlenmiş olur.

Makro gizleme, makro sayısının fazla olduğu zamanlarda ya da diyalog kutusundan kullanılması istenmeyen makroların gizlenmesi gerektiğinde yapılmaktadır. Bazı durumlarda ise bir çeşit güvenlik önlemi olarak makroların gizlenmesi istenebilir. Örneğin çalışma kitabı bir başkasına verilecekse veya çalışma kitabına birden fazla kişi erişiyorsa makroların bilmeden çalıştırılmasını önlemek gibi isteklerle de makrolar gizlenebilir.

Makroların gizlenmesi o makrolara ulaşılamayacağı anlamına gelmez. Sadece diyalog kutusu üzerinden makrolara erişim engellenmiş olur fakat VBA editörü ile gizlenmiş makrolar görüntülenebilirler.

Makrolar gizlenirlerse o makrolar için tanımlanmış kısayol tuşları da çalışmaz hale gelirler.

DİKKAT

MAKROLARDA GÜVENLİK

Makrolar kullanıcıların işlerini kolaylaştırmak amacıyla yazılmış çalıştırılabilir komut setleridir. Bu komut setleri çoğu zaman zararsız komutlardan oluşsa da bazen kötü niyetli yazılımcılar tarafından virüs ya da zararlı kod parçacığı haline getirilebilirler. Zararlı kod parçacıkları sisteme ciddi zararlar verebilirken bazen de bilgi çalmak amacıyla oluşturulurlar.

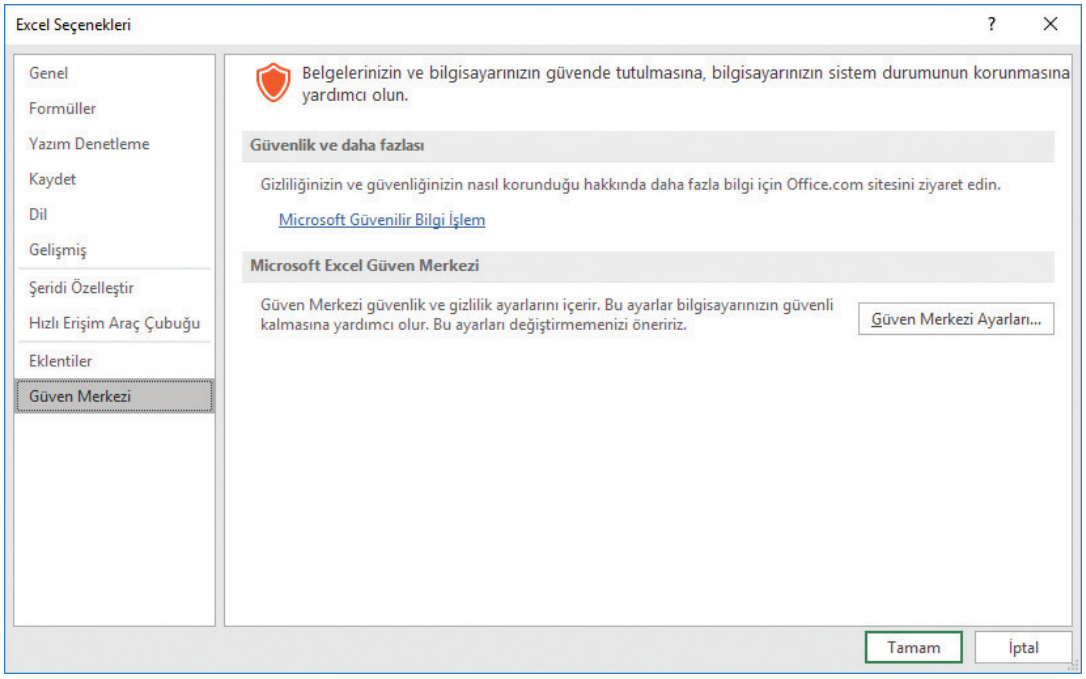
Gizlenmiş makrolar tekrardan görünür hale getirilmek istenirse; makro kodu VBA editörü ile açılarak eklenmiş olan Private ön ekinin silinmesi gerekir.

Kötü niyetli olarak hazırlanmış makroların çalıştırılmasının engellenmesi için Excel bazı güvenlik önlemlerini kullanıcılara sunar. Excelin makrolarla alakalı sunduğu güvenlik önlemlerinin yanı sıra güncel antivirüs programları ve zararlı yazılım tarama araçları da güvenliği artırmak için kullanılabilir.

Excel zararlı makroların çalıştırılarak sistem ve dosya güvenliğinin bozulmasını önlemek için kullanıcıya bazı alternatifler sunar. Bu alternatifleri ya da başka bir deyişle güvenlik ayarlarını kullanıcının kendisi istediği zaman değiştirebilir. Excelde makro güvenliği ile ilgili ayarlara Dosya menüsündeki Excel Seçenekleri sekmesinden erişilebilmektedir.

Resim 2.13

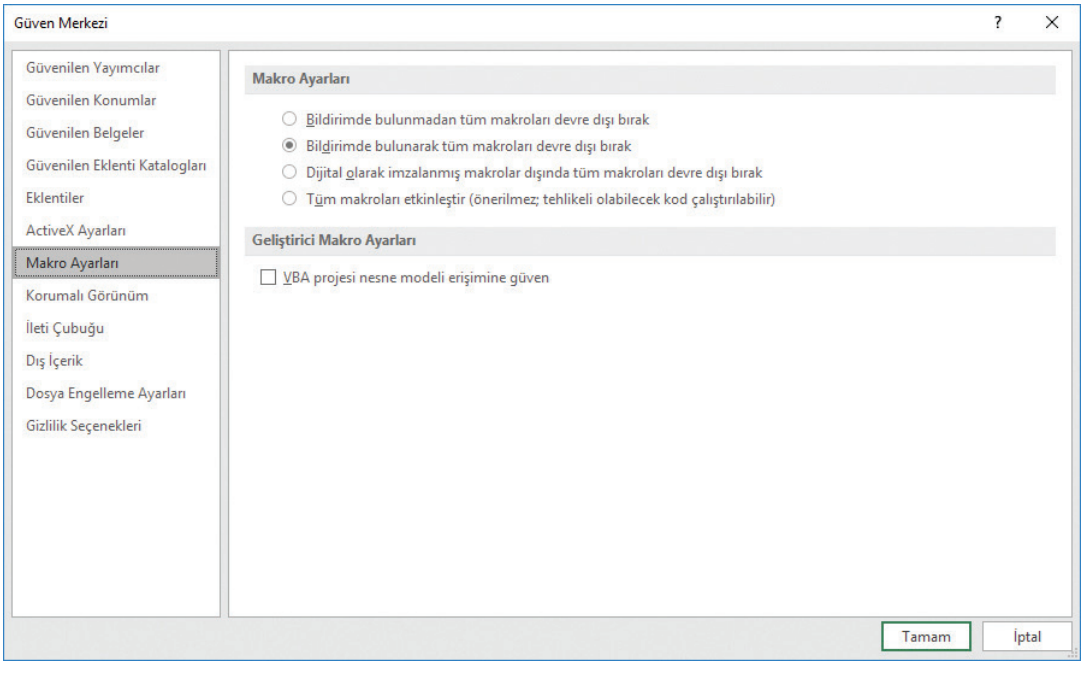
A Excel seçenekleri menüsü- güven merkezi alanı



Makro ayarlarına erişmek için seçenekler menüsündeki Güven Merkezi arayüzü açılır ve sonrasında makro ayarları sekmesi seçilir. Açılan pencerede aktif makro güvenlik ayarı görüntülenir. Excel varsayılan olarak *"Bildirimde bulunarak tüm makroları devre dışı bırak"* makro ayarı ile çalışır .

Resim 2.14

Güven merkezi- makro ayarları alanı



Makro ayarları dört farklı seçenek olarak sunulmaktadır. Bu seçenekleri kullanıcı istediği zaman değiştirebilir. Bu seçenekler sırasıyla:

1- *Bildirimde bulunmadan tüm makroları devre dışı bırak:* Hiç bir makronun çalıştırılamayacağı seçenektir. Bu seçeneğin seçili olduğu durumlarda kullanıcının kendi oluşturduğu makrolar da çalıştırılamaz.

2- *Bildirimde bulunarak tüm makroları devre dışı bırak:* Excelin varsayılan olarak kullandığı seçenektir. Bu seçenekte Excel açılırken kullanıcıya güvenilir kaynaktan gelen fakat dijital imzası olmayan makroların çalıştırılıp çalıştırılmayacağını sorar. Bu seçenekle kullanıcı kendi oluşturduğu makroları sorunsuz bir şekilde çalıştırabilir ve kendi oluşturduğu makrolar için dijital imza kullanmak zorunda kalmaz.

3- *Dijital olarak imzalanmış makroların dışında tüm makroları devre dışı bırak:* Bu seçenekle dijital imzası olmayan hiç bir makro çalıştırılamaz. Kullanıcı kendi oluşturduğu makroları da dijital olarak imzalamak zorundadır.

Kullanıcılar isterlerse Microsoft Office programları ile dağıtılan SelfCert.exe programı ile kendi oluşturdukları makroları imzalayabilirler. Bu durumda imzalayan kullanıcı ile makroyu çalıştıran kullanıcı aynı ise Excel dijital imzayı geçerli sayar. Fakat bu yöntemle imzalanan makrolar farklı kullanıcılar tarafından çalıştırılamazlar, çünkü bu yöntemle oluşturulan dijital imza sadece imzayı oluşturan kullanıcı için geçerli iken diğer kullanıcılar için geçerli dijital imza özelliği taşımaz.

Dijital imzalar güvenli elektronik kimlik denetleme araçlarıdır. Dijital imzalar, dijital sertifikasyon sağlayan firmalardan alınır.

Tüm kullanıcılar için geçerli dijital imza oluşturup kullanabilmek için, makronun dijital imza sertifikasyonu sağlayan firmaların sunduğu ücretli dijital imza ile imzalanması gerekir.

4- *Tüm makroları etkinleştir:* Bu seçenek en güvensiz seçenektir. Excel bu seçenek seçildiğinde tüm makroları çalıştırılabilmesine izin verir. Kaynağa ya da dijital imzaya bakılmaz. Bu seçenek seçildiğinde Excel makro güvenliği ile ilgilenmez. Kullanma zorunluluğu yoksa bu seçeneğin kullanılmaması önerilir.

Dijital olarak imzalanmış bir makro farklı bir kullanıcı tarafından açılıp çalıştırılabilir ve VBA editörü aracılığıyla içeriği değiştirilebilir. İçeriği değiştirilmiş makro artık dijital imzalı halinden farklı olduğundan imzalı kabul edilmez. Bu durumda sadece dijital imzalı makroların çalıştırılabileceği seçenekte içeriği değiştirilmiş makrolar çalıştırılmaz.

DİKKAT

Tüm makroları etkinleştir seçeneği aktif hale getirildiğinde Excel ve sistemler kötü niyetli yazılmış makrolara karşı savunmasız kalabileceğinden güncel bir antivirüs ve zararlı yazılım tarama programı kullanılması güvenlik açısından önemlidir. Antivirüs ve zararlı yazılım tarama programları varsa bile zorunluluk yoksa bu seçeneğin aktif edilmemesi gerekir.

Excel Dosya Çeşitleri

Excelde makro güvenliği, Excel 2010 sürümünden önce sadece makro güvenlik seçenekleri ile kontrol edilmekteydi. Microsoft firması Excel 2010 sürümünden sonra makro oluşturabilecek ve kaydedebilecek dosya uzantıları tanımlaması yapmıştır. Excelde dosya uzantısına bakılarak makro oluşturup oluşturulamayacağı söylenebilmektedir.

Kullanıcılar genelde .xlsx uzantılı excel dosyaları ile çalışırlar. Office programlar grubu içerisinde gelen Excel programı ile yeni bir çalışma kitabı açıldığında dosya uzantısı .xlsx olmaktadır. Bu uzantıya sahip dosyalar bir dizi XML nesnesi şekline getirilerek sıkıştırılır ve kaydedilirler. .xlsx uzantılı dosyalarda makrolar kaydedilemezler.

.xlsx dosyalarında makrolar kaydedilemediğinden bu dosyaları dış kaynaklardan alan kullanıcıların zararlı makro yazılımlarından korkmasına gerek yoktur. Microsoft, .xlsx uzantılı standart Excel dosyalarına makro kaydetme izni vermeyerek kullanıcıların makrolar konusunda güvenliğini artırmayı amaçlamaktadır.

Makrolarla çalışmaya imkan veren standart Excel dosya uzantısı .xlsm'dir. .xlsm uzantılı dosyalar .xlsx uzantılı dosyalara benzerler fakat ilaveten makroları oluşturup kaydedebilecek yapıdadırlar.

Standart Excel dosyalarının yanı sıra Excel şablon dosyaları da makrolarla çalışma konusunda ayrılmışlardır. .xltx uzantılı dosyalar makro kaydedemeyen Excel şablon dosyalarıdır. Şablon dosyalar aynı örneğin birden fazla çalışma kitabında kolaylıkla oluşturulabilmesini sağlayan dosyalardır. Şablon bir kez oluşturulduktan sonra yeni açılacak tüm çalışma kitaplarında kullanılabilmektedir.

Makro çalıştırabilen Excel şablon dosyaları .xltm uzantılı dosyalardır. Bu dosyalar .xltx uzantılı dosyalara ilaveten makro oluşturup kaydedebilme özelliğine sahiptirler.

Kişisel makro çalışma kitapları ikili değer (binary) formatında oluşturulan dosyalardır. .xlsb uzantılı olan bu binary dosyalarda makro çalıştırmak ve kaydetmek mümkündür. Excel de kişisel makro çalışma kitabı da binary formatında olup Personal.xlsb ismiyle kaydedilmektedir.

Özet

Makrolar hakkında temel bilgi sahibi olabilecek

Makrolar önceden kaydedilmiş komut setlerini istenildiği zaman çalıştırabilen araçlardır. Tekrar eden işlemlerin kullanıcı tarafından manuel yapılması yerine makrolar oluşturulup bu tekrar eden işler makrolar aracılığıyla otomatik olarak yapılabilir. Makrolar öngörülebilir sonuçlar verirler, zamandan tasarruf edilmesini sağlarlar, kullanıcı kaynaklı hataların önüne geçerler, taşınabilir ve kopyalanabilirler, karmaşık makrolar hariç ileri seviye programlama bilgisine gerek duyulmadan oluşturulabilirler. Bu faydalarının yanı sıra bazı dezavantajları da vardır. Bu dezavantajlar; makro kaydındaki her hangi bir hatanın sürekli tekrarlanacak olması, farklı ortamlarda çalışma garantisinin olmaması, Excelin değişen sürümlerinde oluşturulan makronun desteklenmeme ihtimali olarak sıralanabilir. Excel programı XLM diliyle başladığı makro yazılımını daha sonra VBA dili olarak güncellemiştir. Güncel Excel sürümlerinde XLM diliyle makro yazılamıyor olsa da bu dille oluşturulan makrolar çalıştırılabilir.

Temel makro işlemlerini açıklayabilecek

Temel makro işlemleri, makroların oluşturulması, kaydedilmesi, çalıştırılması, kod incelemesi yapılması, kod güncelleme yapılması, kısayol tuşu atanması, hızlı erişim butonu ayarlanması, görünmesi istenmeyen makroların gizlenmesi ve otomatik olarak başlangıçta açılmasının sağlanmasıdır. Makro oluşturmak en kolay yöntem makro kaydedicinin kullanılmasıdır. Makro kaydedici kullanıcının tuş hareketlerini kaydederek bu tuş hareketlerini VBA diline çevirerek makro oluşturmaya yarayan araçtır. İkinci yöntem ise VBA editörü ile makro kodu yazılmasıdır. VBA editörü aracılığıyla makro yazmak programlama bilgisi gerektirir. Makro kaydedici ile VBA editörü hibrit yöntemle kullanılabilir. Bu nedenle makro kaydedici ile kaydedilen makro VBA editörü ile yeniden düzenlenerek makrolar geliştirilebilir. VBA editörü üzerinden makro kodları düzenle ve adımla seçenekleri ile açılabilir. Düzenle modunda kodlara ekleme, silme, güncelleme yapılabilir. Adımla modunda ise oluşturulan makro adım adım çalıştırılabilir.

Makrolar için kısayol tuşları makronun ilk oluşturulduğu zaman atanabilmektedir. Ayrıca makro düzenle işlemi ile istenildiği zaman kısayol tuşu güncellenebilir. Şeride de makrolar için hızlı erişim butonları eklenebilmektedir.

Makrolar çalışabilmek için oluşturulduğu çalışma kitabının açık olmasına ihtiyaç duyarlar. Tüm çalışma kitaplarında çalışması istenen makrolar kişisel makro çalışma kitaplarına kaydedilir. Bu makro Personal.xlsb ismiyle kaydedilir. Bir çalışma kitabında oluşturulan makro farklı çalışma kitaplarında çalıştırılmak istenirse makronun kaydedildiği çalışma kitabının da açık olması gerekir.

Makrolarda güvenlik kavramını açıklayabileceksiniz

Makrolar kullanıcı dostudurlar ve tekrar eden işlemleri otomatik yapmak için kullanılırlar. Makrolar temelde komut setlerinden oluşur. Bu komut setleri kötü niyetli kullanıcılar tarafından zararlı yazılımlara dönüştürülebilirler. Makrolar virüs yada zararlı kod barındırabileceği için kullanılırken dikkatli olunması gerekir. Excel bu amaçla makro güvenliği seçenekleri sunmaktadır. Dört farklı seçeneği olan makro güvenliğinde Excel varsayılan olarak bildirimde bulunarak tüm makroları devre dışı bırak seçimini kullanır. Bu orta seviye bir güvenliktir. Bu seçenek dışında tüm makroların çalıştırılması, sadece dijital imzalı makroların çalıştırılması ve hiç bir makronun çalıştırılmaması seçenekleri vardır.

Excelin kendi makro güvenliğinin yanı sıra güncel antivirüs ve zararlı yazılım tarama araçlarının da kullanılmasında fayda vardır.

Excel dosya türleri ile makroların kaydedilip kaydedilemeyeceğini kontrol eder. .xlsx ve .xltx uzantılı dosyalarda makro kaydedilemez. .xlsm ve .xltm uzantılı dosyalar ile .xlsb uzantılı binary dosyalarda makro kaydedilip çalıştırılabilir.

Kendimizi Sınayalım

1. Makroların avantajları hakkında aşağıdakilerden hangisi söylenebilir?

a. Çalıştırıldıklarında öngörülebilir sonuçlar verirler.
b. Sadece oluşturuldukları çalışma kitabında çalışırlar.
c. Oluşturulduktan sonra değiştirilemezler.
d. Hatalı hücre seçimlerini otomatik olarak düzeltirler.
e. Virüs veya zararlı yazılım barındırmazlar.

2. Makrolar için aşağıdaki ifadelerden hangisi söylenemez?

a. Sadece makro kaydedici ile oluşturulurlar.
b. Makrolara kısayol tuşu atanabilir.
c. Makrolar ilk kez Lotus 1-2-3 işlem tablosu programında kullanılmıştır.
d. Makro kodu sonradan değiştirilebilir.
e. Makroların kayıtlı olduğu çalışma kitabı açık değilse makrolar çalışmaz.

3. Makrolar ilk kez aşağıdaki işlem tablosu programlarından hangisinde kullanılmıştır?

a. VisiCalc
b. SuperCalc
c. Excel
d. Lotus 1-2-3
e. MultiPlan

4. Makrolar oluşturulurken aşağıdakilerden hangisi kaydeder?

a. Şerit teki gezinmeler,
b. Menüler arası geçişler
c. Bekleme süreleri
d. İşlem süreleri
e. Hücre seçimleri

5. VBA editörü kullanılarak aşağıdaki işlemlerden hangisi **gerçekleştirilemez?**

a. Makro kodunda hata ayıklama
b. Makro kodu düzenleme
c. Yeni makro oluşturma
d. Oluşturulmuş makroyu silme
e. Makronun kayıtlı olduğu çalışma kitabını değiştirme

6. Excel programında VBA editörünün kısayol tuş kombisyonu hangisidir?

a. Alt+m
b. Alt+F10
c. Alt+F11
d. Shift +F10
e. Ctrl +F11

7. Excelde makrolar oluşturulurken varsayılan olarak hangi referans modunu kullanır?

a. Mutlak referans
b. Satır mutlak referans
c. Sütün mutlak referans
d. Göreli referans
e. A1 referansı

8. Excelde bir çalışma kitabı açıldığında otomatik olarak çalışacak makro yapmak için makroyu hangi isimde kaydetmek gerekir?

a. Auto_Open
b. Auto_Run
c. Auto_Start
d. Open_Macro
e. Start_Macro

9. Makroların görüntülendiği diyalog kutusundan herhangi bir makro gizlenmek istenirse VBA editöründe Sub kelimesinden önce hangi ön ek yazılır?

a. Private
b. Secret
c. Public
d. Volatile
e. Protected

10. Aşağıdakilerden hangisi makrolar için sunulan güvenlik seçeneklerinden biri **değildir?**

a. Bildirimde bulunmadan tüm makroları devre dışı bırak.
b. Bildirimde bulunarak tüm makroları devre dışı bırak
c. Dijital olarak imzalanmış makroların dışında tüm makroları devre dışı bırak
d. Tüm makroları etkinleştir
e. Seçili makrolar dışında tüm makroları devre dışı bırak

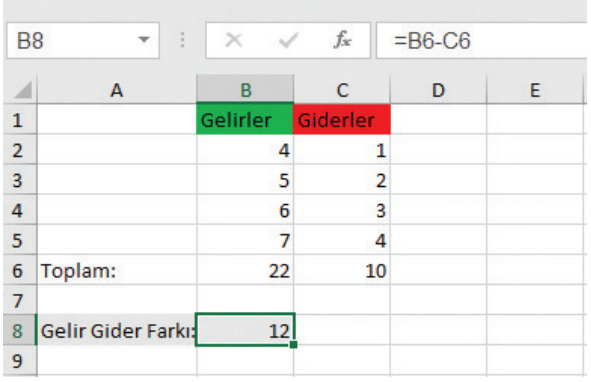
Kendimizi Sınayalım Yanıt Anahtarı

1. a	Yanıtınız Yanlış ise "Giriş" konusunu yeniden gözden geçiriniz.
2. a	Yanıtınız Yanlış ise "Genel Bilgiler" konusunu yeniden gözden geçiriniz.
3. d	Yanıtınız Yanlış ise "Genel Bilgiler" konusunu yeniden gözden geçiriniz.
4. e	Yanıtınız Yanlış ise "Makro İşlemleri" konusunu yeniden gözden geçiriniz.
5. e	Yanıtınız Yanlış ise "Makroların İncelenmesi ve Hata Ayıklanması" konusunu yeniden gözden geçiriniz.
6. c	Yanıtınız Yanlış ise "Makroların İncelenmesi ve Hata Ayıklanması" konusunu yeniden gözden geçiriniz.
7. a	Yanıtınız Yanlış ise "Referans Modu" konusunu yeniden gözden geçiriniz.
8. a	Yanıtınız Yanlış ise "Çalışma Kitabı Açıldığında Otomatik Olarak Çalışacak Makroların Oluşturulması" konusunu yeniden gözden geçiriniz.
9. a	Yanıtınız Yanlış ise "Makroların Gizlenmesi" konusunu yeniden gözden geçiriniz.
10. e	Yanıtınız Yanlış ise "Makrolarda Güvenlik" konusunu yeniden gözden geçiriniz.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1

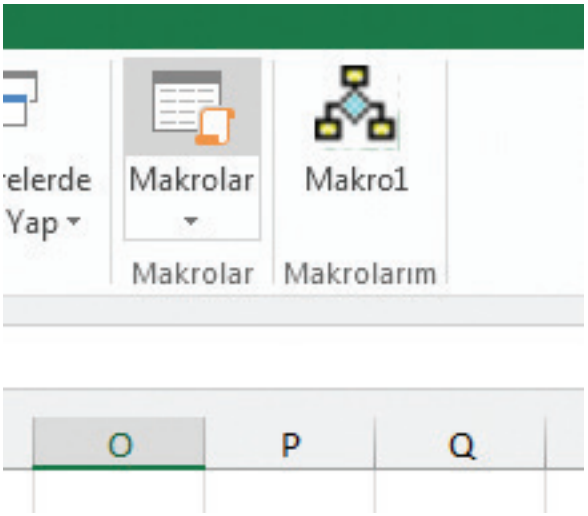
Makro oluşturmak için Görünüm menüsünde yer alan Makrolar alanından Makro Kaydet butonuna tıklanır. Makro ismi ve diğer bilgiler girilerek makro kaydı başlatılır. İstenilen işlemler yapıldıktan sonra makro kaydı durdurulur. Oluşturulan makronun çalışması sonucu oluşacak ekran görüntüsü aşağıdaki gibidir:



Resim 2.15: *Makro çalıştıktan sonraki Ekran görüntüsü*

Sıra Sizde 2

Excel seçenekleri menüsünden şeridi özelleştir penceresi açılır. Sonra yeni grup tanımlaması yapılarak bu grup altına oluşturulan makro eklenir. Makronun şeritte hızlı erişim butonu atanmış hali aşağıdaki gibidir.



Resim 2.16: *Şeritte Makro1 için oluşturulmuş hızlı erişim butonu*

Sıra Sizde 3

Sıra Sizde 1'de oluşturulan makronun VBA kodu aşağıdaki gibidir:

Kitap1 - Module1 (Code)

(General) Makro1

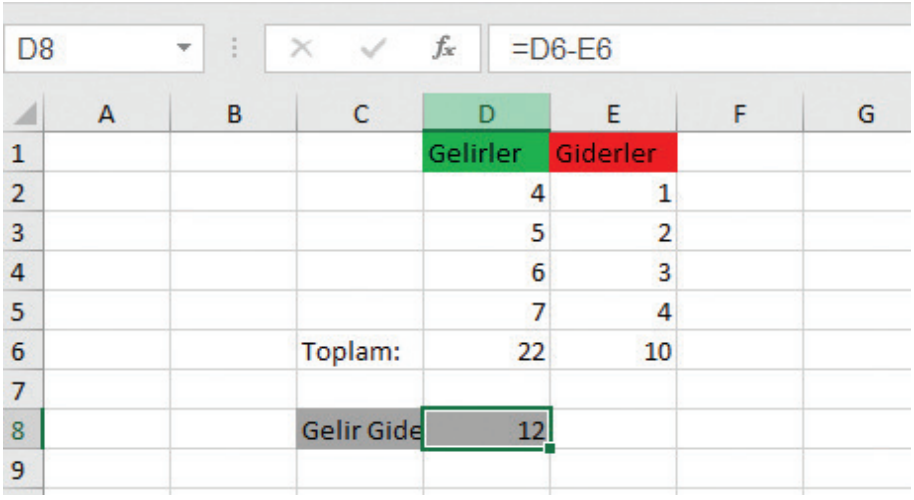
```
Sub Makro1()
'
' Makro1 Makro
' Gelirler-Giderler
'

'
    Range("B1").Select
    ActiveCell.FormulaR1C1 = "Gelirler"
    Range("C1").Select
    ActiveCell.FormulaR1C1 = "Giderler"
    Range("B1").Select
    With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 5287936
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
    Range("C1").Select
    With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 255
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
    Range("B2").Select
    ActiveCell.FormulaR1C1 = "4"
    Range("B3").Select
    ActiveCell.FormulaR1C1 = "5"
    Range("B4").Select
    ActiveCell.FormulaR1C1 = "6"
    Range("B5").Select
    ActiveCell.FormulaR1C1 = "7"
    Range("C2").Select
    ActiveCell.FormulaR1C1 = "1"
    Range("C3").Select
    ActiveCell.FormulaR1C1 = "2"
    Range("C4").Select
    ActiveCell.FormulaR1C1 = "3"
    Range("C5").Select
    ActiveCell.FormulaR1C1 = "4"
    Range("A6").Select
    ActiveCell.FormulaR1C1 = "Toplam:"
    Range("B6").Select
    ActiveCell.FormulaR1C1 = "=R[-4]C+R[-3]C+R[-2]C+R[-1]C"
    Range("C6").Select
    ActiveCell.FormulaR1C1 = "=R[-4]C+R[-3]C+R[-2]C+R[-1]C"
    Range("A8").Select
    ActiveCell.FormulaR1C1 = "Gelir Gider Farkı:"
    Range("B8").Select
    ActiveCell.FormulaR1C1 = "=R[-2]C-R[-2]C[1]"
    Range("A8:B8").Select
    Range("B8").Activate
    With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .ThemeColor = xlThemeColorDark2
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With
    Range("B8").Select
End Sub
```

Resim 2.17: *Sıra Sizde 1 için oluşturulmuş VBA kodu*

Sıra Sizde 4

Makrolar menüsünden Göreli başvuruları kullan seçeneği seçildikten sonra makro oluşturulmaya başlanır. Makro oluşturulduktan sonra C1 hücresi seçili iken çalıştırıldığında ekran görüntüsü aşağıdaki gibi olur.



Resim 2.18: *Makro çalıştırıldıktan sonraki ekran görünümü*

Yararlanılan ve Başvurulabilecek Kaynaklar

1. https://msdn.microsoft.com/

İŞLEM TABLOSU PROGRAMLAMA

3

Amaçlarımız

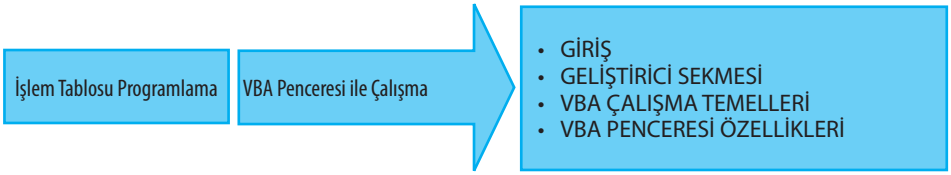
Bu üniteyi tamamladıktan sonra;

- VBA genel kullanımını açıklayabilecek,
- İşlem tablosu programında Geliştirici sekmesini kullanabilecek,
- İşlem tablosu VBA kullanımının genel mantığını tanımlayabilecek,
- VBA uygulama geliştirme mantığını ifade edebilecek,
- VBA Penceresi parçalarının çalışmalarını açıklayabilecek,
- VBA kod penceresi ile çalışabileceksiniz.

Anahtar Kavramlar

- VBA
- Geliştirici Sekmesi
- Kod
- Eklentiler
- Denetimler
- XML
- MAKRO
- VBA Özellikleri
- VBA Pencereleri
- Proje Penceresi
- Özellikler Penceresi
- Araç Kutusu

İçindekiler



VBA Penceresi ile Çalışma

GİRİŞ

İşlem tablosu programlarından olan Microsoft Office grubu içerisindeki Microsoft Excel (MS Excel), yazılım içerisinde birçok farklı hesaplama ve fonksiyonun hazırlanmış şablonları ile desteğini sunmaktadır. Bu özelliklere ek olarak, ikinci ünitede anlatılan Makro kullanımı ile işlemleri kaydetme ve tekrarlama konularında da kullanıcılara yardım etmektedir. MS Excel, makroların kullanımını **VBA** (Microsoft Visual Basic for Applications – Uygulamalar için Microsoft Visual Basic) adı verilen özellik sayesinde mümkün kılar. VBA kullanıcı ile dost bir kodlama penceresi yardımıyla Visual Basic (VB) programlama dilinin komutlarını kullanarak MS Excel'de gerçekleştirmek istenilen işlemleri kolayca yerine getirilmesini sağlar. VBA, programlama aracılığıyla yapılacak işlemlerin, otomatik olarak MS Excel tarafından yapılmasını sağlar. Bu programlama ortamı, VBE (Visual Basic Editor – Visual Basic Metin Düzenleyicisi) adı verilen bir pencere tarafından sağlanır ve otomatik olarak yapılması gereken işlemler VBE ortamında VB dili kullanılarak kodlanır. VB Metin Düzenleyici özelliğine sahip olmak için, öncelikle MS Excel üzerinde Geliştirici sekmesi açılmalı, bu sekme ile ortaya çıkan özelliklerden yardım alınmalıdır. Bir anlamda, makro kullanımı olarak da adlandırılabilecek VBA özellikleri ile tekrarlı işlemlerin otomasyona geçirilmesinde büyük kolaylık sağlanmış olur. Her ne kadar işlemleri kolaylaştırsa da, kodlama penceresi ile işlem tablosu programının içerisine belirli işleri yapan kod parçacıkları ilave edileceği için, bir güvenlik açığı da oluşur. Kötü niyetli yazılımlar, bu biçimdeki kodların arasına saklanarak çalıştırıldığı bilgisayarı ve yazılımlarını tahrip edebilirler. Bu sebeple VBA kullanımından önce işletim sisteminin ve üzerinde bulunan uygulama yazılımlarının güncel bir anti virüs programı ile korunuyor olduğundan emin olunmalıdır. VBA yardımıyla yaratılan özellikler, dosya geri dönüşüm kutusuna gönderildiğinde tahrip olmaktadır. Bu sebeple çalışılan dosyanın dikkatli biçimde korunması, ancak işi tamamen bittikten sonra silinmek amacıyla geri dönüşüm kutusuna gönderilmesi gereklidir.

VBA, Visual Basic for Applications (Uygulamalar için Microsoft Visual Basic) kelimelerinin baş harfleri kullanılarak kısa yazılmasıdır. En basit anlamda, tekrarlanan işlemleri otomatikleştirmeyi sağlayan kod parçacıkları yazma, düzenleme ve geliştirme ortamı sunan bir platformdur.

VBA ile çalışmadan önce, VBA'ya ulaşmayı sağlayan Geliştirici sekmesinin aktif hâle getirilmesi için gerekli işlemleri incelenecektir. Geliştirici sekmesi, butonlar vasıtasıyla çalıştırılacak birçok farklı özellik sunar. Bu özellikleri kısaca tanıdıktan sonra, VBA kullanımına yönelik ekranlar bütününe geçiş yapılacaktır. Farklı özellikleri barındıran pencereler bütününü, her bir pencerenin kullanımı açısından incelenecektir. VBA kullanımı için, programlamada kullanılacak Visual Basic programlama ortamı konusunda

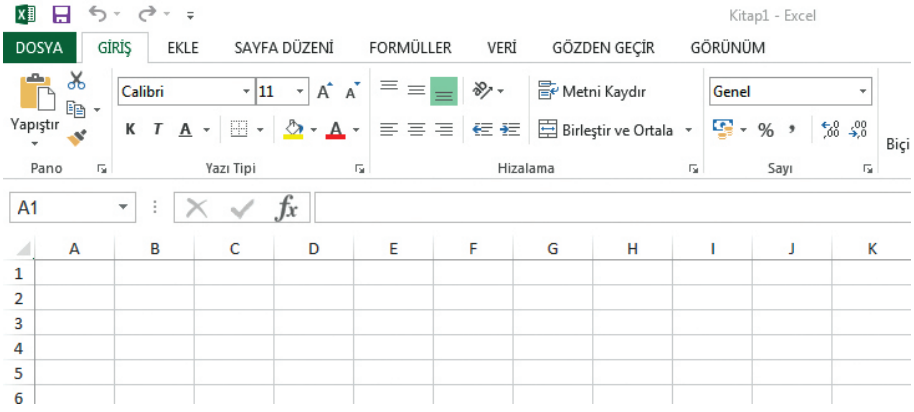
da bilgi sahibi olmak gereklidir. Bu nedenle, ünitenin ilerleyen bölümlerinde Visual Basic programlama ortamına da değinilecektir. Bu ortam ile ilgili ön bilgiler verilecek ve VBA kullanımına yönelik basit örnekler ile bu ortamın kullanımı ve kullanıcıya sağladığı kolaylıklar tanıtılacaktır.

GELİŞTİRİCİ SEKMESİ

Geliştirici sekmesi, makroları yaratabilmek ve programlama ortamına geçiş yapabilmek için MS Excel ile birlikte gelen özelliklerden birisidir. İlk kurulumda varsayılan olarak görüntülenmeyen "GELİŞTİRİCİ" sekmesinin görünür hâle gelmesi ve üzerinde yerleşik bulunan butonlar yardımıyla çeşitli programlama özelliklerine sahip olmak için öncelikle bir dizi işlem yapmak gereklidir. Boş bir sayfa ile açılan MS Excel programında, varsayılan sekmeler olarak, Resim 3.1'de de görüleceği gibi DOSYA, GİRİŞ, EKLE, SAYFA DÜZENİ, FORMÜLLER, VERİ, GÖZDEN GEÇİR ve GÖRÜNÜM sekmeleri görüntülenir.

Resim 3.1

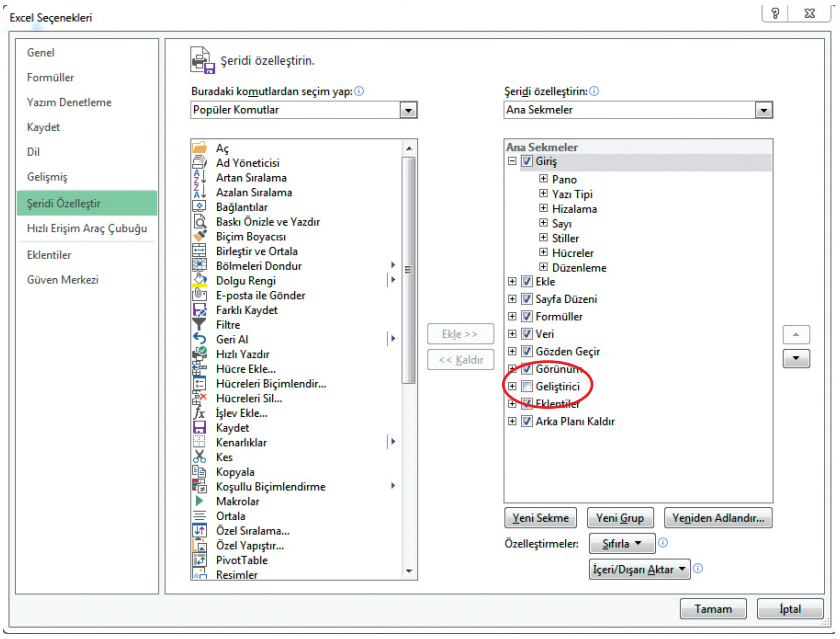
Varsayılan MS Excel Sekmeleri



Geliştirici sekmesini devreye sokmak için, DOSYA sekmesine gelinmeli ve Seçenekler tıklanmalıdır. Seçenekler tıklandığında, Excel Seçenekleri adı altında bir menü açılır. Bu menüden Şeridi Özelleştir seçildiğinde, üst tarafta bulunan sekmelerin tamamının görüntülendiği, devreye alınmamış sekmelerin ise işaretli olmadığı görünür. Geliştirici sekmesi işaretli olmayan ögelerden biridir. Resim 3.2'de bu sekme görülmektedir. Geliştirici sekmesinin sol kısmındaki (+) işareti tıklandığında alt menüleri görünür hâle gelir. Bu menüler Kod, Eklentiler, Denetimler, XML ve Değiştir'dir. Geliştirici sekmesinin başlığındaki kutu işaretlenerek Tamam seçildiğinde, sekme ve alt başlıkları MS Excel menü çubuğunda görünür ve kullanılabilir hâle gelir. Geliştirici sekmesini verimli kullanmak için öncelikle alt başlıklarında bulunan butonların çalıştırdığı eklentiler hakkında bilgi sahibi olmak gereklidir.

Resim 3.2

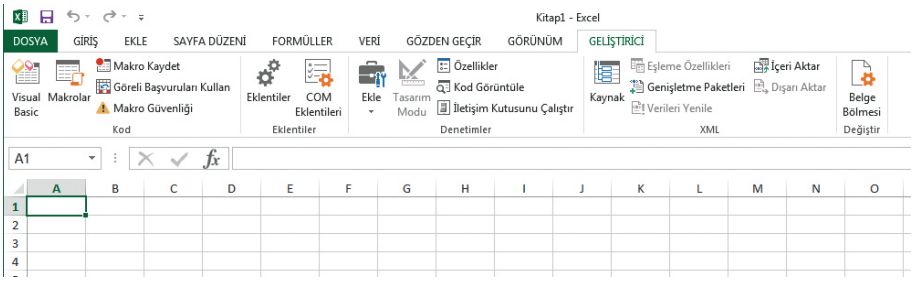
MS Excel Sekmeleri



Resim 3.3'te Geliştirici sekmesi devreye alınmış şekilde menü çubuğu görüntülenmektedir. Geliştirici sekmesi seçildiğinde, sekmeye bağlı bulunan kullanım özellikleri de görüntülenmektedir. Kod, Eklentiler, Denetimler, XML ve Değiştir menüleri, Resim 3.3'te görüldüğü gibi sekmenin alt başlıklarıdır. Her bir alt başlık, içerisinde butonlar hâlinde gösterilen farklı özellikleri de barındırır. Bu özellikler VBA ve Makro yaratma ve düzenleme işlemlerinde yardımcı işlevler sağlarlar.

Resim 3.3

Geliştirici Sekmesi Aktif İken MS Excel Sekmeleri



Kod Menüsü

Kod menüsünün butonları, Visual Basic, Makrolar, Makro Kaydet, Göreli Başvuruları Kullan ve Makro güvenliğidir.

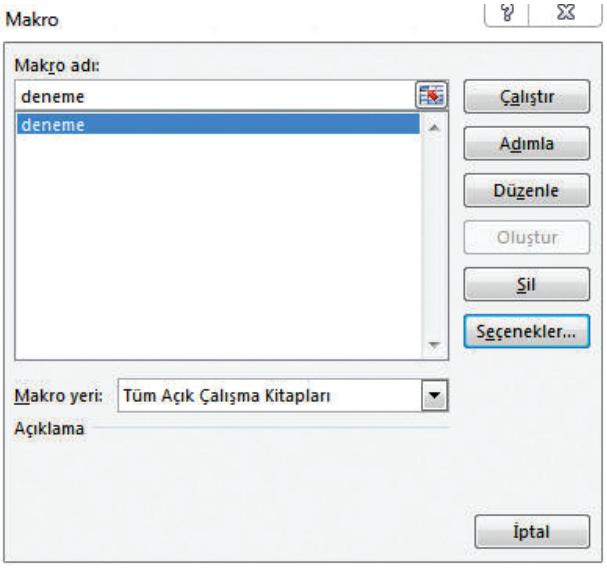
Visual Basic butonu tıklandığında yeni bir sayfada Microsoft Visual Basic for Applications (Uygulamalar için Microsoft Visual Basic) penceresi açılır. Bu pencere ile nasıl çalışılacağı ünitenin ilerleyen bölümlerinde anlatılacaktır.

Makrolar butonu, üzerinde çalışılan MS Excel sayfası için daha önce oluşturulmuş makroları görüntülemeyi sağlar. Açılan bir sayfa ile kayıtlı makrolar adları ile listelenir. Sağ tarafta bulunan butonlarla, kayıtlı makrolar üzerinde gerekli işlemler yapılır. Çalıştır

butonu seçili makronun baştan sona çalışmasını, Adımla butonu seçili makro işlemlerinin adım adım yapılmasını, Düzenle butonu makronun düzenlenmesini ve gerekli değişikliklerin yapılmasını, Oluştur butonu yeni bir makro oluşturmayı, Sil butonu seçili olan makronun silinmesini sağlar. Resim 3.4, Makro penceresini göstermektedir. Seçenekler butonu tıklandığında açılan Makro Seçenekleri penceresi ile seçili makronun klavye kısayolu belirlenebilir ya da değiştirilebilir. Açıklama bölümüne makro ile ilgili gerekli açıklamalar yazılabilir.

Resim 3.4

Makro Penceresi



Makro Kaydet butonu, makro oluşturmak ve yapılacak işlemleri tanımlamak için yardımcı görevi görür. Açılan pencere yardımıyla Makro Adı, Makroyu devreye sokmak için kullanılacak klavye kısayol tuş takımı, makronun saklanacağı dizin ve makro ile ilgili açıklamanın kaydedebileceği bir alan sağlar. Makro kaydet penceresinde gerekli bilgiler doldurulduktan sonra, makro kaydedilmeye başlanır. Bu sırada gerçekleştirilen tüm işlemler, MS Excel tarafından kayıt altına alınır. Makro Kaydet butonu, kayıt esnasında yerini Kaydı Durdur butonuna bırakır. İşlemleri tamamlayıp kayıt durdurulduğunda ise makro belirlenen Makro Adı ile tekrar çağrılmaya hazırdır. Sayfada bulunan her şeyi temizleyerek kaydedilen makroyu, Makro butonundan adı ile seçerek işleme sokulduğunda, kayıt ederken yapılan işlemlerin tamamının yeniden gerçekleştirildiği görülecektir.

Makro, tekrar çalıştırıldığında yalnızca kaydedildiği hücre blokları üzerinde çalıştığı görülecektir. Bir başka deyişle, makro vasıtasıyla otomatikleştirilen işlemlerin parametrik olarak istenen hücre blokları üstünde çalıştırılması yerine sabit hücre blokları üzerinde işlem yaptığı gözlenir. Bu durumu örnekle açıklamak için, A1 adresli hücreden başlayarak, A9 adresli hücreye kadar 1'den 9'a sıralı sayıları yazan bir işlemi otomatikleştiren bir makro olsun. Makro, sadece A1-A9 hücre aralığında geçerli olacaktır. Tekrar çalıştırıldığında, bu hücreler başka sayılar ile dolu olsa bile üzerine yeni sayılar yazma işlemi yapılacak ve eğer ilk sayılar 1'den 9'a kadar sıralı sayılarsa, makro hiç çalışmıyormuş gibi görüntülenecektir. Yapılan işlemin daha sonra başka hücre bloklarında da çalışmasını sağlamak için Göreli Başvuruları Kullan butonu yardımcı olur. Makro kayıt aşamasında Göreli Başvuruları Kullan butonu seçili bir şekilde kayıt başlatılırsa başlangıç değeri için gerekli hücre adresi seçildikten sonra makro, sayfanın tüm hücreleri için devrede olacaktır. Örneğe dö-

nülürse, A1-A9 adresli hücreler arasında 1'den 9'a sıralı sayıların yazılması makro kaydını, Göreli Başvuruları Kullan özelliği seçili iken yapılırsa Makro B5 hücresi üzerinde çalıştırıldığında, B5 adresli hücre içeriği 1 sayısını almak üzere aşağıya doğru B13 hücresine kadar sayıların otomatik olarak yazıldığı görülecektir.

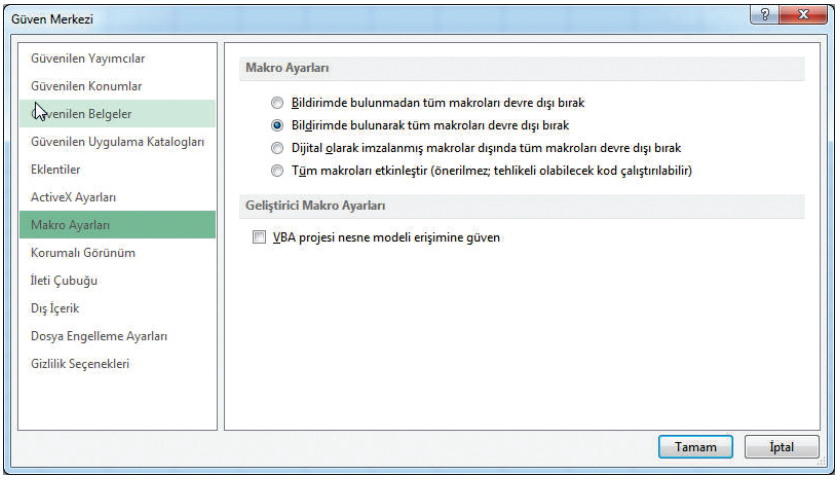
Bilgisayarınızda bulunan MS Excel yazılımı ile örnekteki makroyu oluşturunuz.

SIRA SİZDE 1

Makro kullanımındaki güvenlik açıklarına daha önce değinilmiştir. Virüslerden en çok etkilenen kod parçacıkları olan makroların güvenlik ayarlarını değiştirmek için, Makro Güvenliği butonundan yararlanılır. Resim 3.5'te görüleceği gibi bu buton tıklandığında Güven Merkezi penceresi açılarak Makro Ayarlarına erişim sağlanır.

Resim 3.5

Güven Merkezi Görünümü



Az önce oluşturduğunuz Makronun, farklı hücreler için de çalışmasını sağlayacak bir uygulama ile test ediniz.

SIRA SİZDE 2

Güvenliğin sağlanması için gerekli durumlara göre, Bildirimde bulunmadan tüm makroları devre dışı bırak, Bildirimde bulunarak tüm makroları devre dışı bırak, Dijital olarak imzalanmış makrolar dışında tüm makroları devre dışı bırak ve Tüm makroları etkinleştir seçeneklerinden bir tanesi seçilebilir. Geliştirici makro ayarlarında, VBA projesi nesne modeli erişimine güven seçeneği işaretlenebilir.

Eklentiler Menüsü

Eklentiler menüsünün iki alt başlığı Eklentiler ve COM Eklentileridir.

Çoğu eklenti üç farklı tür altında kategorilere ayrılabilir (1):

- Excel eklentileri: Bunlar tipik olarak Excel eklentisi (.xlam), Excel 97-2003 eklentisi (.xla) veya DLL eklentisi (.xll) dosyalarını içerir veya otomasyon eklentileridir. Çözücü ve Çözümleme Araç Takımı gibi bazı Excel eklentileri MS Excel'i veya Microsoft Office'i yüklemenizden sonra kullanılabilir. Bu eklentileri kullanmak için etkinleştirilmesi yeterlidir.
- Karşıdan yüklenebilir eklentiler: Excel için diğer eklentiler Office.com adresindeki Yüklemeler sayfasından yüklenip kurulabilir. Örneğin, Excel için Başlangıç sekmesini karşıdan yükleyip kurabilirsiniz. Bu sekme Excel kullanmaya başlamayı

sağlayan eğitimlerin, gösterilerin ve diğer içeriğin bağlantılarını içerir. Bu eklentiyi yükledikten sonra, MS Excel yeniden başlatıldığında, sekme şeritte kullanılabilir duruma gelir. Bu sekmeyi ve içindeki komutları Excel şeridinden kaldırmak için, Denetim Masası'nı kullanarak programı kaldırın.

- Özel eklentiler: Geliştiriciler ve çözüm sağlayıcılar genellikle özel COM eklentisi, otomasyon eklentileri, VBA eklentileri ve XLL eklentileri tasarlar. Bunları kullanabilmek için yüklenmesi gerekir.

Eklentiler butonu tıklandığında çıkan eklentiler penceresinde, MS Excel sürümüne bağlı olarak, en iyileştirme ve denklem çözümüne yardımcı araç olan Çözücü Eklentisi, istatistiksel ve mühendislik amaçlı çözümler için veri çözümleme araçları sağlayan Çözümleme Araç Takımı, çözümleme araç takımının VBA destekli araçlarını barındıran Çözümleme Araç Takımı, VBA ve Avro para birimi dönüştürme ve biçimlendirme araçlarını sağlayan Euro Para Birimi Araçları eklentilerini devreye almayı ve devreden çıkartmayı sağlayan seçim ekranları bulunur. Bunların dışında bilgisayarda yüklü uygulamalara bağlı olarak farklı eklentiler de bu pencere yardımıyla devreye sokulabilir. Pencerede bulunan otomasyon butonu ile farklı ortamlarla otomasyon sağlanabilir.

COM Eklentileri penceresi ise Inquire, Microsoft Office PowerPivot for Excel 2013 ve Power View gibi farklı uygulama yazılımları ile birlikte çalışma imkânı sağlar. COM Eklentileri penceresi ile farklı eklentiler sisteme dâhil edilebileceği gibi, gerekli olmayan eklentiler de kaldırılabilir. Eklentiler penceresinde olduğu gibi COM Eklentileri penceresinde de kullanıma açılmak istenen eklentinin kutusunun işaretlenmesi yeterlidir.

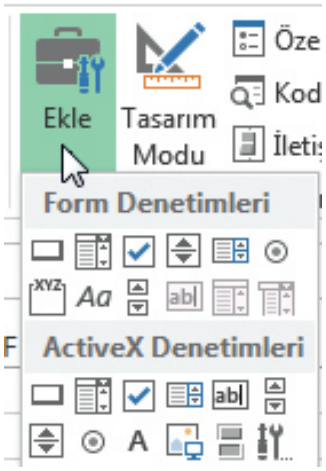
Denetimler Menüsü

MS Visual Basic, Nesneye Dayalı bir programlama ortamıdır. Yazılan program kodunun çalışması için ilgili nesneye bir eylem uygulanmalıdır.

Denetimler menüsü **MS Visual Basic** programlama ortamının MS Excel ortamında kullanıma sunulacak olan nesnelerinden oluşur. Ekle penceresi ile Form ve Active X nesnelerini kitap üzerine ekleme imkânı vardır. Form ve Active X nesneleri kullanıcı ile Makro arasında etkileşimli bir ortam sağlar. Bu nesnelerden bazıları, Buton, Grup Kutusu, Onay Kutusu, Liste, Açılan Liste, Değer Değiştirme Düğmesi, Metin Kutusu gibi nesnelerdir. Eklenen nesneye göre kodlama yapılarak aracın kullanımı sağlanır. Ekle menüsünün alt ögeleri Resim 3.6'da görünmektedir.

Resim 3.6

Ekle Menüsü Görünümü



Tasarım Modu, ActiveX nesnelerinin tasarımlarına yardımcı olmak için seçilmesi gereken menüdür. Tasarım Modu devre dışı bırakıldığı zaman, ActiveX nesne ile ilgili düzenleme yapma imkânı yoktur. Tasarımı tamamlanan nesnenin kullanımına geçmek için ise Tasarım Modu kapatılmalıdır.

MS Visual Basic programlama ortamı mantığına göre, her bir nesne kendine ait özelliklere sahiptir. Bu özellikler nesnenin rengi, boyutları, üzerindeki yazıların çeşitleri gibi görsel özellikler olabileceği gibi, kullanıma açık ya da kapalı olması, kod yazarken kullanılacak nesne adı, görünür ya da gizli olması gibi tasarımsal özellikler de olabilir.

MS Visual Basic, nesneye dayalı bir programlama ortamıdır. Bir başka deyişle, program kodlarının devreye girmesi için nesne-eylem ikilisinin bir araya gelmesi gereklidir. Nesneye eylem uygulandığı zaman devreye girecek olan kodlar, Visual Basic Metin Düzenleyicisi üzerine yazılacaktır. Kod Görüntüle menüsüyle bu kodları görüntülemek ve gerektiğinde değişiklik yapmak mümkündür.

İletişim Kutusunu Çalıştır menüsü, daha eski sürüm MS Excel ile çalışırken kullanılan menülerden birisidir. İletişim kutularının yerini yeni sürümlerde kullanıcı formları almıştır.

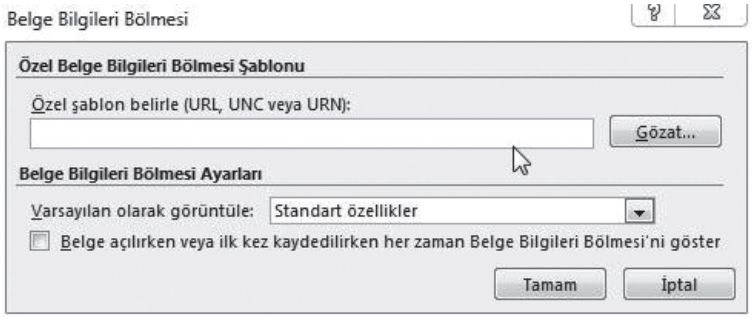
XML Menüsü

XML (Extensible Markup Language - Genişletilebilir İşaretleme Dili), verilerin yapılarını bilinen biçimlere dönüştürerek, diğer uygulamalar ile uyumlu veriler oluşturmak için kullanılır. MS Excel sayfasında bulunan verilerin bilinen biçimlere uygun olarak işaretlenmesi ile XLM biçimine sahip XML dosyaları oluşturulur. XML biçimine dönüştürülmüş veriler, farklı uygulama yazılımları tarafından rahatlıkla kullanılabilirler.

Değiştir Menüsü

Değiştir menüsünün tek sekmesi, Belge Bölmesi sekmesidir. Tıklandığında açılan pencere ile özel şablonlar belirlenebilir (URL, UNC, URN vb.) ve belge bilgileri bölmesinin ayarları değiştirilebilir. Resim 3.7 bu pencereyi göstermektedir. İstenildiğinde, belge açılırken veya ilk kez kaydedilirken her zaman Belge Bilgileri Bölmesinin gösterilmesi için alt tarafta bulunan bir işaretleme kutusunun seçilmesi gerekir.

Resim 3.7



Belge Bilgileri Bölmesi Penceresi

VBA ÇALIŞMA TEMELLERİ

Ünitenin ilk kısmında Makrolar ile nasıl çalışılacağı anlatılmıştır. Geliştirici sekmesini devreye soktuktan sonra, Makro yaratmak ve tekrar kullanmak özellikle tekrarlı işlemleri gerçekleştirmeyi kolaylaştırmaktadır. Daha uzman sistemler kurmak, işlemleri aynı zamanda nesnelerle gerçekleştirmek için ise VBA geliştirme ortamından faydalanılır. VBA ile yaratılan otomasyon sistemlerinin genel adı makrodur. Ancak makrolar ile belirli işlemleri kaydetmek ve kaydı durdurarak işlemi tekrar etmekten daha detaylı işlemler gerçekleştirilebilir. VBA'nın açılımının MS Visual Basic adı verilen nesneye dayalı bir programlama ortamından geldiği daha önce açıklanmıştır. Şimdi, bu programlama ortamı, çalışma mantığı ve basit kullanımı ile ilgili bilgilere değinilecektir.

MS Visual Basic programlama ortamının temeli olan Basic (Basit) programlama dili, adından da anlaşılabileceği gibi kolay bir programlama dilidir. 1964 yılında ABD'de ortaya çıkan programlama dili, amatör kullanıcılar ve hobi kullanıcılarına hizmet vermek amacıyla geliştirilmiştir.

MS Visual Basic, Basic (Beginners-All Purpose Symbolic Instruction Code - Yeni Başlayanlar için Çok Amaçlı Sembolik Talimat Kodu) anlamına gelen programlama dilinin, Microsoft firması tarafından görsel bir ortama taşınmasıyla geliştirilmiş bir programlama dilidir. Kolay bir dil olan Visual Basic, aynı zamanda MS Windows işletim sistemleri ile de uyumludur. Görsel, olaylara ve nesnelere dayanan bir programlama dilidir. Görsel programlama, ortamda bulunan nesneler aracılığıyla etkileşimli programlamayı amaçlamaktadır. Nesneye dayalı tüm programlama dillerinde olduğu gibi, Visual Basic programlama dilinde de asıl ögeler nesnelerdir. Programlama yapmak için en az bir nesne gereklidir. Bu nesne, sayfada bulunan bir hücre olabileceği gibi çalışma sayfasının kendisi

ya da sonradan üzerine eklenen buton, metin kutusu, çerçeve gibi standart form nesneleri ya da Active X nesneleri de olabilir. MS Visual Basic mantığına göre aynı zamanda yazılan program kodlarının devreye girebilmesi için belirtilen nesneye, eylem uygulanmalıdır. Nesne-Eylem ikilisi olmadığı sürece yazılan program kodları çalışmayacaktır. Örnek vermek gerekirse, sayfa üzerine yüklenecek bir Buton nesnesi için en sık kullanılan eylem, tıklanmasıdır. Hem makro oluştururken hem de kod sayfası ile gerçekleştirilmek istenen eylemler dizisi, butonun tıklanması sonucunda devreye girer.

SIRA SİZDE 3

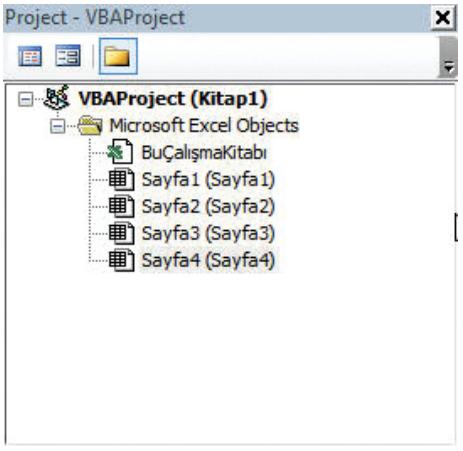
Verilen düğmenin tıklanması örneğindeki nesne ve eylemi tanımlayınız.

Ünitenin konusu olan VBA ise, Microsoft firması tarafından Microsoft Office ürünleri içinde otomasyon oluşturabilmek için geliştirilmiş bir platformdur. VBA sayesinde MS Excel üzerinde yapılacak birçok işlem otomatik hâle getirilebilir.

Bilindiği gibi, MS Excel boş bir sayfa ile açıldığında, Kitap1 olarak adlandırılır. Standart olarak verilen bu dosya ismi daha sonra istenildiği gibi değiştirilebilir. Çalışma kitabı içerisinde birden fazla MS Excel sayfası bulunmaktadır. Gerektiğinde sayfa ekleme işlemleri ile sayfa sayısı da arttırılabilir. VBA kullanımından önce, uygulama kodunun tek bir sayfa için mi, yoksa içindeki tüm sayfalarla beraber kitabı mı kapsayacağına karar vermek gereklidir. Her iki ortam için de ayrı ayrı kod yazmak ve farklı işlevleri gerçekleştirmek mümkündür. Bazı durumlarda, MS Excel kitabı içerisinde birden fazla sayfa ve her bir sayfanın kendine özgü işlemleri olabilir. Böyle durumlarda her bir sayfa için ayrı ayrı ve bir tane de kitabın tamamı için kod yazma imkânı bulunur. Gelişmiş bir otomasyon gerçekleştirmek istenen durumlarda bu sıkça karşılaşılan bir kodlama yöntemidir. Resim 3.8'de dört adet çalışma sayfasına sahip bir MS Excel kitabının VBA proje penceresi görüntülenmektedir.

Resim 3.8

VBA Proje Penceresi



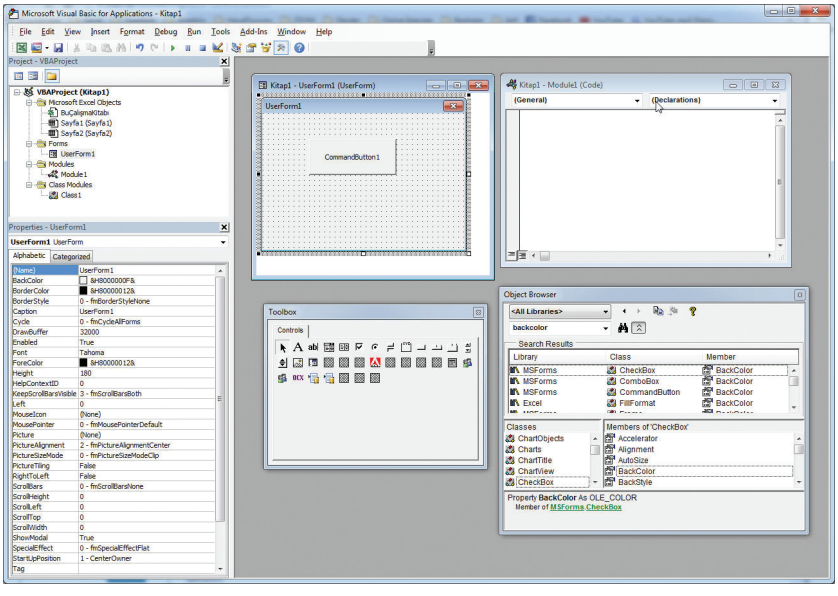
VBA proje penceresinde görüntülenen "Bu Çalışma Kitabı", kitabın tamamı, her bir sayfa ise kendilerine ait kısımlar için kod penceresini açacaktır. Beş ögenin tümü, MS Visual Basic programında bulunan Form yapısı gibi birer nesnedir.

VBA pencerelerini tek tek incelemeden önce programlama ortamının çalıştırılması hakkında bilgi vermek gereklidir. Pencere ilk açıldığında karşılaşılan kısım Tasarım Zamanı (Design Time) olarak adlandırılır. Tasarım zamanında nesneler ve onlara uygulanan eylemlere göre gerekli kodlar yazılır. VBA ortamında yazılan kodlar da Makro olarak kaydedilmektedir. Bir makroyu çalıştırmak için, Obje olarak adlandırılan MS Excel sayfasına dönerek makroyu çalıştırmak yeterlidir. Projeye VBA ortamında sayfalar dışında, işlemlerin gerekliliğine göre farklı ögeler de eklenebilir. Bu tür ögeler eklendiğinde ve gerekli kodlar yazıldıktan sonra yapılması gereken, yazılan kodların hatalı olup olmadığının ve istenilen işlemleri gerçekleştirip gerçekleştirmediğinin testi için Çalışma Zamanına (Run Time) geçmektir. Bu işlemi gerçekleştirmek için Run (Çalıştır) menüsünden Run Makro (Makroyu Çalıştır) tıklanmalı ya da F5 tuşuna

basılmalıdır. VBA ortamı, yazılan kodu önce kontrol edecek, herhangi bir yazım hatası ya da nesne uyuşmazlığı durumunda hatayı bildirecektir. Eğer bir hata yoksa program, nesne-eylem ikilisini devreye sokarak test edilecektir. VBA ortamının hata vermemesi, her zaman programın hatasız olduğu anlamına gelmemektedir. Mantıksal hatalarda ortam, herhangi bir hata göstermezken programın çalışması esnasında istenilen işlemler gerçekleşmeyecek, hatta çoğu zaman hatalı işlemler gerçekleşecektir. Örnek olarak, bir butona sahip bir form ortamı olsun. Butona basılınca devreye girmesi istenilen işlem, iki hücre değerini alarak toplamının başka bir hücreye yazdırılması olsun. Toplama işleci yerine yanlışlıkla çıkartma işleci kullanıldığını varsayalım. Birinci hücreye 5, ikinci hücreye ise 2 değerini girelim. Programı çalışma zamanına geçirdiğimizde herhangi bir hata vermeyecektir. Butonu tıkladığımızda ise sonuç olarak 7 değeri beklenirken, 3 değeri alınacaktır. Bu, yazım ya da nesnelerde bir hata olmadığı ancak mantıksal olarak programın kodlanmasında hata yapıldığını gösterir. Burada mantık hatası, toplama işlemi gerçekleştirilmesi istenen bir makroda çıkartma işlecinin kullanılması olmuştur. Mantıksal hataların düzeltilmesi, yazım hataları ya da nesnesel hataların düzeltilmesi kadar kolay değildir. Yapılmak istenen işlemlerin tamamı tekrar gözden geçirilmeli, eğer gerekiyorsa programın algoritması değiştirilmeli ve kod tekrar yazılmalıdır.

VBA PENCERESİ ÖZELLİKLERİ

VBA ortamını oluşturan ve her birinin kendine özgü görevleri olan pencereleri kısaca tanımak önemlidir. MS Excel programında, Geliştirici sekmesinden Visual Basic tıklanarak açılan VBA ortamı, farklı pencerelerden ve menülerden oluşmaktadır. Basit etkileşimli makrolar yaratmak için kullanılacak pencere ve menüler, Proje Penceresi (Project Window), Özellikler Penceresi (Properties Window), Araç Çubuğu (Toolbox), Nesne Tarayıcısı (Object Browser) gibi pencereler ve Görünüm (View), Ekle (Insert), Çalıştır (Run) gibi menülerdir. Kod (Code) penceresi ise Proje Penceresinde yer alan tüm nesneler için ayrı ayrı kodlar yazmayı sağlayan pencereler bütünüdür. Resim 3.9'da VBA penceresinin genel yapısı görülmektedir.



Resim 3.9

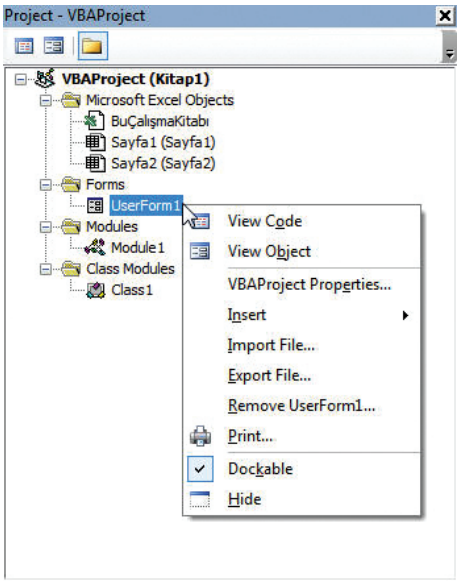
VBA Penceresi

Proje Penceresi (Project Window)

Resim 3.8'de görüldüğü gibi proje penceresi, VBA projesine bağlı bulunan tüm ögelerin aynı anda görüntülenmesini ve gerekli ögenin kolayca seçilmesini sağlar. Ekle menüsü ile eklenebilen Kullanıcı Formları, Modüller ve Sınıflar da proje penceresinde yer alacaktır. Proje Penceresi bir anlamda projenin tüm ögeleri için tarayıcı gibi çalışacaktır. Her bir nesnenin üzerinde sağ tıklayarak, o nesneye ait kod ya da obje kodun görüntülenmesi sağlanır. Farklı nesneler eklenmiş proje penceresi Resim 3.10'da görülmektedir.

Resim 3.10

VBA Proje Penceresi



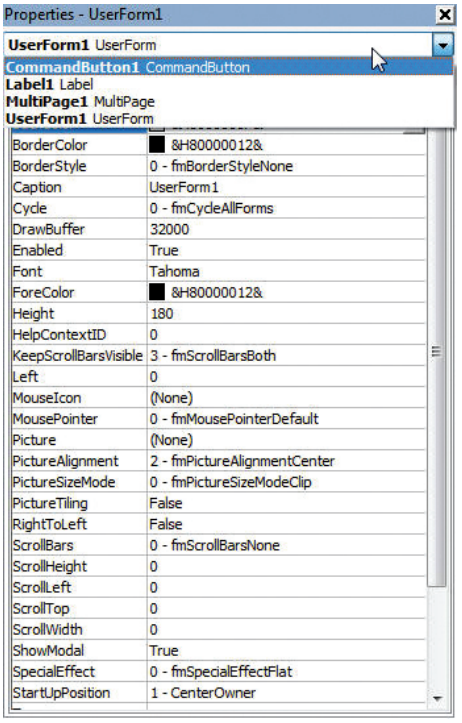
Çalışma kitabı ve sayfalar ana öge oldukları için Proje penceresi ile silinmelerine olanak yoktur. Çalışma kitabı silinirse proje gerçekleşmez. Fakat sayfaları eklemek ve silmek istersek, MS Excel penceresinden bu işlem gerçekleştirilebilir. MS Excel penceresinde yapılan güncelleme VBA Proje penceresini de güncelleyecektir. Bu ögelerin dışında proje penceresinde sağ tıklanma ile Kullanıcı Formları (User Forms), Modüller (Module) ve Sınıflar (Class) oluşturulabilir ve Sil (Remove) işlemi ile projeden silinebilirler.

Özellikler Penceresi (Properties Window)

VBA ortamının nesnelerle çalıştığını açıklamıştık. Özellikler penceresi tüm bu nesneler için ayrı ayrı özellikleri görmemizi ve değiştirmemizi sağlamaktadır. Her nesnenin kendine ait özellikleri vardır. Resim 3.10 tekrar incelenirse ana yapıda 6 adet farklı nesne görülür. Bunlar BuÇalışmaKitabı, Sayfa 1, Sayfa 2, User Form1, Module 1 ve Class 1'dir. Proje penceresinde görünenlerin dışında, kullanıcı formu ya da Sayfa üzerinde de farklı nesneler (Butonlar, Metin Kutuları, Listeler vb.) bulunabilir. Tüm bu sayılan nesneler seçildiğinde, kendilerine ait özellikler penceresi görüntülenecektir. Özellikler penceresinin üst kısmında bulunan açılan menü sayesinde projede bulunan ana nesneye gelindiğinde ona bağlı bulunan tüm nesneler görüntülenir ve seçilebilir. Resim 3.11'de açılan menü sayesinde Kullanıcı Formu (User Form) üzerinde yerleşik bulunan Buton (Command Button), Çoklu Sayfa (Multipage) ve Etiket (Label) nesneleri görüntülenmektedir. Özellikler penceresinde ise seçili bulunan kullanıcı formunun özellikleri görüntülenmektedir.

Resim 3.11

Özellikler Penceresi



Özellikler penceresinde dikkat edilmesi gereken bir önemli kısım da her özelliğin kendine ait değerlerinin olmasıdır. Bir nesnenin en önemli özelliği (Name) olarak gösterilen İsim özelliğidir. Bu özelliğe göre nesnelere gerekli kodlar yazılacağı için kod yazarak değiştirmenin mümkün olmadığı tek özelliktir. Bunun dışındaki özellikler, tasarım zamanında pencereden değiştirilebildiği gibi, çalışma zamanında da kod aracılığıyla yeni değerler alabilir. Atanacak yeni değerler için öncelikle özelliğin ne tür değerler aldığını bilmek gereklidir. Örnek vermek gerekirse Başlık (Caption) özelliği alfa numerik karakter değeri alır, Renk (BackColor, ForeColor, vb.) özellikleri onaltılık değerlere sahiptir, Yükseklik (Height) ve Genişlik (Width) özellikleri piksel değeri alırlar, Çerçeve stili (BorderStyle), başlangıç pozisyonu (StartupPosition) gibi özellikler indekslenmiş değerlere sahiptir. Yazı tipi (Font) değişkeni, seçimli değer alan özelliklerdendir. Özellikler penceresinden değiştirirken bu değerleri belirlemek daha kolayken, kod penceresinde değişiklik yapmak için özelliğe uygun doğru değer girilmelidir. Özelliği kod ile değiştirmek için;

Nesne İsmi . Özellik İsmi = ÖzelliğinYeni Değeri

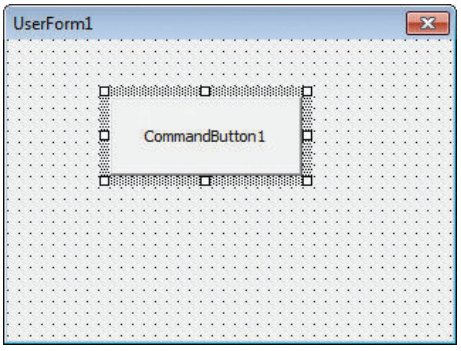
formülünden faydalanılır.

Kod yazımı sırasında nesnenin özelliğine uygun olmayan bir değer atamak, VBA derleyicisinin hata vermesine ve programın çalışmamasına sebep olacaktır.

DİKKAT

Araç Çubuğu (Toolbox)

Araç çubuğu, sadece projeye bir kullanıcı formu eklendiğinde açılan bir penceredir. Bu pencereyi, MS Excel ana ekranında kullandığımız Geliştirici sekmesinin Denetim Menüsü, Ekle butonuna benzetmek mümkündür. Hatırlanacak olursa, ekle butonu tıklandığı zaman, MS Excel'de aktif bulunan sayfa üzerine nesneler eklenebilir. Aynı şekilde, projede bir kullanıcı formu tanımlandığında aktif hâle gelen Araç Çubuğu da form üzerine eklenecek nesneleri barındırmaktadır. Kontroller (Controls) olarak adlandırılan nesnelere gerektiği durumlarda yenilerini eklemek de mümkündür. Nesnelerin altında bulunan boş alana sağ tıklayarak Ek Kontroller (Additional Controls) seçilerek açılan pencere sayesinde istenilen nesnenin görüntülenmesi ya da görünümden çıkarılması sağlanabilmektedir. Kullanıcı formu üzerine eklenen nesne veya nesneler, kendi özellikleri ile birlikte gelmektedirler. Nesnenin adı ve 1'den başlayarak sayısı İsim özelliğine atanır. Kullanıcı isterse bu özelliği tasarım zamanında değiştirebilir. Form üzerinde nesneyi seçmek için üzerinde tek tıklamak yeterlidir. Nesnenin etrafında çıkan sekiz adet nokta ile boyutlandırma da fare ile yapılabilmektedir. Resim 3.12, kullanıcı formu üzerine eklenmiş bir butonun seçili olma durumunu göstermektedir.



Resim 3.12
Kullanıcı Formu Üzerinde Seçili Buton

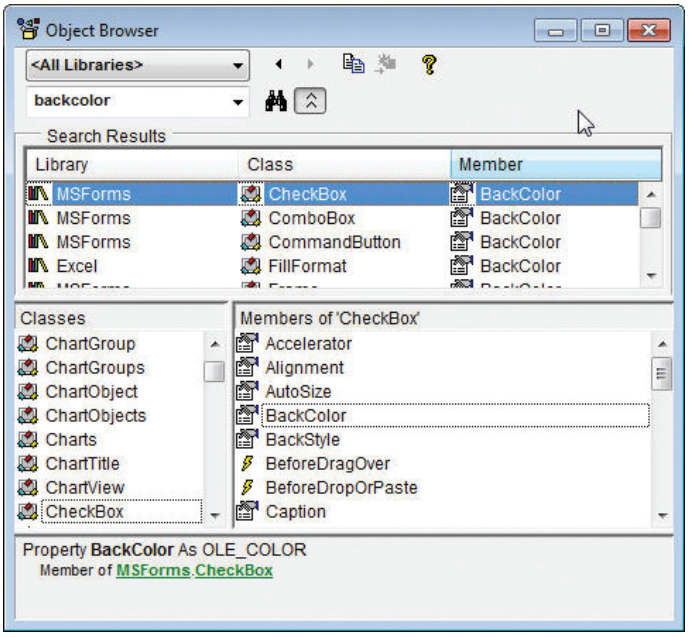
Nesne Tarayıcısı (Object Browser)

Nesne tarayıcı, kod yazımı esnasında kullanıcının en büyük dostudur. Nesnelerle yapılabilecek eylemlerin listelendiği nesne tarayıcıda aynı zamanda nesne özellikleri ve bileşenleri de hiyerarşik bir şekilde yer alır. Gerçekleştirilmek istenen eyleme göre arama imkânı da veren nesne tarayıcı, sınıflar ve o sınıfa ait üyelerin yer aldığı iki kısımdan oluşan bir pen-

cerede sunulur. Arama yapmayı sağlayan menü yapısı sayesinde istenilen özelliğe ulaşmak kolaylaşır. Resim 3.13'te Arkaplan Rengi (Backcolor) özelliği aratılan bir nesne tarayıcısı görülmektedir.

Resim 3.13

Nesne Tarayıcı



SIRA SİZDE 4

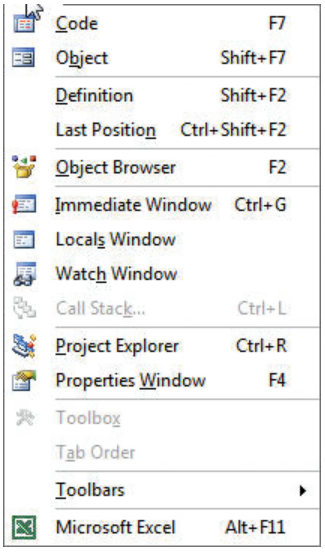
Nesne tarayıcıdan Başlık (Caption) özelliğini aratarak hangi sınıfa ait olduğunu bulunuz.

Görünüm (View) Menüsü

MS Excel ve diğer ofis programlarında olduğu gibi, yukarıdaki menüler, program ayarlarını yapmak ve gerekli olan program eklerini aktif hâle getirmek için kullanılır. VBA ortamında da birçok menü bulunmaktadır. Bu menülerden bir kaçı, bu ünitede ele alınacaktır. Görünüm menüsünün ilk iki alt başlığı, Kod (Code) ve Obje (Object)'dir. Kod, nesneye ait kodlamanın yapılacağı, gerekli değişken atamaları ve prosedürlerin oluşturulacağı VBE (Visual Basic Editor)'ye ulaşmayı sağlayan menü ögesidir. Boş bir sayfa olarak açılan Kod sayfasının kullanımı izleyen bölümde anlatılmaktadır. Bir diğer önemli öge olan Obje ise nesneyi yani sayfaları, kullanıcı formlarını ya da bu nesnelerin üzerine yerleştirdiğimiz diğer nesneleri (Buton, Liste, Etiket vb.) görüntülemeyi sağlayan bölümdür. Hem Kod, hem de Obje kısımları tasarım zamanında çalışmaktadır.

Resim 3.14

Görünüm Menüsü



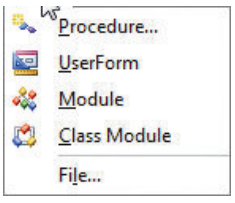
Görünüm menüsüyle, VBA üst araç çubuğunda da kısa yolları bulunan Proje Penceresi, Özellikler Penceresi, Nesne Tarayıcı, Araç Çubuğu gibi nesneleri görünür hâle getirmek mümkündür. Görünüm penceresi aynı zamanda oluşturulan kodun işleyişini görmek

ve değişkenlerin aldığı değerleri inceleyerek mantıksal hataları daha çabuk düzeltmeyi amaçlayan İzleme Penceresi (Watch Window), Yerel Pencere (Locals Windows) ve Acil Penceresi (Immediate Window) gibi pencerelerin görünür hâle gelmesini de sağlar. Resim 3.14 Görünüm menüsü ve alt başlıklarını göstermektedir.

Ekle (Insert) Menüsü

Bilindiği gibi tüm Microsoft ürünlerinde Ekle menüsü, üzerinde çalışılan uygulama sayfasına farklı nesneler eklemek için kullanılır. MS Word ile yazılan bir dosyaya resim eklenmesi istendiğinde Ekle menüsünün kullanımı buna örnektir. VBA ortamında da Ekle Menüsü, ortama bir Prosedür (Procedure), Kullanıcı Formu (UserForm), Modül (Module) ya da Sınıf Modülü (Class Module) eklemek için kullanılır. VBA ortamına farklı bir dosya eklenmek istendiğinde de Ekle Menüsünden faydalanılır. Resim 3.15 Ekle menüsünü göstermektedir.

Şekil 3.15

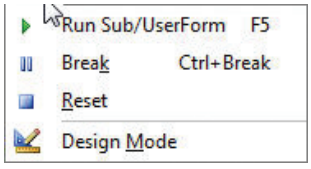


Ekle Menüsü

Çalıştır (Run) Menüsü

VBA ortamı için Tasarım Zamanı ve Çalışma Zamanı olmak üzere iki farklı zamandan bahsedilebileceğine değinilmiştir. Tasarım Zamanı gerekli nesnelerin oluşturulduğu ve nesnelere eylemler uygulanınca yapılması gerekenlerin tanımlandığı kodların yazıldığı zamandır. Bu zamanda program oluşturulur ve otomasyon sağlanır. Programın çalışması içinse Çalışma Zamanına geçmek gereklidir. Bu zaman içerisinde nesneye eylem uygulanarak yazılan kodların belirlenen işlemleri gerçekleştirmesi sağlanır. Tasarım zamanında yapılan tüm işlemler, çalışma zamanında geçerlilik kazanır. Resim 3.16'da görülen Çalıştır menüsü alt başlıkları ile çalışma zamanına geçiş (Run Sub/User Form), gerekli durumlarda programa mola vermek için kısa süreli durdurma (Break), programı tamamen sonlandırma (Reset) ve gerekli durumda tasarım zamanına (Design Mode) geçiş için kullanılan seçenekleri sağlar.

Şekil 3.16



Çalıştır Menüsü

Kod (Code) Penceresi

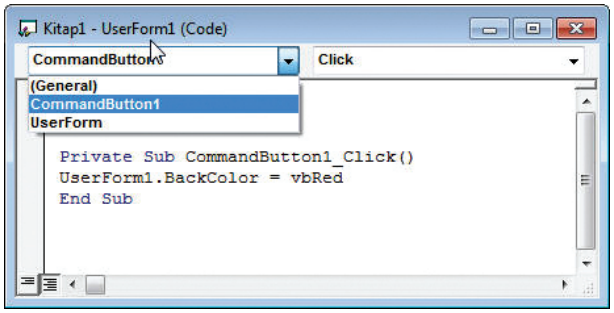
Kod Penceresi, VBE (Visual Basic Editor - Visual Basic Metin Düzenleyicisi) adı da verilen bir pencerenin açılmasını sağlar. Bu pencere ile nesne-eylem ikilisinin çalışma zamanında gerçekleştireceği işlemler tanımlanır. Kod penceresinde gerekli kodları oluşturmak için VBA programlama dili hakkında bilgi sahibi olmak gereklidir. Boş bir sayfa hâlinde çıkan pencerenin üst tarafında iki adet açılır liste bulunmaktadır. Kod penceresinde herhangi bir şey yazılı değilken üst açılır listelerde Genel (General) ve Deklarasyon (Declaration) bulunur. Genel listesi nesneleri seçmeyi sağlar. Liste açıldığında işlev kazandırılmak istenen nesnelerin tamamı görüntülenir. Eğer ana nesneye bağlı farklı bir nesne yoksa Genel listesinde sadece ana nesne görünecektir. İlk yapılması gereken işlem, nesneyi seçmektir. Nesne seçildiği an, en sık kullanılan eylemi otomatik olarak yanına yazılır. Nesne için farklı bir eylem seçilmek istendiğinde Deklarasyon listesi açılmalı ve gerekli eylem seçilmelidir.

Nesne ve eylem seçildiğinde, yazılacak olan kodun, o nesne-eylem ikilisine özel olduğunu belirten bir başlangıç yazısı ve bir bitiş yazısı otomatik olarak pencerede belirir. Örnek vermek gerekirse; VBA üzerine bir kullanıcı formu eklensin. Form üzerine araç çubuğundan alınan bir buton yerleştirilsin. Kod penceresi açıldığında Genel açılan menü-

sünün altında artık Kullanıcı Formu (UserForm) ve Buton (CommandButton1) ayrı ayrı görünecektir. Düğmenin adı seçildiğinde, en sık kullanılan eylemi olan Tıklama (Click) ile beraber başlangıç ve bitiş satırları oluşturulacaktır. Örnek, Resim 3.17'de görülmektedir.

Şekil 3.17

Düğmenin Tıklanmasına Yönelik Kod Sayfası



Private Sub (Alt Yordama Özel) yazılacak olun kodun başlığını, End Sub (Alt Yordamı Sonlandır) ise sonlandığı kısmını göstermektedir. Private Sub ikilisinin yanında nesnenin ismi olan CommandButton1 ve eylemi olan Click görüntülenmektedir. Kod olarak yazılacak her şey, Private Sub ve End Sub satırlarının arasındaki satırlara yazılacaktır. Örneğe devam edilecek olursa, çalışma zamanına geçildiğinde, buton tıklandığı zaman, formun arka plan rengi kırmızı olsun. Bunun için gerekli kod hatırlanacak olursa,

Nesne İsmi . Özellik İsmi = Özelliğin Yeni Değeri

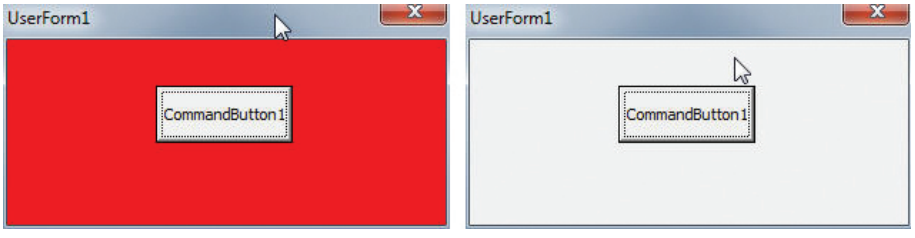
formülüne uygun şekilde yazılmalıdır. Bu örnekte rengi değiştirilmek istenen nesne ismi UserForm1, değiştirilmek istenen özellik ise Arkaplan Rengi (Backcolor) olarak belirlenebilir. Özelliğe atanacak yeni değer ise kırmızı renginin renk kodunu temsil eden vbRed olmalıdır. Yukarıdaki formüle göre Private Sub / End Sub arasına yazılması gereken kod:

User Form1 . Backcolor = vbRed

olmalıdır. Kod yazılıp çalışma zamanına geçildiğinde program kullanıma hazırdır. Form üzerindeki buton tıklandığında form arka plan renginin kırmızıya dönüştüğü görülmektedir. Resim 3.18 formun düğme tıklanmadan önceki ve tıklandıktan sonraki hâlini göstermektedir.

Şekil 3.18

Düğmenin Tıklanması ile Kodun Çalışması



Çalışma zamanında nesneye eylem uygulanınca gerçekleşmesi istenen işlemler, Private Sub / End Sub arasına yazılır.

Örneğe göre ele alınacak olursa Kod sayfasında Private Sub ile End Sub arasına yazılan kod, çalışma zamanına geçildiğinde, Private Sub yanına ismi yazılan nesneye belirtilen eylem uygulanınca devreye girecektir.

VBA pencerelerinin tümü, farklı işlemler için MS Excel otomasyonları yaratmada kullanılmaktadır. İzleyen ünitede VBA ile yapılan örnekler ele alınacaktır.

Özet

VBA genel kullanımını açıklamak.

- *İşlem Tablosu programında Geliştirici sekmesini kullanabilecek,*
- *İşlem tablosu VBA kullanımının genel mantığını tanımlayabilecek,*
- *VBA Uygulama Geliştirme mantığını kavrayacak,*
- *VBA Penceresi parçalarının çalışmalarını açıklayabilecek,*
- *VBA Kod penceresi ile çalışabileceksiniz,*

VBA, makrolar olarak da adlandırılan otomasyon grubu ögelerini kullanarak MS Excel işlem tablosu programı üzerinde daha kolay, daha detaylı ve daha karmaşık işlemleri gerçekleştirmeyi sağlayan programlama ortamıdır. MS Excel sayfası üzerinden Makro yaratılabileceği gibi, VBA pencereleri kullanılarak da nesnelere işlev kazandırılabilir. VBA ortamını kullanmak için MS Visual Basic programlama ortamı hakkında bilgiye sahip olmak gereklidir.

İşlem tablosu programında geliştirici sekmesini kullanmak.

Geliştirici sekmesi olmadan Makro yaratmak ve kullanmak mümkün değildir. VBA ortamı ve Makrolarla çalışmak için devreye alınan Geliştirici sekmesi, MS Excel seçeneklerinden sekmenin açılması ile üst menüde yerini alır. Farklı amaçlar için kullanılan Kod, Eklentiler, Denetimler, XML ve Değiştir menüleri üzerindeki butonların kullanım amaçlarını bilmek, Geliştirici sekmesini verimli kullanmak için gereklidir.

İşlem tablosu VBA kullanımının genel mantığını tanımlamak.

MS Excel üzerinden Makro kaydedilip gerektiği zaman tekrar çalıştırılabilir. Bu işlem özellikle tekrarlı işlemleri yapmayı kolaylaştıracaktır. Daha detaylı işlemleri gerçekleştirmek için ise VBA ortamından faydalanmak ve yazılan kodları uygulamaya sokmak gereklidir. Bu kodlar sayesinde MS Excel programlama ortamında da faydalanılabilir duruma gelecektir.

VBA uygulama geliştirme mantığını ifade etmek.

VBA, MS Visual Basic adı verilen bir programlama ortamını temel alarak çalışan yardımcı bir ortamdır. MS Visual Basic gibi nesneye dayalı programlama mantığı ile çalışır. Nesneye dayalı uygulama geliştirme mantığında, yazılan program kodlarının devreye girmesi için bir nesneye eylem uygulanmalıdır. VBA ortamında bir diğer dikkat edilmesi gereken nokta ise tasarım zamanı ve çalışma zamanı olarak iki farklı zamanda çalıştığını bilmektir. Tasarım zamanı programı oluşturduğumuz, nesneleri ve eylemleri belirlediğimiz, gerekli kodları yazdığımız zamandır. Tüm program geliştirildikten sonra çalışma zamanına geçilerek program test edilir ve tasarlanan işlemlerin gerçekleştirilmesi sağlanır.

VBA penceresi parçalarının çalışmalarını açıklamak.

VBA, kendi içerisinde Proje Penceresi, Özellikler Penceresi, Araç Çubuğu, Nesne Tarayıcı gibi pencereler ve Görünüm, Ekle, Çalıştır gibi menüler içerir. Bu pencere ve menüler yardımıyla geliştirme yapılır. Kod penceresi ise yazılan kodları üzerinde saklayan ve çalıştıran editör olarak görev yapmaktadır.

VBA Kod penceresi ile çalışmak.

Kod penceresi, bir nesneye bir eylem uygulandığı durumlar için gerekli program kodlarını üzerinde tutan ve VBE (Visual Basic Editor) olarak adlandırılan penceredir. Bu pencerede nesne-eylem ikilisi seçildiğinde, otomatik olarak kod satırının başlığı olan Private Sub (Alt Yordama Özel) ve kod satırının sonunu işaret eden End Sub (Alt Yordamı Sonlandır) çıkmaktadır. Yazılacak olan kod, bu iki satır arasına yazılmalıdır.

Kendimizi Sınayalım

1. VBA kısaltmasının açık yazımı aşağıdakilerden hangisidir?
a. MS Visual Basic for Accounting (Hesaplamalar için Visual Basic)
b. MS Visual Basic for Application (Uygulamalar için Visual Basic)
c. MS Visual Basic for Autocad (AutoCad Yazılımı için Visual Basic)
d. MS Visual Basic for Automatic (Otomatik için Visual Basic)
e. MS Visual Basic for Action (Eylem için Visual Basic)

2. VBA ile geliştirilen uygulamaların diğer bir adı aşağıdakilerden hangisidir?
a. MS Excel
b. Yordam
c. Makro
d. Kod
e. Sistem

3. Aşağıdakilerden hangisi "Geliştirici" sekmesinin alt menülerinden biri **değildir?**
a. Görünüm
b. Kod
c. Denetimler
d. Değiştir
e. Eklentiler

4. Verinin yapısını standartlaştırarak diğer uygulamalarda da kolayca çalışmasını sağlayan işlemler, "Geliştirici" sekmesi altında hangi menüde bulunur?
a. Değiştir
b. Kod
c. Denetimler
d. XML
e. Eklentiler

5. Aşağıdakilerden hangisi nesne-eylem ikilisine bir örnektir?
a. Buton-Tıklama
b. Makro-Geliştirici
c. Visual Basic-Uygulama
d. Kod-Yazı
e. Kullanıcı Formu - Buton

6. Aşağıdakilerden hangisi VBA zamanlarından biridir?
a. Sabit Zaman
b. MS Excel Zamanı
c. Tasarım Zamanı
d. Deneme Zamanı
e. VBA Zamanı

7. Projeye bağlı tüm nesnelerin dizin hâlinde görüntülendiği pencere aşağıdakilerden hangisidir?
a. Özellikler Penceresi
b. Nesne Tarayıcı Penceresi
c. Araç Çubuğu Penceresi
d. Kod Penceresi
e. Proje Penceresi

8. VBA pencereleri içerisinde hangi pencere ile kullanıcı formu üzerine nesneler eklenebilir?
a. Özellikler Penceresi
b. Nesne Tarayıcı Penceresi
c. Araç Çubuğu Penceresi
d. Kod Penceresi
e. Proje Penceresi

9. VBE olarak da adlandırılan pencere aşağıdakilerden hangisidir?
a. Özellikler Penceresi
b. Nesne Tarayıcı Penceresi
c. Araç Çubuğu Penceresi
d. Kod Penceresi
e. Proje Penceresi

10. Kod penceresinde otomatik olarak çıkan "Private Sub (Alt Yordama Özel)" yazısı neyi ifade eder?
a. Kod Satırının Başlangıcını
b. Kod Yazılamayacağını
c. Yazılan Kodun Hatalı Olduğunu
d. VBA Sürümünün Yanlış Olduğunu
e. Programın Sonlandığını

Kendimizi Sınayalım Yanıt Anahtarı

1. b	Yanıtınız Yanlış ise "Giriş" konusunu yeniden gözden geçiriniz.
2. c	Yanıtınız Yanlış ise "Giriş" konusunu yeniden gözden geçiriniz.
3. a	Yanıtınız Yanlış ise "Geliştirici Sekmesi" konusunu yeniden gözden geçiriniz.
4. d	Yanıtınız Yanlış ise "Geliştirici Sekmesi" konusunu yeniden gözden geçiriniz.
5. a	Yanıtınız Yanlış ise "VBA Çalışma Temelleri" konusunu yeniden gözden geçiriniz.
6. c	Yanıtınız Yanlış ise "VBA Çalışma Temelleri" konusunu yeniden gözden geçiriniz.
7. e	Yanıtınız Yanlış ise "VBA Penceresi Özellikleri" konusunu yeniden gözden geçiriniz.
8. c	Yanıtınız Yanlış ise "VBA Penceresi Özellikleri" konusunu yeniden gözden geçiriniz.
9. d	Yanıtınız Yanlış ise "VBA Penceresi Özellikleri" konusunu yeniden gözden geçiriniz.
10. a	Yanıtınız Yanlış ise "VBA Penceresi Özellikleri" konusunu yeniden gözden geçiriniz.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1

Bu işlemi gerçekleştirmek için öncelikle Geliştirici Sekmesini devreye alınız. Makro Kaydet menüsü ile kaydı başlatınız. Oluşturmak istediğiniz Makro işlemlerini tamamladıktan sonra Kaydı Durdur butonunu tıklayınız. Sayfanızı tamamen temizleyerek Makroyu çalıştırınız. Makronun doğru biçimde çalıştığından emin olunuz.

Sıra Sizde 2

Oluşturulan Makroyu siliniz. Sıra Sizde 1 bölümündeki işlemlerin aynılarını yapmadan önce bu sefer Göreli Başvuruları Kullan seçeneğinin seçili olduğundan emin olunuz. Makroyu tekrar oluşturduktan sonra, Makronun oluşturulması için kullandığınızdan farklı bir hücre adresini başlangıç hücresi olarak vererek Makroyu tekrar çalıştırınız. Makronun her başlangıç hücresi için doğru çalıştığından emin olunuz.

Sıra Sizde 3

Örneğe göre, Buton nesne, tıklanması ise eylem olarak belirtilmiştir.

Sıra Sizde 4

Başlık (Caption) özelliği Eylem (Action) sınıfına ait bir özelliktir.

Yararlanılan ve Başvurulabilecek Kaynaklar

https://support.office.com/tr-tr/article/Eklenti-ekleme-veya-kald%C4%B1rma-0af570c4-5cf3-4fa9-9b88-403625a0b460?ui=tr-TR&rs=tr-TR&ad=TR

MS Excel 2013 Uygulama Yazılımı Yardım Dosyası.

İŞLEM TABLOSU PROGRAMLAMA

4

Amaçlarımız

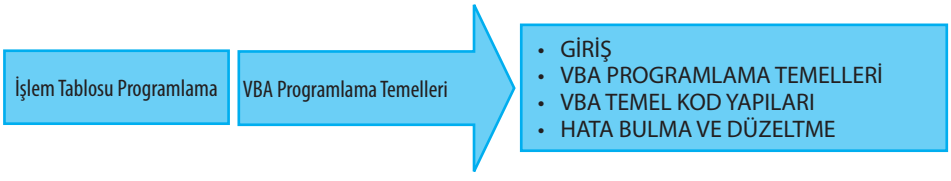
Bu üniteyi tamamladıktan sonra;

- VBA programlama temellerini ifade edebilecek,
- VBA programlamada kullanılan nesneleri tanımlayabilecek,
- VBA nesne–eylem ilişkisini açıklayabilecek,
- VBA temel kodlarının mantığını açıklayabilecek,
- VBA kod hatalarını tespit edebilecek,
- VBA kod hatalarını giderebileceksiniz.

Anahtar Kavramlar

- VBA
- VBA Nesneleri
- VBA Eylemleri
- VBA Kod Penceresi
- VBA Kod Yazımı
- VBA Temel Kod Yapıları
- Eğer Yapısı
- Select Case Yapısı
- For-Next Yapısı
- Do Loop Yapıları
- Go To Yapısı
- VBA Derleme
- VBA Hata Bulma
- VBA Hata Giderme

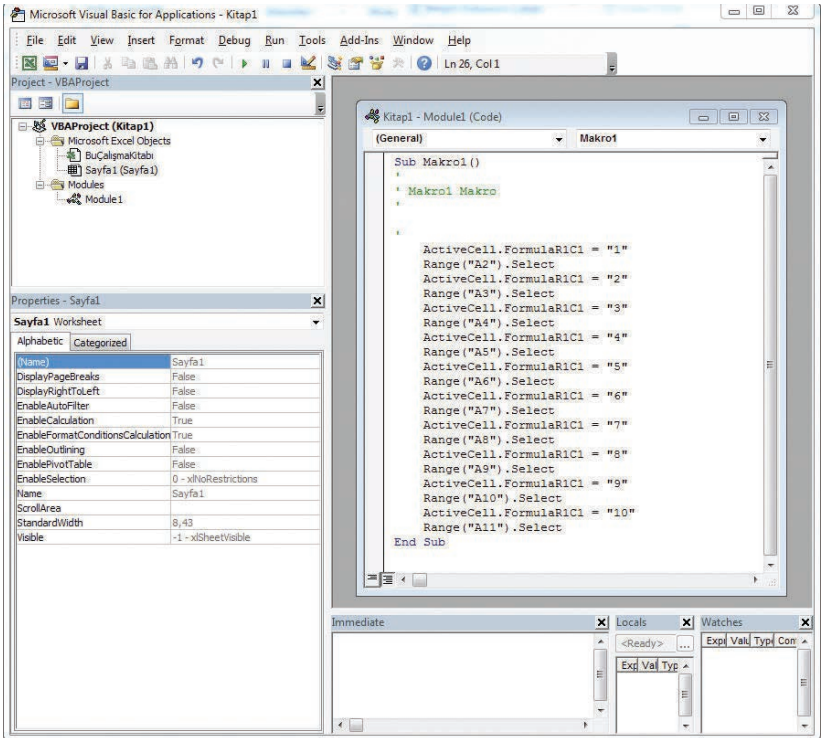
İçindekiler



VBA Programlama Temelleri

GİRİŞ

Bir önceki ünitede, Makro olarak da adlandırdığımız VBA ortamında çalışmak için gerekli ön hazırlığı yaparak, kullanılan menü, buton ve pencereler hakkında genel bilgiler verilmiştir. Hatırlanacak olursa Makro kaydetmek için Geliştirici sekmesinden Makro Kaydet butonu tıklanarak kaydetme işlemi başlatılır. Makro ile gerçekleştirilmesi istenen işlemler el ile yapılarak ve sonrasında Kaydı Durdur butonu tıklanarak Makro kaydı sonlandırılır. Daha sonra bu işlemlerin tekrarı istendiğinde Makro sekmesinden ilgili makro adı ile seçilerek baştan sona çalışması sağlanır. Kullanıcı açısından bakıldığında, yapılan işlemlerin sadece butonlar yardımıyla gerçekleştirilmesini sağlayan sistem, arka planda VBA kodları ile ifade edilmektedir. Bir önceki ünitede gerçekleştirilen A1 adresli hücreden A10 adresli hücreye kadar 1'den 9'a kadar sıralı sayıların yazılması örneğini ele alalım. Makroyu kaydettikten sonra VBA ekranı açıldığında Proje penceresi içerisinde bir Modül eklendiğini görülür. Resim 4.1'de örneğe ait Makro kodu, bir altyordam olarak görülebilir.



Resim 4.1

Yaratılan Makro1 Kodunun Görüntülenmesi

VBA kodlarının en önemli özelliği, kod penceresi içerisinde yazılan kodun satır satır okunmasıdır. Kod basitçe incelenecek olursa bu yazım konusunda fikre sahip olunabilir.

Sub kelimesi, alt yordam (Subroutine) kelimesinin kısaltması olarak kullanılmaktadır ve Makro1 adlı alt programın başlangıcını işaret eder. Başında " ' " işareti bulunan ve kod penceresinde yeşil renk ile görüntülenen yazılar, kod parçacığı hakkında bilgi vermek amacıyla kullanılan yorum kısımlarıdır. Bu yazılar kod çalışırken devreye girmez, sadece programcıyı bilgilendirmek amacıyla kullanılır.

ActiveCell.FormulaR1C1 ifadesi, çalışma kitabı içerisinde, belirlenen çalışma sayfasında aktif olan hücreye yazılacak değerin ataması işlemini gerçekleştirir. "=" atama operatörü ile çift tırnak içerisinde yazılan değer, aktif hücreye atanır.

Alt satıra inildiğinde karşılaşılan Range().Select ifadesi, aktif olan hücreyi değiştirmek/seçmek için gerçekleştirilir.

Hatırlanacak olursa Makro ile gerçekleştirilen işlem, A1 hücresine "1" sayı değerini yazmak, A2 hücresine "2", A3 hücresine "3" ve devam eden şekilde A10 hücresine kadar ilerlemektir. Makro yeniden çalıştırıldığında A1 hücresine "1" değerini yazdıktan sonra alt hücreyi aktif etmek, yeni aktif olan hücreye "2" değerini yazdıktan sonra bir alt hücreyi aktif etmek esası ile çalışmaktadır. Kod incelendiğinde de kod satırlarının işlemlerinin hücreye değer atamak ve diğer hücreyi aktif etmek şeklinde ilerlediği gözlenmektedir. Son olarak, A10 hücresine "10" değeri yazıldıktan sonra bir alt hücre olan A11 hücresi aktif edilerek kodun son satır işlemi de gerçekleştirilir.

End Sub ifadesi, Makro1 adındaki alt yordamın sonlandığını ifade etmektedir. Bir önceki ünitede de belirtildiği gibi Başlangıçta Sub, bitişte ise End Sub ifadelerinin dışına kod yazmak hataya yol açacaktır. Kod satırları, bu iki ifadenin arasında yer almalıdır.

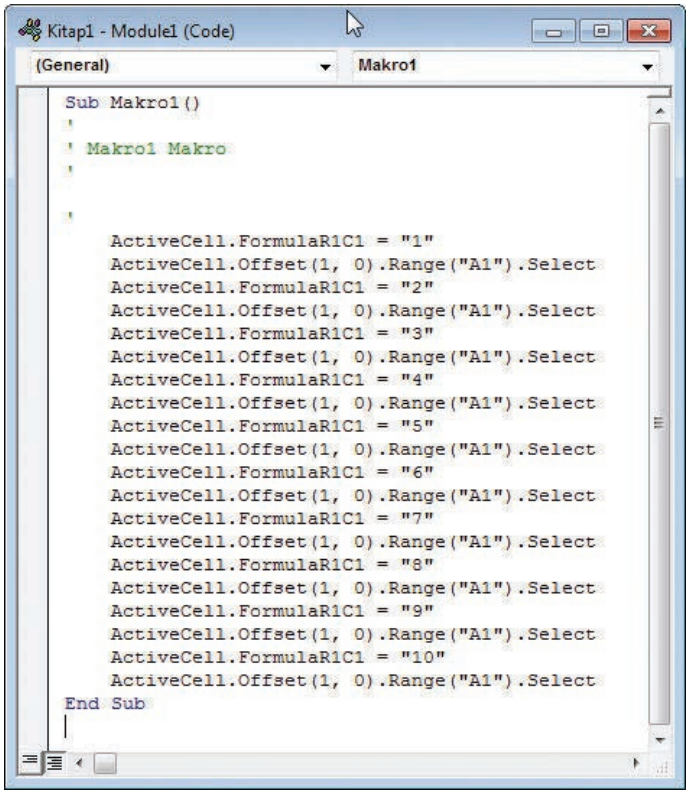
Global değişken tanımlama kodları dışında, Sub/End Sub başlangıç ve bitiş satırlarının dışına yazılacak olan kodlar hataya yol açacaktır.

Önceki ünitede konu edildiği üzere, bir makro yaratıldığında, sadece yaratıldığı hücreler için çalışacaktır. Bu işlemi genişletmek, makronun başlangıcını belirlenecek herhangi bir aktif hücreden başlamak koşulu ile tüm sayfa için geçerli yapmak, makroyu oluştururken "Göreli Başvuruları Kullan" butonunun seçili olması ile sağlanmaktadır. Resim 4.2'de yukarıda bahsedilen makro oluşumu sırasında "Göreli Başvuruları Kullan" butonu seçili olması durumunda kodlardaki farklılık göze çarpmaktadır.

Kod incelenecek olursa Makro1 alt yordamının başlangıcının, gerekli bilgi satırlarının ve bitişi sağlayan End Sub kodunun aynı şekilde muhafaza edildiği görülür. Belirlenen aktif hücrelere değerleri yazdırmayı sağlayan ActiveCell.FormulaR1C1 kod parçacığı da diğer örnekte olduğu gibi atama operatörü ile gerekli atamaları yapmak için kullanılır. Buradaki en önemli farklılık, aktif hücrelerin seçilmesi işleminde bulunmaktadır. Başlangıç noktasına değer yazıldıktan sonra bir alttaki hücreyi aktif etme işlemi, ActiveCell.Offset(1,0).Range("A1").Select kod satırı tarafından sağlanmaktadır. Offset komutundan sonra kullanılan (1, 0) değerleri, koordinat sistemindeki gibi dikey konumu 1 birim artırıp yatay konumu ise değiştirmeden bir alt hücreyi aktif eder.

Resim 4.2

Yaratılan Makro1 Kodunun Görüntülenmesi



Yukarıdaki kod satırında ActiveCell.Offset(1,0).Range("A1").Select kod satırlarının tamamı için offset değeri ActiveCell.Offset(0,1).Range("A1").Select olacak şekilde değiştirildiğinde, Makro1 için çalışma şeklini araştırınız.

SIRA SİZDE 1

Örneklerden de anlaşıldığı üzere Makro, Geliştirici sekmesindeki butonlar kullanılarak dahi yaratılmış olsa, VBA ekranında bir kod parçacığı ile ifade edilir. Yapılan işlemler MS Excel tarafından otomatik olarak VBA kodlarına dönüştürülür ve Makro'nun tekrar çalıştırılması istendiğinde yapılan işlemler, kendisine ait Sub/End Sub arasına yazılan kodların satır satır okunması ve çalıştırılması ile gerçekleştirilir.

VBA PROGRAMLAMA TEMELLERİ

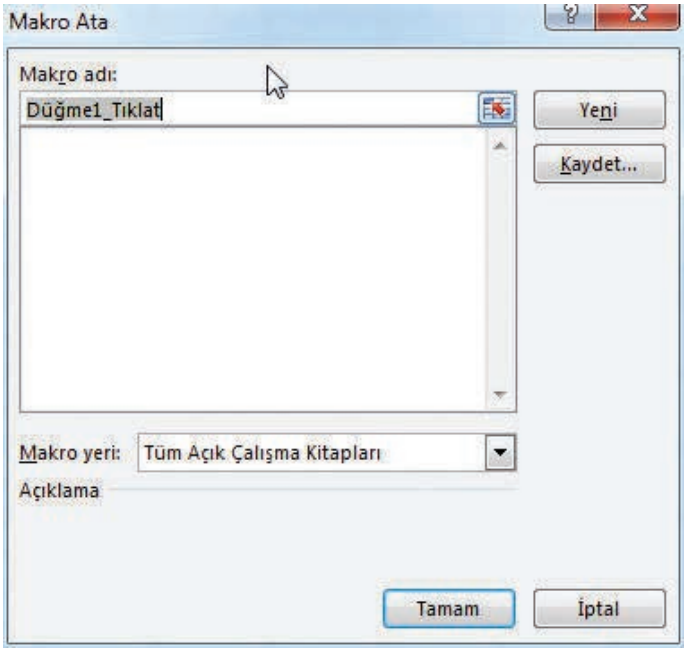
MS Excel sayfası sütun ve satırların kesişimi ile oluşan hücrelerden meydana gelmektedir. Yapılan işlemlerin neredeyse hepsi hücre içi değerlere işlemler uygulanması ile gerçekleştirilmektedir. Tekrar eden işlemleri kolaylaştırmak için oluşturulan makrolar için de asıl öge hücrelerdir.

Hücre işlemleri eğer MS Excel ana ekranında bulunan özellikler ve gerekiyorsa makrolar ile gerçekleştirilebilir işlemlerse, VBA ortamına ihtiyaç duyulmayabilir. Gerekli durumlarda ise oluşturulan kodlar, bir önceki bölümde bahsedilen şekilde görüntülenebilir ve/veya üzerlerinde değişiklik yapılabilir. Hücre işlemleri dışında nesnelere bağlı işlemler gerçekleştirmek için ise VBA kod penceresinden yararlanmak gereklidir. Geliştirici sekmesi içerisinde denetimler menüsünde bulunan Ekle butonu, sayfa üzerine eklenebilecek nesneleri barındırmaktadır. Bu buton sayesinde açılan nesnelerden herhangi birisi seçilip

sayfa üzerine eklendiğinde, bu nesneye uygun kod yazarak nesneye işlev kazandırmak, Makro ile kaydetmekten daha işlevsel olabilir. Nesnenin üzerine gerekli eylem uygulandığında istenilen işlemleri yerine getirmesi için öncelikle bir makro ile bağlantılı olması gereklidir. Bu sebeple, bazı nesneler sayfa üzerine eklendiğinde otomatik olarak Makro Ata penceresi açılarak bir makro oluşturulması istenir. Form denetimleri altındaki buton ekleme durumunda sıkça kullanılan bu işlem, Buton_Tıklat makrosu olarak karşımıza çıkar. Bir önceki üniteden hatırlanacak olursa VBA penceresinde yazılan kodların aktif hâle gelmesi yani çalışması için nesne kullanımı gerektiği durumlarda, nesneadı_eylem ikilisinin kullanılması gerekmektedir. Aynı esas ile sayfa üzerine bir buton eklendiğinde, makro adı otomatik olan butonun adı ve buton için en sık kullanılan eylem olan tıklama ikilisinden oluşacaktır. Resim 4.3'te boş MS Excel sayfası üzerine düğme eklendiğinde açılan Makro Ata penceresi görünmektedir. Makro yeri olarak, oluşturulacak makronun kullanılacağı yer belirtilmelidir. Bu yerler, Tüm Açık Çalışma Kitapları, Bu Çalışma Kitabı ya da Kitap1 olabilir. Düğme1_Tıklat Makrosu kaydedildiğinde gerekli kodlamaların yapılabileceği kod sayfası da Modül altında hazır hâle gelmiştir.

Resim 4.3

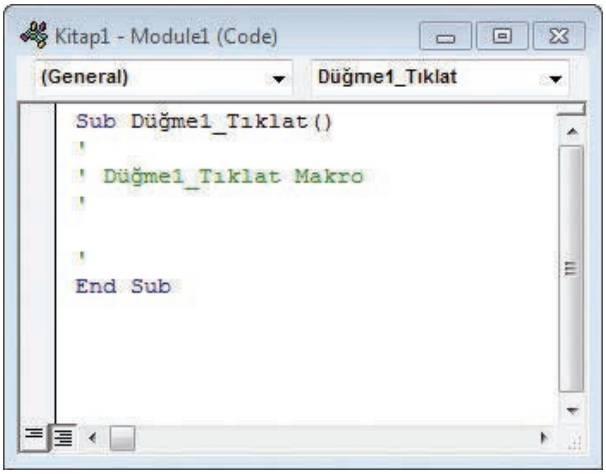
Makro Ata Penceresi Görüntüsü



Makro kaydedildikten sonra yapılması gereken VBA ekranını açmak, proje penceresinden Modülü tıklamak olacaktır. Modül tıklandığında çıkan kod penceresi Resim 4.4'te gösterilmiştir. Önceki ünitede de konu edildiği üzere, VBA programlama işlemlerinde bilinmesi gereken önemli noktalar mevcuttur.

Resim 4.4

Modül Kod Penceresi Görüntüsü

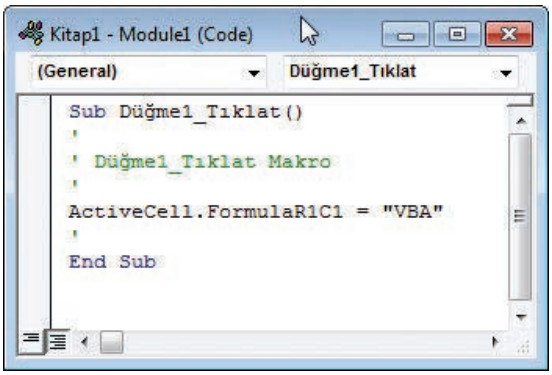


Sub Düğme1_Tıklat() / End Sub daha önce de ifade edildiği gibi kod yazılacak olan satırların başlangıç ve bitişini ifade eder. Sub kodunun yanında yazılı olanlar, nesne adı ve eyleme karşılık gelmektedir. VBA ortamında yazılan kodun çalışması için bir nesneye, bir eylem uygulanmalıdır. Ünite başından beri bu işlem, yaratılan makronun çalıştırılması ile sağlanmıştır. Burada ise Düğme1 adına sahip bir buton (nesne) ve butonlar için en sık kullanılan eylem olan Tıklatma eylemi kullanılacaktır. Sub satırının altında ve satır başında üst apostrof işareti ile simgelenen satırlar bilgi amaçlıdır ve kod penceresinde yeşil renkli olarak yazılacaktır. Bu satırlar çalıştırıcı kod içermezler fakat hem kullanıcıya hem de daha sonra kodu incelemek ya da üzerinde değişiklik yapmak isteyen programcıya yardımcı bilgiler içerirler. Amaca yönelik yazılması istenilen kod, satırlar arasına yazılır. Yazılan kod, çalışma zamanına geçildiğinde, yani MS Excel ana penceresinden buton tıklandığında, satır satır okunacak ve gerekli işlemler gerçekleşecektir. Bu sebeple kod yazımındaki sıraya dikkat edilmelidir. Basit bir örnek olarak, buton tıklandığında, aktif hücreye "VBA" metnini yazacak bir kod parçacığı oluşturulsun. Hatırlanacağı gibi ActiveCell.FormulaR1C1 kod parçacığı, aktif olan hücreye değer atamayı sağlar. Aynı kod parçacığından faydalanarak istenen işlem gerçekleştirilebilir. Bu durumda; ActiveCellFormulaR1C1 = "VBA" kod satırını Sub/End Sub arasına yazarak MS Excel ana penceresine dönülsün. Dikkat edilecek olursa makro, hâlihazırda kayıt hâlindedir. Kaydı Durdur butonunu tıklayarak makroyu kullanıma alabilirsiniz. Yapılan işlem ile fare ile aktif hâle getirilen hücre içerisine "VBA" yazılması için bir makro oluşturulmuştur. Bu makro butonun tıklanma olayına atanmıştır. Artık MS Excel ana sayfası üzerinden istenirse makro çalıştırılarak, istenirse de buton tıklanarak işlem gerçekleştirilebilir. Resim 4.5'te gerekli kod satırları ilave edilmiş şekilde modül kod penceresi görüntülenmektedir.

Makronun doğru çalıştığının test edilmesi için öncelikle boş bir MS Excel hücresi aktif hâle getirilmeli, sonra Makro çalıştırılmalıdır. Üzerinde "VBA" yazılı bir hücrenin aktif olması durumunda kullanıcı bir değişiklik göremeyecektir ve makronun çalışmadığını düşünebilir.

Resim 4.5

Düğme1_Tıklat Makrosu Kod Penceresi Görüntüsü



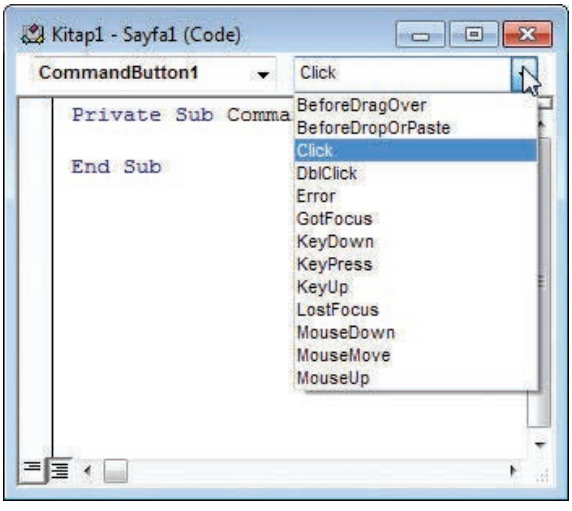
Makro atama dışında kod penceresi ile işlem yapılmak istendiğinde, VBA ortamının programlama kısmından daha fazla yararlanmak mümkündür. Bu durumda Geliştirici Sekmesi altındaki Ekle kısmından Form Nesneleri değil de ActiveX Nesneleri eklemek gereklidir. ActiveX Nesneleri ile çalışmayı öğrenmek VBA programlama becerilerini geliştirmek açısından önemlidir.

Kod penceresinde, içerisine herhangi bir kod yazılmayan, yani boş bir özel altyordam pasif hâlde olacağından, pencere içerisinde bulunması kodun çalışmasını etkilemez.

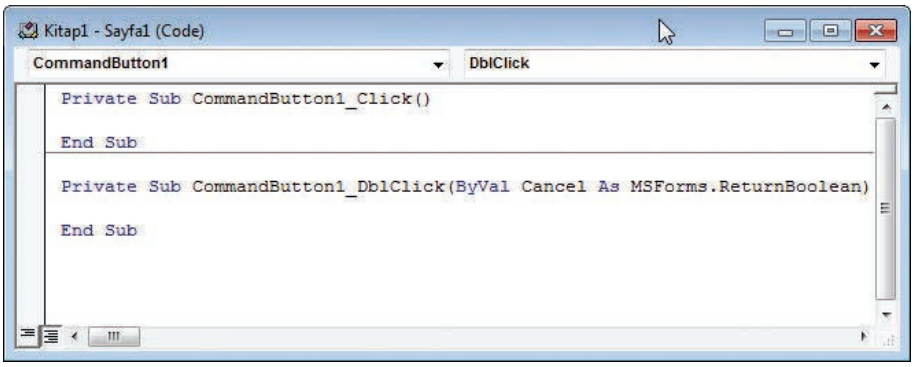
Başlangıç olarak, Ekle butonu tıklanarak ActiveX Nesneleri altından bir buton nesnesi seçilsin. Komut Butonu olarak da adlandırılan butonu seçerek MS Excel sayfası üzerinde istenilen bir yere istenilen boyutta bir nesne oluşturulsun. Nesne üzerinde isim olarak Buton yerine CommandButton1 yazısı çıkacaktır. VBA kod sayfası açıldığında, Modül altında Buton bağlantısı çıkmadığına ya da bir makro kaydedilmesi veya bir makro atanmasının istenmediğine dikkat etmek gereklidir. ActiveX nesneleri ile çalışma işlemleri, proje penceresinde bulunan Sayfa1 üzerinden yapılacaktır. Sayfa1 çift tıklandığında açılan kod penceresi, boş bir yapı olarak açılır ve bu pencerenin üst tarafında iki adet açılır menü bulunmaktadır. Genel (General) yazılı açılır menü nesneleri, Deklarasyon (Declarations) yazılı menü ise eylemleri göstermektedir. Genel sekmesi açıldığında üzerinde işlem yapılabilecek iki nesne olan CommandButton ve Worksheet nesneleri görülecektir. CommandButton seçildiğinde en sık kullanılan eylemi olan tıklatma (Click) eylemi otomatik olarak atanır ve bir özel alt yordam (Private Sub) açılır. Tıklamadan farklı bir eylem kullanılması gerektiği durumlarda eylemi değiştirmek için sağ üst kısımda bulunan açılır menüden faydalanmak gereklidir. Resim 4.6'da farklı eylemler görünmektedir.

Resim 4.6

Farklı Eylemler Sağ Kısımdaki Açılır Menü ile Seçilebilir.



Farklı bir eylem seçildiğinde, ilk eylemle birlikte oluşturulan Private Sub/End Sub tanımlarından farklı bir özel altyordam grubu oluşturulur. İçerisine kod yazılmayan özel altyordam grubu, kodun çalıştırılması esnasında kendiliğinden silinecektir. Resim 4.7'de tıklatma eylemi yerine çift tıklatma (DblClick) eyleminin kod grubunun aktif hâle getirilmiş şekli görünmektedir. Eylem çeşidine göre bazı gerekli değişken ve kullanım değerleri de nesne-eylem ikilisinin yanında tanımlanmaktadır. Bu tanımlar, o eylemin kullanımı sırasında atanması, sonlandırılması ya da değiştirilmesi gereken değerler topluluğu da olabilir.



Resim 4.7

Çift Tıklatma Eylemi İçin Özel Altyordam Grubunun Aktifleşmesi

VBA TEMEL KOD YAPILARI

Kod yazma ile ilgili genel bilgiler, ünitenin ilk bölümünde sunulmuştur. Kod yazımında kullanılan bazı kalıplar, amaçlanan programları gerçekleştirmede yardımcı olmaktadır. Bu bölümde sıkça karşılaşılan kod yapılarının çalışma mantıkları üzerinde durulacaktır. İstenilen işlemi gerçekleştirmek için kod yazarken kullanılan yardımcı yapıların çalışma mantıkları birçok programlama dilinde aynı olmasına rağmen, yazım kuralları ve yazım şekilleri programlama dillerine göre farklılık göstermektedir. Bilindiği gibi VBA ortamı, MS Visual Basic programlama dili kullanılarak geliştirilmiştir. Bu sebeple ünitenin bu bölümünde anlatılan yapılar, MS Visual Basic diline özgü yazım stilleridir.

Eğer Yapısı (If–Then–Else–End If)

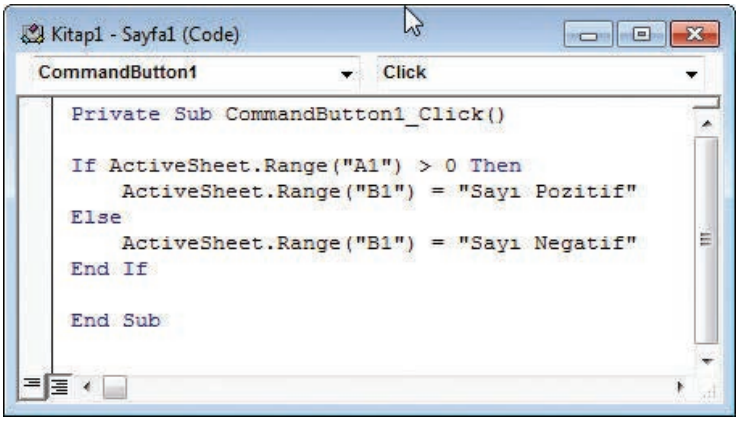
Eğer kalıpları, programlama ortamında sıkça faydalanılan karar verme kalıplarıdır. Bir koşul ve o koşulun sağlanması (doğru) ya da sağlanmaması (yanlış) durumlarına göre iki farklı işlem dizisinin gerçeklenmesi mantığı ile çalışır. Koşulun doğru olması durumunda yapılacaklar ve koşulun yanlış olması durumunda yapılacaklar olmak üzere iki farklı işlem seti tanımlanabilir. Genel yazım şekli aşağıda gösterilmiştir.

```
If <Koşul> Then
…… (Koşul sağlanıyorken yapılacaklar)
Else
…… (Koşul sağlanmazken yapılacaklar)
End If
```

If ile Then kelimeleri arasında bir koşul bulunur. Bu koşulun sağlanması yani doğru olması durumunda Then ile Else arasına yazılan kod parçaları, sağlanmaması yani yanlış olması durumunda ise Else ile End If arasına yazılan kod parçaları çalıştırılır. Bir örnek için Resim 4.8'deki kodu inceleyiniz.

Resim 4.8

Eğer Kod Örneği



Kod incelenirse If ve Then arasında yazan koşul dikkate alınmalıdır. (ActiveSheet. Range("A1") > 0) ifadesi ile üzerinde çalışılan sayfadaki A1 adresli hücre içerisinde bulunan değerin 0'dan büyük olma ya da olmama durumu test edilir.

A1 hücresine yazılan değer 0'dan büyük bir değer olursa, kod çalıştırıldığında Then ile Else arasında bulunan kod satırı çalıştırılacaktır. Bu kod satırı (ActiveSheet.Range("B1") = "Sayı Pozitif") aynı sayfa üzerindeki B1 adresli hücreye "Sayı Pozitif" ifadesini yazacaktır.

Eğer A1 hücresine yazılan değer 0'dan küçük veya eşit bir değer ise bu sefer Else ile End If arasındaki kod çalışacak (ActiveSheet.Range("B1") = "Sayı Negatif") ve yine aynı sayfa üzerindeki B1 adresli hücreye "Sayı Negatif" ifadesi yazılacaktır.

SIRA SİZDE 2

Örnekte, A1 adresli hücrenin içerisinde "0" değeri yazılı iken buton tıklandığında sonucun ne olacağını araştırınız.

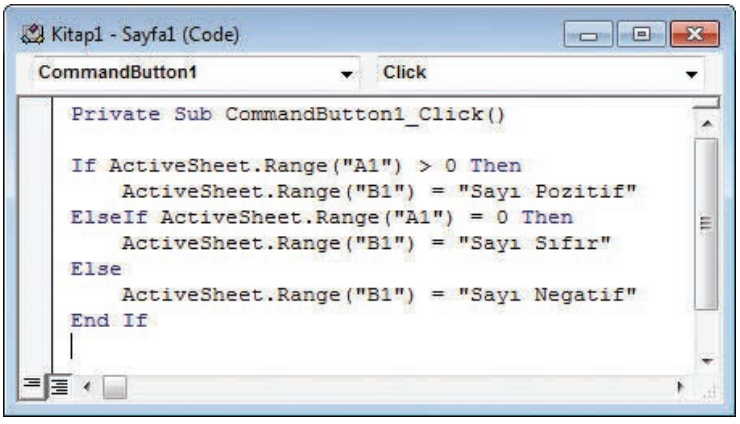
Bilgisayarlar, yazılan kodları mantıksal hatalara bakmaksızın uygulamak esasına göre tasarlanmıştır. Mantıksal olarak hatalı çalışmaları, programcıların tasarımında hata olduğunun işaretidir.

A1 hücre değerinde "0" yazması durumunda ise koşul sağlanmıyor olduğu için Sayı Negatif yazılacaktır. Sonuç matematiksel açıdan yanlıştır. Sıfır sayısı negatif bir sayı değildir. Bu hatanın sebebi programın tasarlanması aşamasında yapılan mantıksal yanlışlıktır. Dikkat edilecek olursa programın çalışması esnasında bir hata çıkmamıştır. Bunun sebebi hatanın, program kodlarının yazımında olmamasıdır. Tüm bilgisayar programlarında olduğu gibi VBA ortamında da mantıksal hataları fark etmek çoğu zaman programcı dikkati ile mümkün olacaktır.

Yukarıda söz edilen hatadan kurtulmak için, program kodlarına ilave yapmak gereklidir. Aşağıdaki kod değişikliği ile artık "0" değeri için de doğru sonuçlar alınacaktır.

Resim 4.9

Düzeltilmiş Eğer Kod Örneği



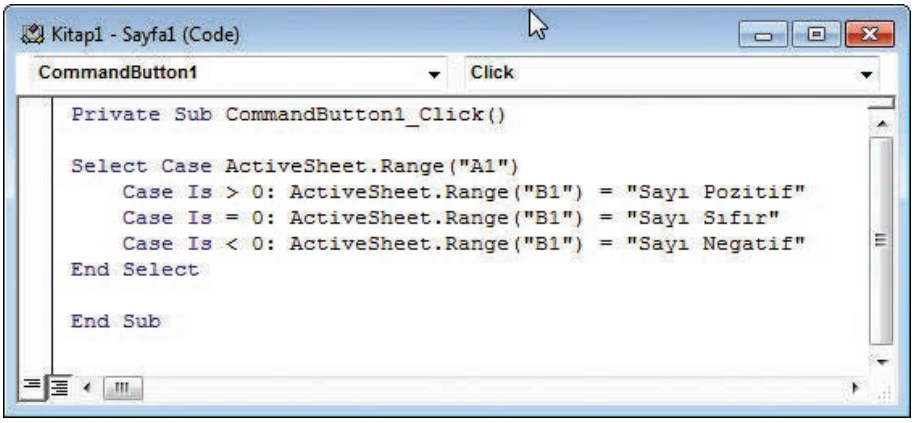
Resim 4.9'da görüntülenen koda göre, Eğer kalıpları iç içe de kullanılabilir. Else ifadesine bitişik olarak oluşturulan yeni bir If yapısı ile sayısız If yapısı iç içe kullanılabilir. Bu durumda ilk sağlanan koşul için Then ile Else arasındaki işlemler yapılır, sağlanmıyorsa Else kısmına atlanarak diğer If yapısının koşulu araştırılır. Koşul sağlandığı durumda altında yazılan kodlar uygulanır, sağlanmadığı durumda ise bir alttaki Else kısmına atlanır ve kodun çalışması bu şekilde devam eder.

Select Case Yapısı

Bir değişkenin değişik durumlarının koşul olarak kullanıldığı If kalıplarında, koşulların sayısının arttığı durumlarda kullanım kolaylığı sağlayabilecek Select Case yapısı kullanılabilir. Case olarak ilgili değişkenin alacağı farklı değerler belirlenir ve bu farklı durumlarda yapılacaklar, yapı içerisinde tanımlanır. If yapısına göre anlaşılması daha kolay olan yapı, If yapısı kadar sık kullanıma sahip değildir. Resim 4.10'da yukarıdaki örneğin Select Case yapısı ile gerçekleştirilmiş şekli bulunmaktadır.

Resim 4.10

Select Case Yapısı Kod Örneği



Select Case yapısını kullanmak için, öncelikle Case seçiminde kullanılacak koşulun belirlenmesi gerektiğini görebilirsiniz. Bu koşul, A1 adresli hücrede yazan değerdir. Bu tanımlamadan sonra artık A1 hücresinin değeri Case olarak nitelendirilecek ve 0'dan büyük, 0'a eşit ve 0'dan küçük olma durumları test edilecektir. Her bir Case için yapılması gerekenleri Case Is ile başlayan kod satırının devamına yazmak gereklidir. Select Case yapısı da tıpkı eğer yapısında olduğu gibi End ifadesi ve yapının adı ile son bulmalıdır. End Select ifadesi, yapının son bulduğuna işaret etmektedir.

Case için kullanılan seçeneklerden herhangi birisine uymayan durumlarda kullanılmak üzere Case Else ifadesi gereklidir. Test edilmesi gereken tüm durumlar yazıldıktan sonra koşulun değerinin hiçbir koşula uymadığı durumlarda yapılması gerekenler Case Else kısmına yazılmalıdır. Bu kod parçacığı bazı durumlarda unutulan ya da atlanan koşul durumlarının yakalanması ve programın devam etmesi için kullanılmaktadır.

Sonuç çıktısı aynı olan kodun her iki şekilde de yazılması mümkündür. Hangisinin kullanılacağı programcının tercihidir.

For–Next Döngüsel Yapısı

Kodlama aşamasında bazı durumlarda aynı işlemi tekrar etmek için, üzerine kod yazılan butona defalarca tıklamak gerekebilir. Böyle durumlardan kurtulmak ve kodun tekrar edilmesi işleminin programcıdan/kullanıcıdan alınarak bilgisayar tarafından tekrarlanması için döngüsel yapılara başvurulur. Döngüsel yapılarda dikkat edilmesi gereken nok-

ta, tekrar edilecek işlemlerin birbirinin aynısı olmasıdır. Farklı işlemleri tek bir döngüsel yapıda kullanmak mümkün değildir. For–Next döngüsel yapısı, programlama dillerinin hemen hepsinde bulunan For döngüsel yapısının MS Visual Basic programlama dili üzerindeki karşılığıdır.

For-Next döngüsel yapısını kullanmak için değişkenlere ihtiyaç duyulur. Değişken kavramı bir sonraki ünitede detaylı olarak anlatılmakla birlikte, işlem yapmak istenen değerleri bilgisayar içinde belirli konumlarda saklayarak, bir isim ile onları çağırmak ve gerektiğinde işleme soktuktan sonra, gerektiğinde tekrar kullanılmak üzere yine belirli konumlarda saklamak amacı ile kullanılır. For–Next döngüsel yapısı da bir değişkenin başlangıç değerinden bitiş değerine kadar değerleri sıra ile alması ve tüm değerler için tanımlanan işlemleri tekrar yapması şeklinde tanımlanabilir.

For *Değişken_Adı = Başlangıç_Değeri* To *Bitiş_Değeri*
Yapılması İstenilen İşlemler
Next *Değişken_Adı*

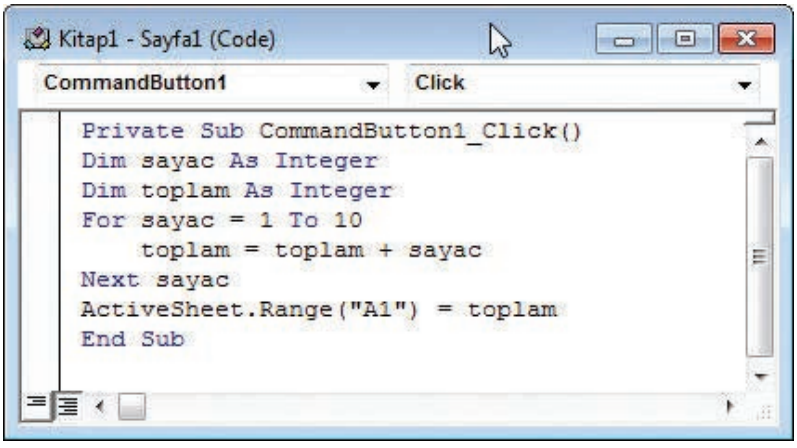
Yukarıdaki şekilde tanımlanan For–Next döngüsel yapısında hiçbir farklı belirtme olmadığı durumlarda, değişken değeri başlangıç değerinden başlayarak, bitiş değerine kadar 1'er artarak işleme sokulacaktır. Farklı şekilde ilerleme isteniyorsa (2'şer, 3'er ya da 5'er vb.) bitiş değeri yanına Step ifadesi ve artırım değerini girmek yeterli olacaktır.

For x=1 To 5 Step 2 şeklinde başlangıç yapılan bir kod için x değişken değeri önce 1, sonra 3 ve son olarak 5 değerini alacak, yani başlangıç değerine her bir adım da artırım değeri ilave edilerek sonraki değerler hesaplanacaktır.

Resim 4.11'de For-Next döngüsel kalıbı ile gerçekleştirilen toplama örneği görülmektedir. Bu örnekte ilk önce 1'den başlayarak 10'a kadar değer almak için *sayac* adında bir tamsayı değişkeni ve üzerinde toplam değerlerini saklamak için *toplam* adında ikinci bir tamsayı değişkeni tanımlanmıştır. Sayac değişkeninin başlangıç değeri olan 1 sayısı ve son alacağı değer olan 10 sayısı ise For yanına yazılan kod ile belirtilmiştir. İlk okumada sayaç değeri 1 olacak, Next Sayac ifadesine kadar yazılı bulunan kod gereği toplama koduna girecek ve toplam değerini değiştirecek, daha sonra yukarıdaki For satırına tekrar gelerek 2 değerini alacak ve yeniden toplam işlemine girecektir. Bu şekilde 10 kez tekrar ettikten sonra son alacağı değer olan 10 değeri ile son kez toplama koduna girecek ve Next Sayac ifadesi artık kodun altında yazan A1 adresli hücreye sonuç değeri olan toplam değişkeninin güncel değerini yazdırma işlemini gerçekleştirecektir.

Resim 4.11

For-Next Döngüsel Yapısı Kod Örneği



1'den başlayarak her tıklamada toplam değişkeninin üzerine değer eklenmesi istenirse, 10 tıklama sonunda 1+2+…+10 işleminin sonucuna ulaşılır. Döngüsel yapı kullanılarak bu işlem tek tıklamaya düşürülmüş, böylece hem zamandan tasarruf edilmiş hem de kullanım zorluğundan ortadan kaldırılmıştır. Sonuç değeri olan 55 değeri bu örnek sonucunda A1 adresli hücrede görülecektir.

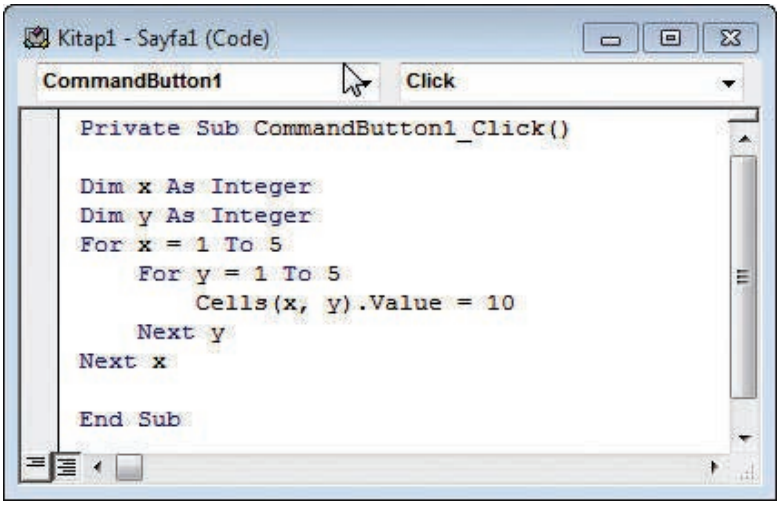
Aynı kod için başlangıç satırı For sayac=1 To 9 Step 2 olacak şekilde değiştirildiğinde sonucun ne olacağını kodlayarak araştırınız.

SIRA SİZDE 3

For döngülerinin özel durumlarda durdurulması ve döngü dışına çıkılması gerektiğinde, Exit For kod satırı gerekli yere konmalıdır. Exit For ifadesi kullanımı konusunda dikkatli olunması gereken bir ifadedir. Yanlış yerde kullanımı, sebepsiz yere kodun döngüden çıkmasına ve işlemlerin yarıda kalmasına sebep olabilir.

For döngüleri bazı durumlarda sadece belirli yinelemeler için atlanabilir ve diğer yinelemeler için işlemlerin yapılması istenebilir. Bu durumda Continue For kalıbı kullanılmaktadır.

For döngüleri de gerekli durumlarda iç içe kullanılabilir döngülerdir. Basit bir örnek vermek gerekirse üzerinde işlem yapılan MS Excel sayfasında A1 hücresinden başlayarak 5x5 boyutlarında, tüm değerleri 10 olan bir matris oluşturulsun. Bu işlemi el ile yapmak yerine bir buton aracılığıyla programlanması istenirse Resim 4.12'de de görülen biçimde iç içe 2 For döngüsü yazılarak işlem sağlanır. Öncelikle For yapısında kullanmak üzere x ve y adı verilen iki değişken oluşturulmuş, daha sonra x ve y için iç içe iki ayrı For döngüsü tanımlanmıştır. Yazma işlemi yapılacak hücrelerin boyut değerleri, döngülerdeki kontrol değişkenlerin değerlerinden alınabilir. Bunu sağlamak için Cells(x, y) ifadesinden faydalanılmıştır. Her iki For döngüsünde de kullanılan x ve y değişkenlerinin alacağı başlangıç değerleri 1, bitiş değerleri ise 5 olarak belirlenmiştir. Value (Değer) komutu ile tüm hücre değerleri 10 olarak belirlenmiştir. Buton tıklandığında A, B, C, D ve E sütunlarının ilk 5 hücrelerine değer olarak 10 sayısı atanmıştır.



Resim 4.12

İç İçe Kullanılan For-Next Döngüsel Yapısı Kod Örneği

Do While Döngüsel Yapısı

For-Next döngüsel yapısının kullanımında, döngünün başlangıç ve bitiş değerleri programcı tarafından belirlenir. Bu sayede, döngünün tekrarlanma sayısı belirlenmiş olur. Örnek vermek gerekirse For x=1 To 5 şeklinde kodlanan bir döngünün, x değişkeninin değerleri sırasıyla 1, 2, 3, 4 ve 5 olacak şekilde 5 kez çalışacağı söylenebilir. Kodlama aşamasında bazı durumlarda, belirlenen koşul sağlandığı sürece, döngü değişkenine bağlı kalmaksızın döngünün tekrarlanması istenebilir. Bu tür durumlarda For-Next döngüsel yapısı kullanmak yerine Do-Loop olarak adlandırılan döngüsel yapılar içerisinden Do While döngüsü kolaylık sağlayacaktır.

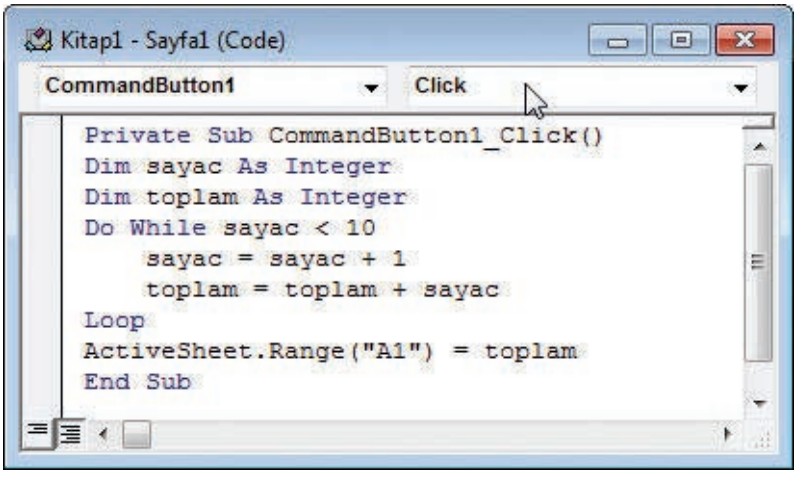
```
Do While <Koşul>
Yapılması İstenilen İşlemler
Loop
```

Genel yazım şekli incelenecek olursa Do While Yapısı, koşul gerçekleştiği sürece döngü içerisinde kalan ve işlemleri tekrar tekrar gerçekleştiren bir döngüsel yapıdır. Döngünün sonlanması, koşulun sağlanmamasına bağlıdır.

Her ne kadar döngü sayısının belli olmadığı durumlarda kullanıldığı belirtilse de belirli örneklerde Do While yapısı, For Next yapısı yerine, aynı işlemleri gerçekleştirmek için kullanılabilir. Resim 4.13'te, yukarıda For-Next döngüsel yapısı için kullanılan 1'den 10'a kadar sayıların toplam sonucunu bulan örneğin, Do While döngüsel yapısı ile kodlanmış hâli görülmektedir.

Resim 4.13

Do While Döngüsel Yapısı Kod Örneği



Kod incelenecek olursa, sayaç ve toplam adı verilen iki değişken oluşturulduktan sonra, Do While sayac < 10 ifadesi yer almaktadır. Do While yapısında koşul sağlandığı sürece döngü içinde kalınacağı hatırlanırsa bu kod satırı, "sayaç değişkeninin değeri 10 sayısından küçük olduğu sürece döngü içerisinde kal" şeklinde yorumlanabilir. Döngüsel kod içerisinde her okumada sayaç değişken değerinin 1 arttırılması ve bu yeni değerin toplam değerine eklenmesi işlemleri yer almaktadır. Loop ifadesi Do yapıları için döngüye geri dönmeyi sağlar. Yani okunan satırı tekrar Do While ile başlayan satıra yönlendirir. Döngünün okunması ya da döngü dışına çıkılması ile ilgili kararı Do While ile başlayan ve koşulu içeren satır verir.

Koşul sağlanmadığı durumda, yani başka bir ifade ile sayaç değişkeninin değerinin 10'dan büyük olduğu durumda ise döngü dışına çıkılır ve toplam değişkeninin değeri olan 55, A1 adresli hücreye yazdırılır.

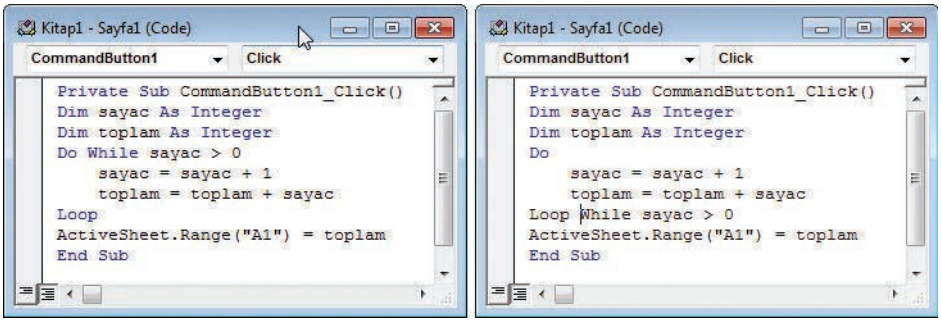
Do yapılarının yazınsal olarak farklı bir şekli daha mevcuttur. Bu şekil, While <Koşul> ifadesinin Do yanına değil de Loop yanına yazılması ile oluşturulur. Aşağıda genel yazım şekli görülen yazım şeklinde Do sadece döngüye girme emri olarak döngü içerisindeki tüm işlemlerin en az bir kere gerçeklenmesini sağlar. Bir kere gerçeklendikten sonra döngü işleminin yapılması ya da yapılmaması ise Loop While <Koşul> ifadesinin sorumluluğundadır.

```
Do
Yapılması İstenilen İşlemler
Loop While <Koşul>
```

While <Koşul> kod parçacığının yazım yerinin Do ya da Loop ifadesinden sonra olması çoğu örnek için fark oluşturmasa da bazı sınır değer problemlerinde, değişiklik yapılması, programın hata vermesine ve çalışmasının kesilmesine yol açabilir. Resim 4.14'te bu şekilde bir örnek kod verilmiştir.

Resim 4.14

While <Koşul> Yazım Yeri Değişikliği Örneği



Örneğe dikkat edilecek olursa koşul, sayaç değişken değerinin "0" sayısından büyük olması durumuna göre belirlenmiştir. MS Visual Basic ortamında oluşturulan sayısal tipteki bir değişkenin ilk değeri her zaman "0" dır. Bu sebeple Do While sayac>0 satırındaki koşul sağlanmadığı için döngü içerisine girilmeyecektir. Toplam değişkeninin değerinde de herhangi bir değişiklik olmayacağı için başlangıç değeri olan "0", a1 Adresli hücreye yazılacaktır. Program hata vermeyecektir.

İkinci kısımda ise Do komutundan sonra sayaç değişkeninin değerinin "1" sayı artırımı ve toplam değişkenine bu yeni sayaç değişken değerinin eklenmesi işlemleri yapıldıktan sonra koşula bağlı olarak döngü sağlanacaktır. Dikkat edilecek olursa Loop While sayac > 0 satırına gelindiğinde sayaç değişkeninin değeri "1" olacaktır. Bu sebeple döngüye devam edilir. Bu örnekteki sorun, döngünün ne zaman biteceği ile ilgili bir kısıt olmamasıdır. Bu sebeple sonsuz döngü olarak adlandırılan döngüden hiç çıkmama durumu oluşur ve değişken değeri kendine ait limiti aştığı an program hata ile çalışmasını durdurur.

Görüldüğü üzere değişken tanımları ve işlemler dizisi aynı olmasına rağmen, birinci kısımda çalışan ve "0" sonucunu veren program, While <Koşul> kısmının yer değiştirmesinden sonra hataya yol açmıştır. Hatanın önlenmesi için gerekli durumlarda döngüden çıkılması ya da ilgili tekrarın atlanması için sırasıyla Exit Do ve Continue Do deyimlerinden faydalanılabilir.

While <Koşul> kod parçacığının, Do ya da Loop ifadesinin yanına yazılması, sınır değer problemlerinde dikkat edilmesi gereken önemli bir değişikliktir.

DİKKAT

Do while döngüsel yapısını, benzer şekilde While-Wend yapısı ile de görmek mümkündür. Bu yapıda da koşul sağlandığı sürece döngü içerisinde kalınarak, koşulun sağlanmadığı durumlarda döngü sonlandırılmaktadır. Döngü oluşturma terimi olarak Loop ifadesi yerine Wend ifadesi kullanılmaktadır.

Do Until Döngüsel Yapısı

Do While döngüsel yapısı ile yazım şekli olarak bire bir aynı olan Do Until döngüsel yapısı, çalışma mantığı açısından bakıldığında Do While ile taban tabana zıt işlemler yapar. Hatırlanacak olursa Do While döngüsel yapısında döngü içerisindeki kodların okunması ve çalıştırılması için koşulun sağlanıyor olması gerekir. Koşul sağlanmadığı durumlarda ise döngü içerisindeki kodlar çalıştırılmaz. Do Until yapısında ise döngü içerisindeki kodların çalıştırılması için koşulun sağlanamıyor olması gereklidir. Döngü içerisindeki kodlar, koşul sağlanana kadar tekrar tekrar çalıştırılır, koşul sağlandığı anda ise döngü durdurulur. Do Until döngüsel yapısının genel yazım şekli aşağıda verilmiştir.

Do Until <Koşul>
Yapılması İstenilen İşlemler
Loop

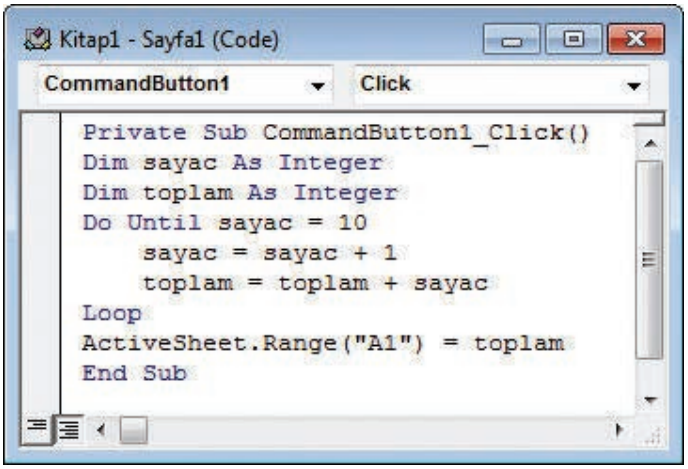
Bu yapıda da Until <Koşul> kısmı, diğer Do döngüsünde olduğu gibi Loop ifadesi yanında da yer alabilir.

Do
Yapılması İstenilen İşlemler
Loop Until <Koşul>

Karşılaştırma yapmanın kolay olması açısından daha önceki yapılarda da gerçeklenen 1'den 10'a kadar sayıların toplamını bulan kod parçacığını, Do Until yapısı ile Resim 4.15'te görebilirsiniz.

Resim 4.15

Do Until Döngüsel Yapısı Kod Örneği



Do Until sayac = 10 ifadesi, sayaç değişkeninin başlangıç değerinin "0" olduğu hatırlanırsa daha anlamlı olacaktır. Dikkat edileceği üzere koşul sağlanmamakta ve döngü içerisine girilerek ve kodlar çalıştırılarak gerekli işlemler yapılmaktadır. Bu döngü, koşul satırından da anlaşılacağı gibi sayaç değişken değeri 10 olana kadar tekrar edecek, daha sonra ise döngü dışına çıkarak toplam değişken değeri olan 55 sayısı A1 adresli hücreye yazılacaktır.

Until <Koşul> ifadesi Do ifadesinin ya da Loop ifadesinin yanına yazıldığında kod okunurken yapılacak değişiklikler, Do While döngüsel yapısı ile aynıdır. Döngünün içerisinde gerekli durumlarda sürdürülmesi ya da sonlandırılması için ise sırasıyla Continue Do ve Exit Do ifadeleri kullanılmaktadır.

Aşağıda yazılı kod parçacığı çalıştırıldığında elde edilecek program çıktısını belirleyiniz.

SIRA SİZDE 4

```
Private Sub CommandButton1_Click()
Dim sayac As Integer
Dim toplam As Integer
Do
    sayac = sayac + 1
    toplam = toplam + sayac
Loop Until sayac = 0
ActiveSheet.Range("A1") = toplam
End Sub
```

Go To Yapısı

MS Visual Basic programlama dilinin en basit yapılarından birisi olan Go To yapısının kullanılabilmesi için öncelikle kod yazımında satır numaraları vererek kodu oluşturma gereksinimi vardır. Go To ifadesinden sonra gidilmesi istenilen satır numarası belirtilerek, o satıra ulaşmak amaçlanmaktadır. Bu sayede kod sayfasını satır satır yukarıdan aşağıya doğru okumak mecburiyeti ortadan kalkarak, istenilen sıçramalara imkân tanınmış olur. Ancak bu sıçramalar programın okunurluğunu azaltacak ve ne yapılmak istendiğini kavramayı zorlaştıracaktır. Bu nedenle Go To yapısı yerine gerekli durumlarda yukarıda bahsedilen If, Select Case, For-Next, Do While, Do Until gibi yapıları kullanmak, yazılan kodun daha kontrol edilebilir olmasını sağlamaktadır.

HATA BULMA VE DÜZELTME

VBA ortamında yazılan kodların, **mantıksal hatalar** nedeniyle istenilen sonuçları vermeyebileceği daha önce açıklanmıştır. Algoritma kurulması aşamasında yapılan hataların sonucu olarak programda mantıksal hataların bulunması, program kodlarının doğru çalışmasına rağmen istenen işlemleri gerçekleştirmemesine yol açar. Mantıksal hataların düzeltilmesi, kodun tekrar gözden geçirilmesi ile sağlanabileceği gibi her şeyin başa dönülerek tekrar yapılandırılması ya da yeniden tasarlanmasına kadar giden zorlu bir süreçtir.

Mantıksal hatalar, hiç uyarı vermeksizin yanlış sonuçlar çıkartabileceği gibi bazı durumlarda derleyici tarafından da algılanarak düzeltilmesi sağlanabilen hata türlerindendir.

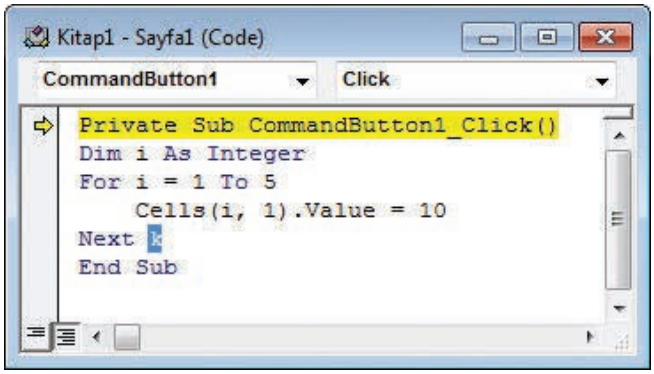
Mantıksal hatalar dışında, programın çalışmasını engelleyen, özellikle yazım hataları ya da programın çalışırken kesilmesine yol açan çalışma zamanı hatalarından da bahsetmek mümkündür. Yazım ve çalışma zamanı hatalarının düzeltilmesi için kod üzerinde kodlamayı gerçekleştirdiğimiz VBA Editörü'nden yardım alınabilir.

Bilindiği üzere, yazılan tüm kodların işlemcinin anlayabileceği şekle dönüştürülmeden çalıştırılması mümkün değildir. VBA ortamı tarafından Editör üzerine yazılan kodlar da otomatik olarak işlemci tarafından anlaşılır hâle dönüştürülmektedir. Bu işleme kod derleme, bu işlemi gerçekleştiren ortama da derleyici adı verilmektedir. VBA derleyicisi, yazılan kodları derlerken yazım hatası bulunan kısımları hata mesajı ile bildirir. Genellikle derleme işlemi yarıda kesilir. Bir mesaj yardımıyla hatanın türü programcıya bildirilir. Eğer programcı isterse Debug (Hata Ayıklama) butonunu tıklatarak, hatalı olduğu düşünülen satır ya da alt yordam, sarı renk ile çizilerek düzeltilmesi için tasarım zamanına geri dönülmesi sağlanır. Bu işaretlemenin amacı, kod sayfasında hata bulunan yerin işaret edilerek programcıya kolaylık sağlanmasıdır. Programcı gerekli satırları inceleyerek hatayı giderir ve derleyiciyi tekrar çalıştırır. Bazı durumlarda ise hata, yazım hatası değil de eksik tanımlanmış ya da tanımlanmamış bir değişken ya da nesne olabilir. Hata esnasında

derleyici tarafından iletilen mesaj, hata tipini de öğrenmeye yardımcı olur. Eksik nesne ya da tanımlanmamış değişken gibi hatalarda ise gerekli tanımlamaları kod sayfasına ilave etmek hatanın giderilmesini sağlayacaktır. Resim 4.15'te i değişkeni için tanımlanan bir For döngüsel yapısının *k* adlı başka bir değişken için çevrilmesi sebebiyle oluşmuş bir hata ve işaretlenmesi görülmektedir.

Resim 4.16

Hata Yakalama Anında Kod Sayfası

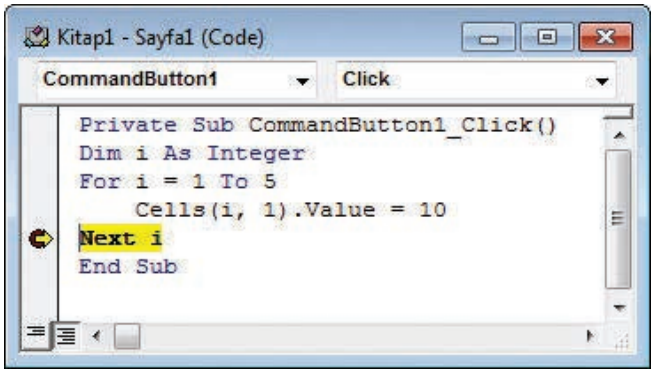


Bazı durumlarda ise değişken değerleri ya da işlemler istenilen sonuçları vermez. Bir anlamda mantıksal hataların yakalanması için de kullanılabilecek iki farklı yöntem ile program kodlarının gözden geçirilmesi gerekebilir. Bu yöntemlerden ilki, F8 tuşuna basılarak program kodlarının birer birer okunarak ilerletilmesi ve bu sırada gerekli ise VBA pencerelerinden birisi olan İzleme Penceresi ile değerlerin incelenmesidir. Bu sayede programın çalışma mantığı daha iyi kavranır ve belirlenen hatalar giderilebilir.

İkinci bir yöntem ise program kodlarının arasında belirli noktalara Kesme Noktası (Break Point) yerleştirmektir. Program çalışırken kodun Kesme Noktasına kadar olan kısmı çalıştırıldıktan sonra, tasarım zamanına geri dönülerek program kesilir ve o andaki program verileri incelenir. Kesme noktası farklı yerlerde oluşturularak program hatalarının yakalanması ve düzeltilmesi sağlanır. Resim 4.17'de kesme noktası oluşturulmuş bir kod çalıştırıldıktan sonra kesme noktasına gelince durması ve kod penceresi görüntülenmektedir.

Resim 4.17

Kesme Noktası (Break Point) Kullanım Örneği



Hataların düzeltilmesi sayesinde, oluşturulan program istenilen sonuçları vermek üzere tamamlanmıştır. Nesne-Eylem ikilisi kullanılarak defalarca çalıştırılabilir.

VBA kullanımı, MS Excel için birçok kolaylığı da beraberinde getirmektedir. MS Visual Basic ortamında tanımlanan kodlamalarla işlemler baştan sona kadar istenilen şekilde otomatik biçimde gerçekleştirilebilir.

Özet

VBA programlama temellerini ifade etmek.
MS Excel kolay kullanımı olarak da adlandırabilecek VBA kullanımının temeli Makrolara dayanmaktadır. Makro oluşturma işlemi Kayıt butonlarıyla dahi yapılsa, bir VBA koduna karşılık gelmektedir. Makro işlemlerini MS Excel ana sayfası üzerinde gerçekleştirerek kaydetmek ya da VBA ortamında kodlayarak tanımlamak programcının tercihidir. Yoğun işlem gerektiren programlarda VBA ortamından faydalanmak kolaylık sağlayacaktır.

VBA programlamada kullanılan nesneleri tanımlamak.
Makro kullanımında kaydedilen makro ismi ile çağırılabileceği gibi Geliştirici sekmesi Ekle butonu yardımıyla da farklı nesnelerin MS Excel ana sayfası üzerine eklenmesi mümkündür. Form nesneleri olarak adlandırılan ilk grup nesneler eklendiğinde, kullanımları bir makro ile bağlanabilir ve gerekiyorsa VBA ekranında Proje penceresinde Modül kod sayfasına tanımlamalar yapılabilir. ActiveX nesneleri ise direkt sayfa üzerinden kod yazımı ile geçerlilik kazanan, herhangi bir makroya bağlanması gerekmeyen ikinci tür nesne grubudur.

VBA nesne–eylem ilişkisini açıklamak.
VBA ortamında yazılan kodların çalıştırılması için Makro olarak bile atansa bir nesneye, bir eylem uygulanılması gerekmektedir. Bir düğme nesnesi ile çalışmak için onun en sık kullanılan eylemi olan Tıklama işlemi, Makro ile çağırılsa bile arka planda kullanılması gereken yordamdır.

VBA temel kodlarının mantığını açıklamak.
Kod yazımı esnasında istenilen işlemlerin kolay gerçekleştirilmesi için belirli yapılardan faydalanılır. Neredeyse tüm programlama dillerinde bulunan bu yapıların yazım şekilleri programlama diline göre değişmektedir. Eğer yapıları ile koşulun doğru ya da yanlış çıktığı durumlarda farklı eylemlerin yapılması, For-Next yapısı ile birden çok kez tekrar edilmesi gereken işlemlerin döngüsel bir yapı içerisinde tekrarlanması, koşula bağlı olarak döngü sayısının tahmin edilmediği durumlarda işlemlerin kolayca gerçekleştirilmesi için Do Loop yapılarından faydalanılması, VBA ortamında kullanılan yapılara örnektir. Bu yapılar sayesinde programcıların ya da son kullanıcıların kod geliştirmesi de kolaylaşmıştır.

VBA kod hatalarını tespit etmek.
VBA Editör aracılığıyla yazılan kodlar, derleyici aracılığıyla işlemcinin anlayabileceği şekle dönüştürülür ve bu işleme derleme ismi verilir. Derleme işlemi esnasında gerek mantıksal, gerek yazınsal, gerekse de çalışma zamanında ortaya çıkan hata türleri mevcuttur. Derleyici, bu hata türlerine karşı uyarı vererek düzeltilmesini sağlar.

VBA kod hatalarını gidermek.
Derleyici tarafından hata uyarısı bir mesaj ile birlikte verilir. Bu mesaja göre hatanın türü hakkında bilgi sahibi olan programcı, gerekli düzeltmeleri gerçekleştirerek programın doğru ve istenilen şekilde sonuç vermesini sağlar. Kod hatalarını düzeltmek için Hata Ayıklama (Debug) moduna geçilerek kodun F8 tuşu ile adım adım çalıştırılması ya da Kesme Noktaları (Break Point) belirleyerek programın parçalı halde çalıştırılmasıyla hatalar tespit edilebilir.

Kendimizi Sınayalım

1. Kod penceresinde yazılan kod aşağıdaki hangi ikili arasında bulunmalıdır?

a. Sub / End Sub
b. Start / Stop
c. Begin / Finish
d. If / End If
e. Başla / Bitir

2. "Sub" kelimesi aşağıdakilerden hangisinin kısaltmasıdır?

a. Kod
b. Döngü
c. Altyordam
d. Proje
e. Makro

3. Makro ile gerçekleştirilen işlemin, başlangıç hücresi verilen tüm hücrelerde çalışacak şekilde genellenmesi için aşağıdaki özelliklerden hangisi kayıt anında seçili olmalıdır?

a. VBA Kod Sayfası
b. Göreli Başvuruları Kullan
c. ActiveX Nesneleri Kütüphanesi
d. Özellikler Penceresi
e. Eklentiler Sekmesi

4. MS Excel penceresinde aktif hücreye "VBA" yazılmasını sağlayan kod aşağıdakilerden hangisidir?

a. Private Sub CommandButton1_Click()
b. ActiveSheet.Range("A1") = "VBA"
c. ActiveCell. Range("B1") = "VBA"
d. ActiveCell.FormulaR1C1="VBA"
e. ActiveSheet.İnterior.Color=VbRed

5. Aşağıdakilerden hangisi bir eylem türü **değildir?**

a. CommandButton
b. Click
c. GotFocus
d. DblClick
e. Error

6. Bir koşulun doğru ya da yanlış olmasına göre iki farklı işlem yaptırmak için aşağıdaki deyimlerden hangisi kullanılır?

a. Go To
b. While - Wend
c. If – Then – Else – End If
d. Do While
e. Do Until

7. Aşağıdakilerden hangisi "For-Next" döngüsel yapısında artırım değerini değiştirmek için kullanılır?

a. Next İfadesi
b. Başlangıç Değeri
c. Değişken İsmi
d. Bitiş Değeri
e. Step ifadesi

8. "Do–Loop" yapılarında döngünün sonlandırılması için hangi deyim kullanılır?

a. While Do
b. End Do
c. Exit Do
d. Until Do
e. Loop Do

9.
```
Private Sub CommandButton1_Click()
Dim sayac As Integer
Dim toplam As Integer
Do Until sayac = 6
    sayac = sayac + 2
    toplam = toplam + sayac
Loop
ActiveSheet.Range("A1") = toplam
End Sub
```

Yukarıdaki kod çalıştırıldığı zaman A1 hücresinde yazan değer aşağıdakilerden hangisi olur.

a. 6
b. 10
c. 12
d. 21
e. 55

10. Programın hatalarını tespit edebilmek amacıyla programın adım adım ilerletilmesi için aşağıdaki fonksiyon tuşlarından hangisi kullanılır?

a. F2
b. F4
c. F6
d. F8
e. F10

Kendimizi Sınayalım Yanıt Anahtarı

1. a	Yanıtınız yanlış ise "Giriş" konusunu yeniden gözden geçiriniz.
2. c	Yanıtınız yanlış ise "Giriş" konusunu yeniden gözden geçiriniz.
3. b	Yanıtınız yanlış ise "Giriş" konusunu yeniden gözden geçiriniz.
4. d	Yanıtınız yanlış ise "VBA Programlama Temelleri" konusunu yeniden gözden geçiriniz.
5. a	Yanıtınız yanlış ise "VBA Programlama Temelleri" konusunu yeniden gözden geçiriniz.
6. c	Yanıtınız yanlış ise "VBA Temel Kod Yapıları" konusunu yeniden gözden geçiriniz.
7. e	Yanıtınız yanlış ise "VBA Temel Kod Yapıları" konusunu yeniden gözden geçiriniz.
8. c	Yanıtınız yanlış ise "VBA Temel Kod Yapıları" konusunu yeniden gözden geçiriniz.
9. c	Yanıtınız yanlış ise "VBA Temel Kod Yapıları" konusunu yeniden gözden geçiriniz.
10. d	Yanıtınız yanlış ise "Hata Bulma ve Düzeltme" konusunu yeniden gözden geçiriniz.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1

ActiveCell.Offset(0,1).Range("A1").Select şeklindeki değişiklik ile başlangıç hücresine "1" değeri atandıktan sonra aktif hücre seçimi bir alt hücreye değil, sağdaki hücreye geçecektir. "2" değeri bu hücreye atandıktan sonra aktif hücre seçimi yine bir sağdaki hücreye geçecektir. Değişiklikler sayesinde, Makronun ilk amacı olan başlangıç hücresinden itibaren alta doğru 10 hücreye sayı yazmak yerine, başlangıç değerinden sonra sağa doğru 10 hücreye 1'den 10'a kadar sayılar yazılacaktır.

Sıra Sizde 2

Örneğin doğru çözümü için koşula göz atmak gereklidir. Koşula göre A1 hücresindeki değer 0'dan büyük olduğu durumlarda Sayı Pozitif yazısı yazılacaktır. Hücre Değeri olan 0, koşul için sınama değeri olan 0'dan büyük olmadığı için koşul sağlanmıyormuş gibi düşünülecek ve gerçekte yanlış olmasına rağmen, Sayı Negatif yazılacaktır.

Sıra Sizde 3

For sayac=1 To 9 Step 2 koduna göre sayaç değişken değeri 1 olarak başlayacak 2'şer artarak sırası ile 3, 5, 7, 9 değerlerini alacaktır. Bu durumda toplam değeri 1+3+5+7+9 =25 olarak değişecektir ve A1 adresli hücrede 25 yazacaktır:

Sıra Sizde 4

Do kalıbından sonra sayaç değişken değerinin 1 tamsayı arttırılması ve bu yeni değerin toplam değişken değerine eklenmesi işlemleri bir koşula bağlı olmadığı için en az bir kere yapılacaktır. Sayac değişken değeri 1 sayısı ile Loop Until sayac=0 satırı okunacak ve koşulda yazdığı üzere sayac değişken değeri 0 olmadığı için döngü devam edecektir. Sonsuz döngü durumundaki program, yaratılan değişken değeri limiti aştığı an hata vererek kesilecektir. Program, çıktı yerine hata verecektir.

Yararlanılan ve Başvurulabilecek Kaynaklar

https://msdn.microsoft.com/
https://support.microsoft.com/

İŞLEM TABLOSU PROGRAMLAMA

5

Amaçlarımız

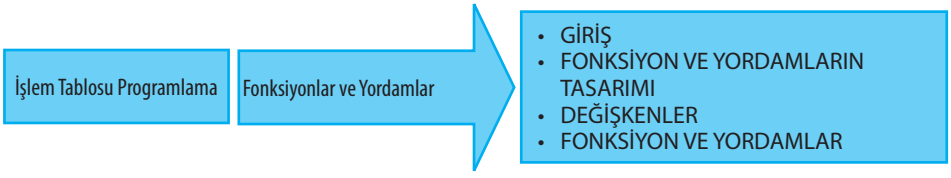
Bu üniteyi tamamladıktan sonra;

- VBA tasarım mantığını açıklayabilecek,
- Değişkenleri ve verileri tanımlayabilecek,
- Fonksiyon ve yordamları açıklayabileceksiniz.

Anahtar Kavramlar

- VBA
- Yazılım Geliştirme Yaşam Döngüsü
- Veri Tipleri
- Operatör ve Deyimler
- Zaman Fonksiyonları, Karakter Fonksiyonları, Matematik Fonksiyonları

İçindekiler



Fonksiyonlar ve Yordamlar

GİRİŞ

VBA yazılımları ile günlük sorunları çözecek sistem tasarımları yapılmaktadır. Öncelikle sorunun tanımlanması ardından da çözümün sistematik bir biçimde üretilmesi önemlidir. Bu ünite kapsamında yazılım tasarımında kullanılan Yazılım Geliştirme Yaşam Döngüsü anlatılacaktır.

VBA'da tasarlanan çözümler yordam ve fonksiyonları içerir. Fonksiyon ve yordamlar önceki ünitelerde anlatılan makro yapısının gelişmiş halleridir. VBA kullanılanılar özel bir tür dil ile fonksiyon ve yordamlar oluşturulur. Bu dil ile kullanıcıdan alınan veriler işlenerek istenen çıktılara dönüştürülürler. Fonksiyonlar VBA içerisinde tanımlanmış fabrikalar gibi çalışır. Klavyede basılan tuşların veya tıkladığımız butonun VBA'da hazırlanmış fonksiyon ve yordamlara aktarılabilmesi için değişken ismi verilen özel paketler kullanılmaktadır. Kullanılan verilerin özelliklerine göre değişken türleri de değişiklik gösterir. Örneğin harf verilerinin saklandığı değişken paketleri metin değişkenleri olarak tanımlanırken, rakam verilerinin saklandığı değişken paketleri ise sayısal değişken olarak tanımlanmaktadır.

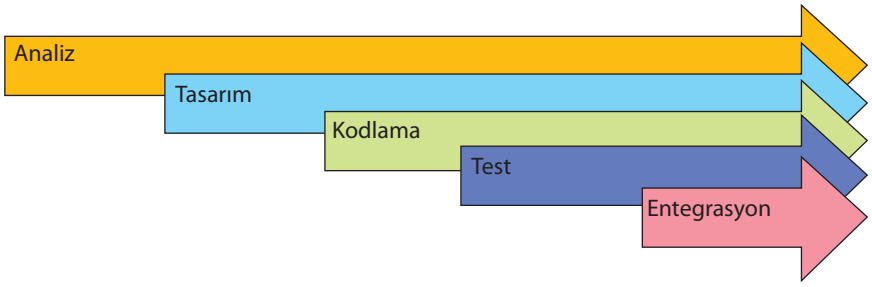
Değişken paketlerinin işlenmesinde VBA içerisinde kayıtlı bulunan komutlardan yararlanılır. Fonksiyon ve yordamlar bu komutları bir araya getirerek daha önceki ünitelerde anlatılmış olan operatörleri mantıksal işlemlerle birleştirmeye yarar. Örneğin aritmetik bir işlem sonucunda bir sayı sıfıra bölünmeye çalışılıyorsa mantıksal sınamalar ile hatanın önüne geçilebilir. Bunların yanında VBA'da tanımlanmış zaman fonksiyonları, karakter fonksiyonları, matematik fonksiyonları kullanılarak değişken paketleri içerisindeki veriler kullanıcıların ihtiyacını karşılayacak biçimde işlenir.

FONKSİYON VE YORDAMLARIN TASARIMI

Günümüzde yöneticilere ve çalışanlara işlerini yaparken yardımcı olan araçlardan biri bilgi sistemleridir. Yaygın olması ve kolay kullanımı nedeniyle Excel VBA tercih edilen bir bilgi sistemi yazılımıdır. Verimliliği arttırmak için kullanılan Excel VBA çalışanların gün içerisinde yaptıkları işin süreçlerine göre tasarlanır ve planlanır. Genel olarak bakıldığında bilgi sistemlerinin tasarımında 5 aşamalı bir model olan Yazılım Geliştirme Yaşam Döngüsü (Software Development Life Cycle - SDLC) kullanılmaktadır.

Şekil 5.1

Yazılım Geliştirme Yaşam Döngüsü Modeli



SDLC modelinde yazılım geliştiricileri öncelikle sorunu analiz ederler. İkinci aşamada sorunu çözecek bir tasarım geliştirilir. SDLC'nin üçüncü aşamasında tasarıma uygun kodlama yapılır. Dördüncü aşamada yazılım güvenlik, kullanışlılık vb. gibi farklı testlerden geçirilir. Son aşamada ise sorun çözecek yazılım kullanıma sunularak entegrasyonu tamamlanır.

Yazılım tasarımı programlamanın ilk aşamasıdır. VBA kodlama işlemine başlamadan önce mutlaka ön çalışma yapmak ve programın nasıl işleyeceğini ortaya koymak gerekir. Tasarım aşamasında nasıl bir VBA sistemi oluşturulacağına karar verilir. Sistem, en temel anlamıyla belirli bir amaca ulaşmak için girdileri süreçleyerek çıktılara dönüştüren yapıdır. Sistemler diğer sistemler ile etkileşimli olabilir. Bir sistemin çıktısı bir diğer sistem tarafından kullanılıyorsa buna açık sistem denir. Kapalı sistemlerde çıktılar üretildikten sonra sistem sonlanır.

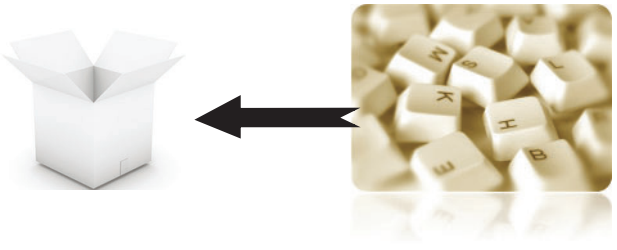
Yazılım geliştiricilerinin temel işlevi sorunları çözecek sistem tasarımı yapmaktır. Bu nedenle hangi verilerin kullanıcıdan alınacağına, bu verilerin hangi süreçler ile elde edileceğine ve çıktıların neler olacağına karar vermek gerekir. Yazılım geliştiricinin tasarlayacağı sistemin büyüklüğü, karmaşıklığı ve diğer sistemler ile ilişkisi sonsuzdur. VBA programlamasında girdiler Excel programı hücrelerinden alınır, fonksiyon ve yordamlamalarla süreçlenir, hücreler veya mesaj kutuları ile çıktı üretir.

DEĞİŞKENLER

VBA programında girdilerin yordam ve fonksiyonlarda kullanılması için değişkenler kullanılır. Değişkenler yazılım içerisindeki paketlere benzer. Değişken paketlerinin içine veriler konulur ve yazılımda gerektiği zaman bu paketler açılarak verilerin kullanılması sağlanır.

Şekil 5.2

Yazılım Geliştirme Yaşam Döngüsü Modeli



Excel'de VBA uygulamalarında, genellikle veriler Excel hücrelerindeki rakam veya harflerden oluşur. Bu verilerin VBA yazılımına girebilmesi için öncelikle değişken paketleri içerisine yerleştirilmesi gerekir. Değişken paketi tanımlandıktan sonra paketin içerisine veri ataması için = (eşit) simgesi kullanılır.

Değişkenlerin Tanımlanması

VBA içerisinde değişken paketlerinin kullanılmadan önce tanımlanması gereklidir. Değişken paketinin içerisinde hangi tür verilerin gireceği ve bu değişken paketinin adının ne olacağı bilgisi VBA yazılımına tanıtılmalıdır. Değişken paketlerinin tanımlama işlemi değişken paketinin taşıyacağı verilerin özelliklerine göre yapılır. Değişken paketi adı tanımlanırken bir harf ile başlanmalıdır. Değişken paketlerinin adı nokta, boşluk, Türkçe karakter içeremez ve 255 karakterden fazla olmaz. Karışıklıkların önlenmesi için değişken paketlerinin adı Excel VBA kodlarında bulunan deyim, fonksiyon ve yordamlardan oluşamaz. Bunun yanında kapsama alanı içinde değişken paketi adı tek olmalıdır.

Değişken paketleri tanımlanmaz veya yanlış tanımlanırsa "Variable not defined" hata mesajı ile karşılaşılır.

Değişken paketlerinin VBA yazılımında verileri taşıyabileceği yerlere yaşam alanı denilir. Değişken paketleri sadece yaşam alanları içerisinde kullanılır. Yaşam alanları değişkenlerin tanımlanma biçimine göre genel (global) seviye, modül (module) seviye ve yordam (procedure) seviye olabilir. Örneğin, *Dim* ve *Static* deyimleri ile bir yordamın içinde tanımlanan değişken paketi sadece o yordamın içinde veri taşıyabilir. Başka bir yordam içerisinde kullanılabilmesi için *Private* veya *Dim* anahtarları kullanılarak modül seviyesinde tanımlanması gereklidir. *Public* anahtarı kullanılarak genel seviyede tanımlanan bir değişken paketi diğer modüllerden de çağrılabilir.

Veri Tipleri

Günlük hayatta herhangi bir malzemenin taşınması için paketler kullanılır. Bu malzemenin sıvı, katı ve gaz türünde olmasına göre paketlerin yapısı değişir. Örneğin, pasta taşımak için tasarlanan kâğıttan yapılmış bir pakette su taşınamaz. Benzer şekilde VBA yazılımında da verilerin özelliğine uygun değişken paketlerinin kullanılması gereklidir. Veri tipleri Tablo 5.1'de açıklandığı gibi sayısal, metin, sabit, mantıksal değişken ve tarih tipinde olabilir.

Tablo 5.1
Yazılım Geliştirme Yaşam Döngüsü Modeli

VERİ TİPLERİ

Sayısal Veri Tipleri
Sayısal tanımlanmış değişken paketleri; toplama, çıkartma, bölme ve çarpma gibi çeşitli aritmetik işlemler için rakamlardan oluşan verileri Excel hücrelerinden alıp VBA yazılımındaki fonksiyon ve yordamlarda yeri geldiği zaman kullanmak üzere taşırlar. Sayısal veri tipleri için tanımlanmış veri paketleri ve kullanım örnekleri aşağıdaki gibidir:

İsim	İsim	Kullanım Örneği
Byte	Tamsayı	Dim degisken_paketi As Byte
Integer	Tamsayı	Dim degisken_paketi As Integer
Long	Tamsayı	Dim degisken_paketi As Long
Single	Ondalık sayı	Dim degisken_paketi As Single
Double	Ondalık sayı	Dim degisken_paketi As Double
Currency	Ondalık sayı	Dim degisken_paketi As Currency

Metin Veri Tipi
Metin veri tipi karakterlerden (genellikle harflerden) oluşur. Metinlerin yordam ve fonksiyonlarda kullanılması için metin veri tipini taşıyacak veri paketleri tanımlanmalıdır. Örneğin, VBA uygulamalarında yazılan bir kullanıcı adı ve şifrenin kontrolünde genellikle metin veri tipini içine alabilecek değişken paketi kullanılır. Örnek: Dim sifre As String. Sabit uzunluktaki bir karakter dizisinde veri paketi olarak tanımlanabilir. Örnek: Dim sifre As String*6. Tanımlanan sifre değişken paketi en fazla 6 karakter uzunluğunda veriyi saklayabilir. Daha uzun bir metin bu değişken paketine atanırsa metnin fazlalık kısmı kaybolur.

Sabit Veri Tipi
VBA yazılımı içerisinde yer alacak ve değişmemesi gereken veriler olabilir. Bu durumda değişken paketleri tanımlanırken sabit veri tipi olduğu belirtilir. Genellikle sabit verileri tutacak değişken paketleri tamamen büyük harflerle ve kelimeler arası alt çizgilerle ayıracak şekilde tanımlanır. Örnek: Const FAIZ As Double=0.17

Tarih Veri Tipi
Tarih veri tipi Excel hücrelerindeki takvim ve/veya saat şeklinde yazılmış tarih verileri için kullanılır. Tarih verileri değişken paketleri içerisinde ondalık sayı olarak saklanır. Örnek: Dim doğum_tarihi As Date

Mantıksal Veri Tipi
VBA yazılımında sadece doğru/yanlış (true/false) verilerini alabilen değişken paketlerinin tanımlanması için Boolean değişken paketinden faydalanılır. Örnek: Dim karar As Boolean, karar=True

Esnek Veri Tipi
VBA yazılımında tipini ön göremediğimiz verileri değişken paketlerine yerleştirmek istediğimizde esnek bir yapı kullanmamız gerekmektedir. İki şekilde tanımlanabilir: Örnek: Dim esnek_veri_paketi veya Dim esnek_veri_paketi As Variant. Bu değişken paketi sayıları, metinleri, tarihleri, dizileri ve diğer veri tiplerini tutabilir.

FONKSİYON VE YORDAMLAR

Fonksiyon ve yordamlar VBA yazılımında tasarlanan sistemin temel parçalarıdır. Yordamlar VBA yazılımında tasarlanan sistemi tanımlamakta kullanılan özel bir tür dildir. Tasarlanan sistemin kavramları ve kavramların çalışma biçimi bu dil sayesinde tanımlanabilir. Yordamlar tıpkı insanların kullandığı dil gibi özel bir yapıdaki söz dizimi içinde belirtilir. Bu söz dizimleri çoğunlukla gerçek anlamlarına benzese de insanların kullandığı dilden farklıdır. Yordamlar kalıplaşmış sözcük topluluğu ya da cümleler olarak tanımlanabilir. Başka bir ifade ile yordamlar yazılım dilindeki kelime ve cümle yapıları kurallarıdır. Değişken paketleri ve veri tipleri kalıplaşmış sözcük topluluğu örneklerinden biridir.

VBA yordamları 4 ana grupta toplanabilir:

- Alt Yordam (Sub Procedure): VBA yazılımında kullanılan kodların bir arada bulunduğu yordamdır. Alt yordamın temel amacı tekrarlanan kodların azaltılmasıdır. Alt yordamlar günlük hayatta lojistik işletmelerine benzetilebilir. Çünkü yordam tek yönlü çalışır. Tıpkı bir gönderi paketini lojistik işletmesine teslim edilmesinde geri dönüş olmaması gibi yazılım içerisinde yordam çağırıldığında geri dönüşü yoktur. Amacı olan işlemleri gerçekleştirir ve kod kaldığı yerden devam eder.
- Fonksiyon Yordamı: Alt yordamla aynıdır. Temel farkı geriye değer döndürmesidir.
- Özellik Yordamı: Kullanıcı tarafından tanımlanan sınıflardaki özelliklere ulaşmada kullanılır.
- Olay Yordamı: Nesneler ile ilişkili yordamlardır. Herhangi bir olay gerçekleştiğinde otomatik çalışan yordamlardır. Bir düğme veya liste kutusu gibi kontrol nesnelerine bağlı olarak çalışır.

VBA yazılımında fonksiyonların temel işlevi girdileri süreçlemek ve fonksiyonun amacına yönelik çıktılar üretmektir. Örneğin, günlük hayatta kullanılan bir kahve makinesinin temel fonksiyonu su, kahve ve elektrik girdilerini alarak içilebilir sıcaklıkta kahve çıktısı üretmektir. Benzer şekilde VBA yazılımında fonksiyon çalıştırıldığında, çalıştırılan fonksiyon amacına uygun şekilde veriler ile işlem yaparak sonuç üretir. Sık kullanılan fonksiyonların bir bölümü, Excel VBA tabanında hazır olarak bulunur ve komutlar şeklindedir. Örneğin, bu günün tarihini öğrenmek için VBA yazılımında *Date* komutu bulunur. *Date* komutu bilgisayarın **sistem** saatinden tarihi alır ve yazılımın içerisinde tanımlanmış veri paketi içine taşır. VBA fonksiyonları kullanıcı ihtiyaçlarına bağlı olarak tek bir komuttan oluşabileceği gibi bir işletmenin tüm muhasebe kayıtlarını tutabilecek kadar karmaşık olabilir. Açık sistem yaklaşımı ile tanımlanmış bir VBA yazılımında fonksiyonların ürettiği çıktı diğer fonksiyonlar tarafından kullanılır.

Sistemler diğer sistemler ile etkileşimli olabilir. Bir sistemin çıktısı bir diğer sistem tarafından kullanılıyorsa buna açık sistem denir. Kapalı sistemlerde çıktılar üretildikten sonra sistem sonlanır.

Günlük hayattan bir örnek vermek gerekirse; canlılarda sindirim sisteminin çıktıları boşaltım sisteminin girdileridir. Benzer şekilde bir VBA yazılımında da fonksiyonlar kullanılarak alt sistem grupları oluşturulabilir. Bu bir tür fabrika sistemine benzer. Cep telefonu üretimi yapan bir işletme düşünülecek olursa, cep telefonu üretim işletmesi telefonun plastik aksamlarını farklı bir fabrikada, ekranı farklı bir fabrikada, ana kartı farklı bir fabrikada üretebilir. Bu parçalar montaj fabrikasına gelerek birleştirilir. Öncesinde cep telefon üretim işletmesi diğer fabrikalara ürünlerin özelliklerini gönderir, üretimi yaptırır ve üretimin çıktılarını denetleyerek kullanır. Benzer şekilde VBA yazılımında tekrarlanan komut setleri alt fonksiyonlara yollanılarak çıktılar alınabilir.

Excel VBA örneklerinin gerçekleştirilebilmesi için 3. Ünite "VBA PENCERESİ İLE ÇALIŞMA" ünitesindeki "Geliştirici Sekmesi" eklenmelidir.

Operatörler

Kullanıcıların ihtiyaçlarına göre VBA yazılımında aritmetik işlemler sıklıkla kullanılmaktadır. Toplama çıkarma, çarpma, bölme vb. işlemler aritmetik işlemler olarak tanımlanır. Aritmetik işlemlerin yapılabilmesi için VBA yazılımına özgü simgeler vardır. Örneğin, toplama işlemi simgesi, matematik dilinde gösterilen simgeyle aynı olan + simgesidir. Fakat çarpma işlemi için VBA yazılımında kullanılan simge * işaretidir. Excel VBA yazılımını hazırlarken kullanılabilecek operatörlerden önemlileri Tablo 5.2'de özetlendiği gibidir.

Tablo 5.2
Operatörler

OPERATÖRLER	
	Aritmetik Toplama +, Çıkarma -, Çarpma *, Bölme /, Tamsayı bölme \, Bölmede kalan MOD, Üs alma ^, Karekök alma SQR()
	Karşılaştırmalar Eşit =, Eşit değil <>, Büyük >, Büyük Eşit >=, Küçük <, Küçük Eşit <=
	Mantıksal Ve AND, Veya OR, Değil NOT

Operatörlerin işlevlerini anlamak için basit bir aritmetik işlem yapan VBA tasarımı yapılabilir. Aritmetik bir işlem VBA yazılımında toplam 7 aşamada gerçekleştirilir. Bir toplama işleminde:

Birinci aşamada, Şekil 5.3'te görüldüğü gibi Excel sayfası hazırlanır. Excel sayfasında B1 hücresine 4 rakamı, B2 hücresine 7 rakamı yazılır.

İkinci aşamada geliştirici sekmesinde yer alan ekle tuşuna basılarak komut düğmesi eklenir.

Üçüncü aşamada tasarım modu düğmesine basılır ve command tuşu üzerine çift tıklanır veya klavyede alt ve F11 tuşlarına beraber tıklanır.

Dördüncü aşamada Şekil 5.4'te görüldüğü gibi VBA yazılımına hücrelerdeki verilerin girdi olarak alınabilmesi için değişken paketlerinin türleri tanımlanır.

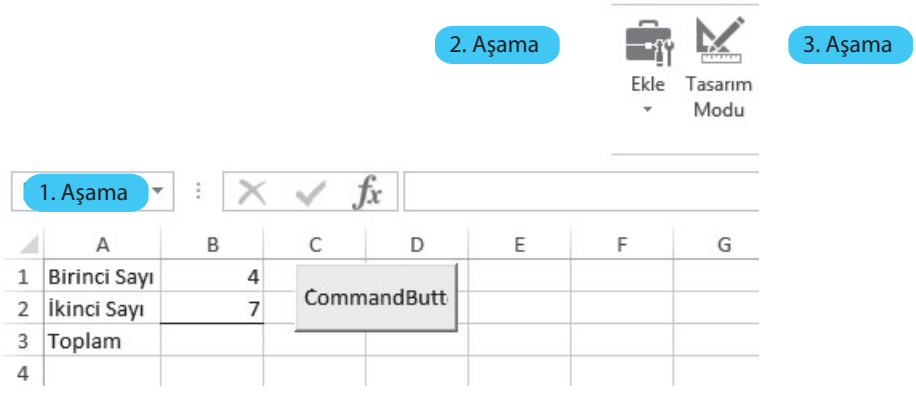
Beşinci aşamada Excel sayfasındaki veriler (4 ve 7 rakamı) hazırlanan değişken paketlerine aktarılır (atanır).

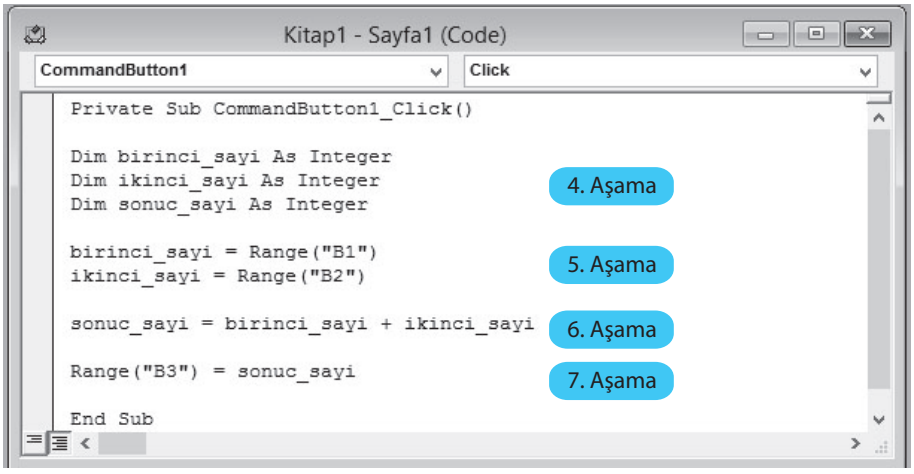
Altıncı aşamada toplama işlemi gerçekleştirilir.

Yedinci aşamada fonksiyon sonucu (4+7) yani 11 rakamı B3 hücresine yazdırılarak çıktı üretilir ve VBA yazılımı sonlanır.

Şekil 5.3

Aritmetik İşlem Örneği Excel Sayfası Tasarım Örneği





Şekil 5.4
Aritmetik İşlem Örneği VBA Sayfası Tasarım Örneği

Aritmetik İşlem Örneğindeki VBA kodlarını kullanarak B1 hücresi ile B2 hücresi arasında çarpma işlemi yapınız.

SIRA SİZDE 1

Deyimler

Deyimler VBA'da tasarlanan fonksiyonlar için hayati öneme sahiptir. Örneğin, bir sihirbazlık gösterisinde, hem 2'den hem de 13'ten büyük bir tam sayı tutulması istenildiğinde ne yapılmalıdır? İnsan beyninin bunu nasıl yaptığı incelenecek olursa, bunun için öncelikle beynin öğrenmiş olduğunu tamsayılar liste şeklinde sıralanır. Sonraki aşamada bu sayılardan biri seçilir. Seçilen sayının sihirbazın istediği iki koşulu aynı anda sağlayamayıp sağlamadığı ikili karşılaştırmalar ile kontrol edilir. Seçilen sayı iki koşulu aynı anda sağlamıyorsa elenir. Koşul gerçekleşinceye kadar işlem tekrarlanır. Böyle bir işlemi VBA yazılımında bir fonksiyona yaptırmamız için koşul deyimlerine, mantıksal sınamalara ve karşılaştırma operatörlerine ihtiyaç vardır. VBA yazılımında sık kullanılan deyimler Tablo 5.3'te özetlenmiştir.

DEYİMLER	
	Koşullu Deyimler
	if… Then… Else: Belirli bir mantıksal şartın doğru olması durumunda if bloğu, yanlış olması durumunda ise else bloğu çalışır.
	Select Case: Bir ifadeyi değerlendirip buna göre birkaç deyim bloğundan birini çalıştıran koşullu deyimdir.
	Döngü Deyimleri
	For…Next: Belirtilen tekrar sayısına göre VBA deyim bloğunu çalıştırır.
	For Each...Next: Belirli bir deyim grubunu bir koleksiyonun her bir öğesi için bir kez çalıştırır.
	Do...Loop: Şarta bağlı olarak deyimler bloğunda döngü oluşturur.
	While…Wend: Şarta bağlı olarak deyimler bloğunda döngü oluşturur.

Tablo 5.3
Deyimler

Zaman Fonksiyonları

Kullanıcı ihtiyaçlarına bağlı olarak zamana ve sürüye bağlı işlemler yapılabilir. Bu durumda VBA yazılımında önceden tanımlanmış fonksiyonlar kullanılır. Bu fonksiyonlara istenen özelliklerde veriler gönderilir ve bu fonksiyonlar işlem yaparak çıktı olarak bize işlenmiş verileri geri gönderir. Tablo 5.4'te zaman fonksiyonlarından önemli olanları özetlenmiştir.

Tablo 5.4
Zaman Fonksiyonları

ZAMAN FONKSİYONLARI	
	Date () Bilgisayarın sistem saati ile bağlantı kurarak bu günün tarihini alır.
	CDate (string) String veri tipindeki verileri *Date* formatına dönüştürmekte kullanılır. Örnek: CDate*("06.05.1972")*, sonuç: 06.05.1972
	Minute, Day, Month, Year(date_value) *Date* veri tipi bu fonksiyonlardan biri ile yollandığında sırasıyla dakika, gün, ay, yıl *Integer* veri tipinden geri döner. Örnek: *Day ("06.05.1972")*, sonuç: 06
	DateDiff (interval, date1, date2) Bu fonksiyon iki tarih arasındaki farkı, istenen tarih türünden bulmak için kullanılır. Geri dönüşü *Integer* veri tipinde olur. "*yyyy*" yıl, "*m*" ay, "*y*" yılın kaçıncı günü, "*d*" gün, "*w*" haftanın günü, "*ww*" hafta, "*h*" saat, "*n*" dakika, "*s*" saniye türünden iki farklı tarih arasındaki farkı bulur. Örnek: *DateDiff("yyyy", "20.07.1982","17.11.2015")*, sonuç: 33
	DateAdd (interval, number, date) Bu fonksiyon bir tarihe zaman eklemek için kullanılır. Geri dönüşü *Date* tipinde olur. Eklenen zaman *Integer* tipinde olmalıdır. "*yyyy*" yıl, "*m*" ay, "*y*" yılın kaçıncı günü, "*d*" gün, "*w*" haftanın günü, "*ww*" hafta, "*h*" saat, "*n*" dakika, "*s*" saniye türünden tarih verisine eklemeler yapılabilir. Örnek: *DateAdd("d", 12173, "20.07.1982")*, sonuç: 17.11.2015

Excel hücrelerinde verilen tarihleri alıp bu güne kadar geçen süreyi hesaplayan bir VBA yazılımı hazırlanmak istendiğinde veri tipleri, operatörler, deyimler ve zaman fonksiyonlarının kullanıldığı bir tasarım yapmak gerekir. Bu örnekte hesaplama işlemlerini yapan ana programa bağlı bir fabrika gibi çalışan alt fonksiyon tasarımı da yapılacaktır. Böylece birden fazla tarih için tekrarlanan komut setleri alt fonksiyonlara yollanılarak hesaplama yapılabilecektir. Bu yazılımın tasarım ve kodlama aşamaları aşağıdaki gibidir:

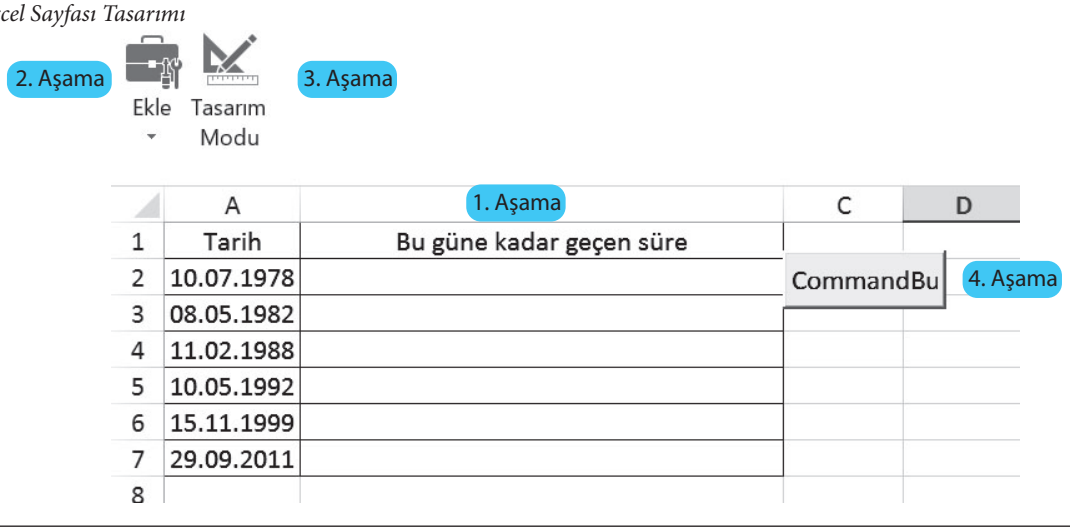
Birinci aşamada Excel hücrelerindeki sütun ve satırların Şekil 5.5'te görüldüğü gibi tasarlanması gerekir. Excel sayfasında A sütununda önceden belirlenmiş toplam 6 tarih verilmiştir. B sütununda tarihlerin karşısı boş bırakılmıştır. Komut düğmesine basıldığında bu güne kadar geçen süre VBA tarafından hesaplanarak B sütununa yazılacaktır.

İkinci aşamada Geliştirici sekmesinde Ekle tuşuna basıp komut düğmesi eklenir.

Üçüncü aşamada tasarım modu tuşuna basılır.

Dördüncü aşamada komut düğmesinin üzerine çift tıklanır veya klavyede alt ve F11 tuşlarına beraber tıklanır.

Şekil 5.5
Excel Sayfası Tasarımı



Bu aşamadan sonra kullanıcı fonksiyonlarını tanımlamak için VBA kodlarına geçiş yapılır. Bir fabrika gibi çalışacak alt fonksiyon *hesaplama* etiketi ile tanımlanabilir. Bu alt fonksiyon kullanıcı ihtiyaçlarına göre hazırlanacaktır. Farklı bir deyişle alt fonksiyon ile Excel'de kullanıcı kendi komut yapısını tasarlayabilir. *Hesaplama* olarak adlandıracağımız bu alt fonksiyona tarih verisi yollandığında bu güne kadar geçen süreyi hesaplayıp yazı değişkeni şeklinde çıktı üretecektir. Böylece VBA yazılımında yer alan hazır komutları da kullanan yeni bir komut tasarlanmış olacaktır.

Beşinci aşamada Public anahtarı kullanılarak genel seviyede çalışacak şekilde *hesaplama* etiketli alt fonksiyon tanımlanır. Alt fonksiyona girecek veri tipi parantez içerisinde belirtilir. Örnekte *hücredeki_tarih* etiketli bir değişken paketi *Date* veri tipi verilerini almaktadır. Alt fonksiyonun çıktısı ise parantezin bitimine tanımlanır. Örnekte alt fonksiyon *hesaplama* adında *String* veri tipinde çıktı üretmektedir.

Altıncı aşamada alt fonksiyon içerisinde kullanılacak değişken paketlerinin tanımını yapılmıştır. Hesaplamada bu günün tarihi *Date*, bu güne kadar geçen yıl, ay ve gün *Integer* tipinde olacak şekilde tanımlanmıştır.

Yedinci aşamada bu günün tarihini bulmak için Excel içerisindeki hazır zaman fonksiyonlarından biri olan *Date* komutunu kullanılmıştır.

Sekizinci aşamada Excel içerisindeki hazır fonksiyonlardan bir diğeri olan *DateDiff* komutundan yardım alınmıştır. *DateDiff* komutuna sırasıyla parantez içinde istenilen sürenin tipi (yıl, ay, saat vb.), hücredeki tarih ve bu günün tarihi yollanır. *DateDiff* komutu bilgisayar tarafından çalıştırılıp girdileri değerlendirir, iki zaman arasındaki farkı istenilen süre tipine dönüştürerek *Integer* veri tipinde çıktı sağlar. Çıktılar *altıncı aşamada* tanımlanmış olan değişken paketlerinin içerisine = simgesi ile atanır.

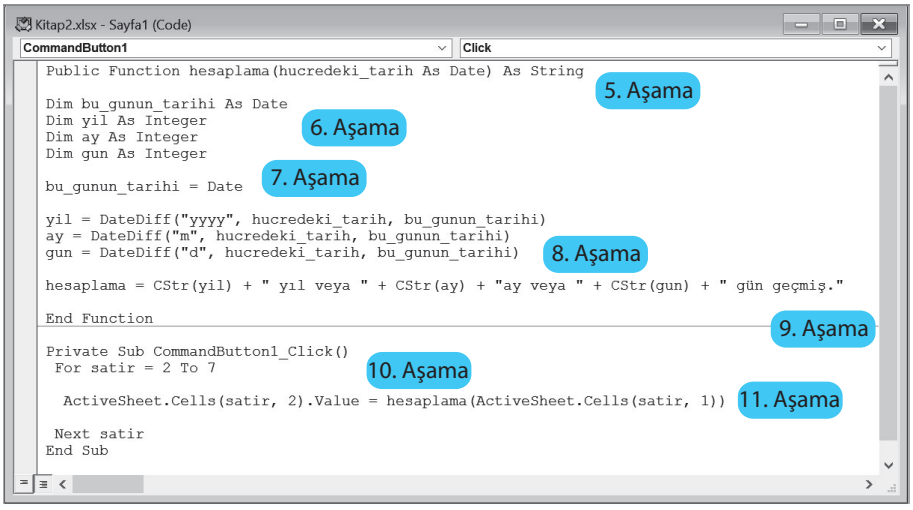
Dokuzuncu aşamada DateDiff komutu ile *Integer* tipinde veri üretilmiştir. Fakat *beşinci aşamada hesaplama* alt fonksiyonun çıktısı *String* yani yazı tipinde olacak şekilde tasarlandığından, *Integer* veri tipindeki verileri *String* veri tipine dönüştürmek için *CStr* komutundan yararlanılır. Bu komut parantez içerisinde gönderilen değişken paketinin içindeki veriyi yazı veri tipine yani *String* tipine dönüştürür. *String* tipine dönüştürülen yıl, ay ve gün değişken paketleri ve ne anlama geldiğini göstermek için tırnak işaretlerini kullanarak yapılan açıklama yazıları + işareti ile bir birlerine bağlanır. Bir birlerine tren katarları gibi bağlanan yazı tipindeki veri Alt fonksiyondan dışarı çıkabilmesi için aynı zamanda alt fonksiyonun isim etiketi olan *hesaplama* değişken paketinin içine atanır. Böylece üretilen değer fabrikadan çıkmış olur. Örnekte hesaplama değişken paketine "37 yıl veya 447ay veya 13612 gün geçmiş." ifadesini yerleşmiş olur.

Onuncu aşamada Ana fabrikanın yapacağı işlemler tanımlanır. Örneğimizde butona basıldığında yapılacak işlemler bu aşamada tanımlanacaktır. Düğmeye basıldığında Excel sayfasındaki 2. satırdan, 7. satıra kadar olan tarih verileri ile ilgili işlem yapılması gerekmektedir. Bu 2. satırdan, 7. satıra kadar tekrarlanan bir döngünün gerekliliğini gösterir. *For* komutu ile *satir* değişken paketi oluşturulur ve bu değişken paketi 2'den başlayıp 7'ye kadar devam edebilir. *Next* komutu döngünün sonuna konulur. *Next* komutuna geldiğinde yazılım *satir* değişken paketinin değerini bir arttırır ve döngüyü tekrar tetikler. Döngünün tetiklenmesi *satir* veri paketinin alabileceği en fazla değer olan 7'ye kadar devam eder. Döngü toplam 6 kez tetiklenerek başa dönecek ve her bir aşamada *For* ve *Next* arasında tanımlanmış fonksiyonu yerine getirecektir.

On birinci aşamada For ve *Next* komutları arasında yapılacak fonksiyon tanımlanır. Fonksiyon Şekil 5.5'te tasarlanmış Excel sayfasındaki A sütunundaki (yani 1. sütundaki) tarih verilerini *beşinci aşamada* tanımlanan hesaplama alt fonksiyonuna göndermektedir. Excel sayfasındaki A2 hücresinden A7 hücresinde kadar olan tarih verileri döngü sayesinde tek tek *hesaplama* fonksiyonuna yollanacaktır. *Hesaplama* fonksiyonu bir fabrika gibi çalışarak çıktı üretecektir. *Hesaplama* alt fonksiyonunun ürettiği *String* tipindeki çıktı işlem yapılan satırın B sütununa yazdırılır.

Şekil 5.6

Bu Güne Kadar Geçen Süreyi Hesaplama Örneği VBA Tasarımı



On ikinci aşamada Şekil 5.7'de görülen örneğin *ikinci aşamasında* oluşturulan komut butonuna basıldığında A sütununda yazan tarihlerden işlem yapılan güne kadar geçen süre B sütunundaki satırlara yazdırılır.

Şekil 5.7

Bu Güne Kadar Geçen Süreyi Hesaplama Örneği Excel Sayfası Çıktısı

	A	B	C	D
1	Tarih	Bu güne kadar geçen süre		
2	10.07.1978	37 yıl veya 447ay veya 13612 gün geçmiş.	CommandBu	
3	08.05.1982	33 yıl veya 401ay veya 12214 gün geçmiş.		
4	11.02.1988	27 yıl veya 332ay veya 10109 gün geçmiş.		
5	10.05.1992	23 yıl veya 281ay veya 8559 gün geçmiş.		
6	15.11.1999	16 yıl veya 191ay veya 5814 gün geçmiş.		
7	29.09.2011	4 yıl veya 49ay veya 1478 gün geçmiş.		

12. Aşama

SIRA SİZDE 2

Excel VBA'da DateDiff ile benzer bir kullanıma sahip DateAdd komutunun amacı verilen tarihlere gün veya ay veya yıl eklemektir. Yukarıdaki örneğin bir benzerini hazırlayıp DateAdd komutunu kullanarak VBA yazılımında komut düğmesine hangi tarihte basıldığını bulmaya çalışınız.

Karakter Fonksiyonları

Excel VBA yazılımda sayısal ve tarih işlemlerinin yanında metin veri tipindeki veriler ile de işlemler yapılabilmektedir. Karakter işlemlerinin bir bölümü ASCII (American Standard Code for Information Interchange- Bilgi Değişimi İçin Amerikan Standart Kodlama Sistemi) tablosu ile ilgilidir. ASCII kodu bilgisayarların birbirleri ile veri alış verişinde kullandığı bir tür dildir. VBA yazılımında kodlamada doğrudan görünmesini istenmeyen şifre vb. işlemlerinde kriptolama için kullanılabilmektedir. Excel hücrelerindeki metnin yerini bulma, karakterlerin önündeki veya sonundaki boşlukları almak gibi işlemlerde karakter fonksiyonları kullanılabilmektedir. Excel VBA yazılımda sıkta kullanılan karakter fonksiyonları Tablo 5.5'te verilmiştir.

Tablo 5.5
Karakter Fonksiyonları

KARAKTER FONKSİYONLARI	
	Asc(string) Komut içerisine karakter konulur. Komut karakterin karşılığı olan ASCII değerini döner. Örnek: *Asc ("W")*, ASCII değeri: 87
	Chr(ascii_value) Komut içerisine bir karakterin ASCII değeri konulur ve karşılığı olan karakteri döndürür. Asc fonksiyonunun ters işlemidir. Örnek: *Chr (87)*, Karakter: W
	InStr, InStrRev([start], string, substring, [compare]) Bir karakter katarının diğer bir karakter katarı içinde geçtiği ilk yeri bulmak için kullanılır. InStrRev karakter katarının sonundan itibaren bulur. Örnek: *InStr(Başlangıç Yeri, Araştırılacak Katar, Aranan Katar, Karşılaştırma Türü)*
	RTrim, LTrim, Trim(text) Bir karakter katarından sırasıyla sağından, solundan ve hem sağından hem de solundan boşlukları kaldırır. Örnek: *LTrim(" Anadolu ")* , Sonuç: "Anadolu "
	Left, Mid, Right(text, [number_of_characters]) Karakter katarının başlangıcından, ortasından veya sağından itibaren belirtilen miktarda katar alır. *Left(Stirng,n)* kataın solundan itibaren n karakter alır. Örnek: *Left("Anadolu Üniversitesi",7)*, sonuç: Anadolu
	Replace(old_text, start, number_of_chars, new_text) Bir karakterin içerisindeki tanımlanmış karakterleri yenileriyle değiştirir. Örnek: *Replace("alfabe", "be", " ışını")*, sonuç: alfa ışını
	Len(text, [number_of_characters]) Bir karakter katarındaki karakter sayısını bulur ve döndürür. Boşlukları da sayar. Örnek: *Len("Anadolu Üniversitesi")*, sonuç: 20
	StrComp(string1, string2 [, compare]) İki karakter katarını bir birleriyle karşılaştırır. Karakter katarları aynı ise 0 sonucunu döndürür. Örnek: *StrComp ("www.anadolu.edu.tr", "www.anadolu.edu.tr")*, sonuç: 0

Excel hücrelerinin içerisine http://www.anadolu.edu.tr Anadolu Üniversitesi İnternet sayfasında yayınlanmış olan duyuruları kopyalandığında duyuruların öncesinde ve sonrasında karakter boşlukları yer alır. Excel VBA'da aşağıda yer alan örnekteki gibi Excel hücrelerindeki metinlerin sağındaki ve solundaki boşlukları alan bir yazılım hazırlanabilir:

Birinci aşamada Excel hücrelerindeki sütun ve satırların Şekil 5.8'de görüldüğü gibi tasarlanması gerekir. Excel sayfasında A sütununda Duyuru metinleri yapıştırılır.

İkinci aşamada Geliştirici sekmesinde Ekle tuşuna basıp komut düğmesi eklenir.

Üçüncü aşamada tasarım modu tuşuna basılır.

Dördüncü aşamada komut düğmesinin üzerine çift tıklanır veya klavyede alt ve F11 tuşlarına beraber tıklanır.

Şekil 5.8

Karakter Fonksiyonu Örneği için Excel Sayfası Tasarımı

Beşinci aşamada Şekil 5.9'de görülüğü gibi VBA kodlarına geçiş yapılır. *satir_no* etiketli *Integer* veri tipi taşıyacak bir değişken paketi ve *hucre_ici_veri* etiketli *String* tipinde veri bir değişken paketi tanımlanır. *satir_no* değişken paketi üzerinde işlem yapılan satır numarasını tutmak için kullanılacaktır. *hucre_ici_veri* değişken paketi ise sağından ve solundan boşluk alınması planlanan duyuru metinlerini Excel hücrelerinden alıp, verileri VBA yazılımının içerisine taşıyacaktır.

Altıncı aşamada hücre_ici_veri değişken paketinin içerisine daha sonraki aşamalarda tasarlanması planlanan döngüyü tetikleyecek rastgele bir metin ataması yapılmıştır.

Yedinci aşamada Excel hücrelerinde hangi satırdan başlanacağı bilgisi girilmiştir. Şekil 5.6'da görüldüğü gibi A sütununu 1. hücresinde Duyuru şeklinde bir başlık vardır. İşlemler A2 hücresinden başlayacağından *satir_no* içerisine 2 sayısı atanmıştır.

Sekizinci aşamada hücrelerdeki verileri tek tek VBA yazılımı paketlerine aktarılacak bir döngü oluşturulmuştur. Boş hücre gelinceye kadar döngü devam edecektir. Bunun için *hücre_ici_veri* etiketli değişken paketi eşit değilse boşa (<>"") yordamı kullanılmıştır.

Dokuzuncu aşamada Excel sayfasındaki hücre içindeki veri *hücre_ici_veri* değişken paketine atanmıştır. Atama işlemi yapılırken karakter fonksiyonlarından *Trim* komutundan faydalanılmıştır. Trim komutu ilgili satırdaki veriyi hücreden aldıktan sonra sağ ve solundaki boşlukları aldıktan sonra veri *hücre_ici_veri* paketinin içerisine atmaktadır.

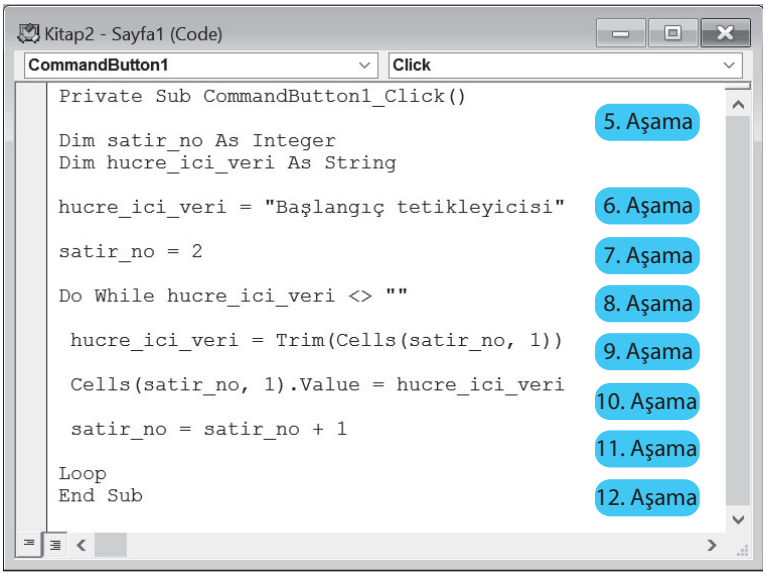
Onuncu aşamada işlem yapılan Excel hücresinin içine *hücre_ici_veri* paketindeki sağ ve solundan boşluklar alınan veri yüklenir.

On birinci aşamada sonraki satıra geçilebilmesi için satır sayısı bir arttırılır.

On ikinci aşamada döngü başa dönerek tetik mekanizmasını yani hücre içindeki veri boş değilse ifadesini tetikler. *hucre_ici_veri* paketinin içi boş ise son satıra gelinmiş demektir. Bu durumda tetik mekanizması çalışmaz ve döngüden çıkılır. Son olarak yazılım sonlanır.

Şekil 5.9

Karakter Fonksiyonu Örneği VBA Tasarımı



Matematik Fonksiyonları

Excel VBA yazılımda aritmetik işlemlerin yanında matematik, istatistiksel ve geometrik işlemlerde hazır matematik komutları kullanılabilmektedir. Bu komutlardan *Abs()*, sayıların mutlak değerlerinin alınması için kullanılmaktadır. *Sin, Cos, Tan, Cot* gönderilen sayıların sinüs, kosinüs, tanjant ve kotanjant değerlerinin hesaplanmasında kullanılan komutlardır. Rassal sayılar üretilmesi için *Rnd* komutu kullanılmaktadır. *Rnd* komutu istenilen olasılarda (sürekli olasılık dağılımı, ayrık olasılık dağılımı vb.) olacak şekilde düzenlenebilmektedir. Aksi bildirilmediğinde *Rnd* komutu bilgisayar sistem saatini temel alarak 0 ile 1 arasında rastgele sayılar üretir. *Val* komutunu kullanılarak ise bir metin içerisindeki rakamlar bulunabilir. Ondalık sayıların yuvarlanmasında hazır VBA kodlarından *Raund* komutu kullanılır. Excel VBA yazılımda sıkça kullanılan matematik fonksiyonları Tablo 5.6'da verilmiştir.

MATEMATİK FONKSİYONLARI	**Abs(number)** Sayısal veri tipinde verilen bir verinin mutlak değerini alır. Örnek Abs(-1223), sonuç: 1223
	Sin, Cos, Tan, Cot(number) Sayısal veri tipinde verilen bir verinin sırasıyla sinüs, kosinüs, tanjant ve kotanjantını almak için kullanılır. Örnek Sin(3), sonuç: 0.141120008
	Rnd Bilgisayar saati ile bağlantı kurarak Single veri tipinde (0 ile 1 arasında pozitif) rassal sayı üretir. Tam sayı üretmesi için Int ((upperbound - lowerbound + 1) * Rnd + lowerbound) formülü kullanılabilir. Örnek: Int((10 - 5 + 1) * Rnd + 5), sonuç: 10 ile 5 arasında rassal sayı üretir.
	Val(string) String tipinde verilen bir yazının içerisindeki rakamları bulmak için kullanılır. Dönen değer Double tipindedir. Örnek: Val("26470 Eskişehir"), sonuç: 26470
	Round(expression, [decimal_places]) Ondalık sayıyı yuvarlamak için kullanılır. Virgülden sonra kaç rakamın yuvarlanacağı verilmelidir. Örnek: Round(34.8233341677, 2) , sonuç:34.82

Tablo 5.6
Matematik Fonksiyonları

Günümüz bilişim teknolojilerine erişim için en çok kullanılan unsurlardan biri şifrelerdir. E-posta adresinden, İnternet bankacılık hesabına, vatandaşlık işlemlerinden, öğrenci işlemlerinin büyük bölümüne erişebilmek için şifreler kullanılır. Excel VBA yazılımı yardımı ile 6 karakterden oluşan içerisinde harf ve sayılar bulunan rassal bir şifre üretme aracı aşağıdaki gibi tasarlanabilir:

Birinci aşamada Excel hücrelerindeki sütun ve satırların Şekil 5.10'da görüldüğü gibi tasarlanması gerekir. Excel sayfasında A sütununda üretilecek geçici şifre A2 hücresine yazdırılacaktır.

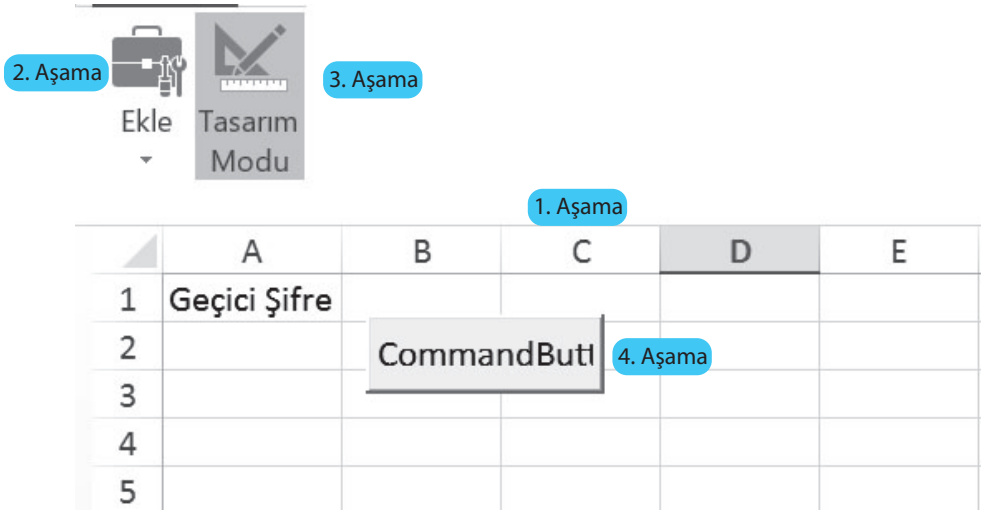
İkinci aşamada Geliştirici sekmesinde Ekle tuşuna basıp komut düğmesi eklenir.

Üçüncü aşamada tasarım modu tuşuna basılır.

Dördüncü aşamada komut düğmesinin üzerine çift tıklanır veya klavyede alt ve F11 tuşlarına beraber tıklanır.

Şekil 5.10

Matematik Fonksiyon Örneği İçin Excel Sayfası Tasarımı



Beşinci aşamada Geçici şifrelerdeki karakterleri üretebilecek Şekil 5.11'de görüldüğü gibi *karakter_uret* etiketli bir fonksiyon tasarlanmalıdır. Bu fonksiyon bir fabrika gibi çalışarak rassal olarak büyük harf, küçük harf ve sayısal karakterler üretecektir. Rassal karakter üretebilmesi için girdi olarak Tablo 5.7'deki ASCII tablosundaki değerleri kullanacaktır. *karakter_uret* fonksiyonuna *Integer* veri tipinde iki sayı yollandığında *String* tipinde veri üretecektir. *Integer* tipinde yollanan ilk sayı ASCII kod tablosundaki en küçük değeri, ikinci değer ASCII kod tablosundaki en büyük değeri temsil edecektir. Fonksiyon bu iki değer arasında bir karakter üretebilecek. Örneğin, 65 ile 90 değeri fonksiyona yollandığında A-Z harfleri arasında rassal bir karakter üretecek, *karakter_uret* fonksiyonu aynı zamanda *String* tipinde verileri de taşıyabilecek bir değişken paketi olacaktır.

Değer	Karakter	Değer	Karakter	Değer	Karakter	Değer	Karakter
48	0	71	G	87	W	109	m
49	1	72	H	88	X	110	n
50	2	73	I	89	Y	111	o
51	3	74	J	90	Z	112	p
52	4	75	K	97	a	113	q
53	5	76	L	98	b	114	r
54	6	77	M	99	c	115	s
55	7	78	N	100	d	116	t
56	8	79	O	101	e	117	u
57	9	80	P	102	f	118	v
65	A	81	Q	103	g	119	w
66	B	82	R	104	h	120	x
67	C	83	S	105	i	121	y
68	D	84	T	106	j	122	z
69	E	85	U	107	k		
70	F	86	V	108	l		

Tablo 5.7
Matematik Fonksiyonları Örneğinde Kullanılan ASCII kod Tablosu

Altıncı aşamada Integer tipinde verileri taşıyabilecek *rassal_sayi* etiketli bir değişken paketi tanımlanmıştır.

Yedinci aşamada En büyük ASCII kod değeri ile en küçük ASCII kod değeri arasında rassal olarak sayı üretecek bir *Rnd* komutu hazırlanmıştır. Rassal üretilen sayı *rassal_sayi* etiketli değişken paketinin içine atanmıştır.

Sekizinci aşamada Rassal üretilmiş olan sayının ASCIII kod tablosundaki karşılığının bulunması için *Chr* komutundan faydalanılmıştır. Bu aşamadan sonra Excel sayfasında *ikinci aşamada* tasarlanan buton için fonksiyon oluşturulacaktır.

Şekil 5.11

Matematik Fonksiyonu Örneği İçin VBA Tasarımı Karakter Üretme Fonksiyonu

```
Public Function karakter_uret(en_kucuk_ASCII As Integer, en_buyuk_ASCII As Integer) As String   5. Aşama
Dim rassal_sayi As Integer   6. Aşama
rassal_sayi = Int((en_buyuk_ASCII - en_kucuk_ASCII + 1) * Rnd + en_kucuk_ASCII)   7. Aşama
karakter_uret = Chr(rassal_sayi)   8. Aşama

End Function
```

Dokuzuncu aşamada rassal_karakter_turu_secimi etiketli *Integer* tipinde verileri taşıyabilecek bir veri paketi ve *gecici_sifre* etiketli *String* verilerini taşıyacak bir başka veri paketi tanımlanmıştır.

Onuncu aşamada 0'dan 6'ya kadar dönecek bir *For Next* döngüsü kurulmuştur. Bu döngü her seferinde rassal karakter üreterek geçici veri içeresine bu karakteri ekleyecektir. Böylece 6 karakterden oluşan geçici şifre oluşturulacaktır.

On birinci aşamada rassal_karakter_turu_secimi değişken paketi içerisine 1 ile 3 arasında *Rnd* komutu ile bir sayı üretip atanmıştır.

On ikinci aşamada bir önceki aşamada *rassal_karakter_turu_secimi* değişken paketine atanmış veri kullanılarak *Select Case* yapısı kurulmuştur. Bu yapıya göre rassal olarak üretilen sayı 1 ise *karakter_uret* fonksiyonuna 0-9 arasında bir sayı üretecek en büyük ve en küçük ASCII kod değerleri yollanmaktadır. Üretilen rassal sayı 2 ise *karakter_uret* fonksiyonuna a-z arasında bir sayı üretecek en büyük ve en küçük ASCII kod değerleri

olan 97 ve 122 sayıları yollanmaktadır. Eğer sayı 3 ise *karakter_uret* fonksiyonuna 65 ve 90 sayıları yollanarak fonksiyonun A-Z arasında harf üretmesi sağlanmıştır. Üretilen karakterlerin *dokuzuncu aşamada* tanımlanmış *gecici_sifre* veri paketi içerisine eklenebilmesi için *gecici_sifre=gecici_sifre+*Üretilen karakter gibi bir denklem kurulmuştur. Denklemde *gecici_sifre* değişken paketine daha önce üretilmiş olan geçici şifre ve yeni üretilen karakter art arda eklenerek atanmaktadır. Çünkü bir değişken paketi içerisinde tek seferde tek veri taşımaktadır. Bu nedenle değişken paketi içerisine atama yapılmadan önce değişken paketinin içindeki veri çıkartılır ve yeni veri ile birleştirilerek tekrar atanır. Bu yaklaşım güncel hayatta alışılmış bir durum değildir. Örneğin, bir seyahat valizinin hazırlandığı bir durumda valize bir pantolon koymak istendiğinde, bu işlem VBA mantığında yapılsaydı önce tüm valizin boşaltılması gerekecekti. Sonra çıkarılan kıyafetlerin üzerine pantolon koyulup tüm diğer kıyafetler valize tek seferde koyulacaktı. Bu mantıktaki denklemlere VBA yazılımında sıklıkla başvurulmaktadır.

On altıncı aşamada döngü *Next* komutu ile tekrar tetiklenecektir.

On yedinci aşamada Excel sayfasında A2 hücresine geçici şifre yazdırılarak buton fonksiyonu tamamlanmıştır.

Şekil 5.12

Matematik Fonksiyonu Örneği İçin VBA Tasarımı Karakter Üretme Fonksiyonu

```
Public Function karakter_uret(en_kucuk_ASCII As Integer, en_buyuk_ASCII As Integer)   9. Aşama
Dim rassal_sayi As Long
rassal_sayi = Int((en_buyuk_ASCII - en_kucuk_ASCII + 1) * Rnd + en_kucuk_ASCII)
karakter_uret = Chr(rassal_sayi)
End Function
Private Sub CommandButton1_Click()
Dim rassal_karakter_turu_sec As Integer
Dim gecici_sifre
For dongu_sayisi = 0 To 6   10. Aşama
    rassal_karakter_turu_sec = Int((3 - 1 + 1) * Rnd + 1)   11. Aşama
    Select Case rassal_karakter_turu_sec   12. Aşama
        Case 1
        gecici_sifre = gecici_sifre & karakter_uret(48, 57)   13. Aşama
        Case 2
        gecici_sifre = gecici_sifre & karakter_uret(97, 122)   14. Aşama
        Case 3
        gecici_sifre = gecici_sifre & karakter_uret(65, 90)   15. Aşama
    End Select
Next   16. Aşama
Range("A2").Value = gecici_sifre   17. Aşama
End Sub
```

İNTERNET

VBA fonksiyonları ile ilgili ayrıntılı örnekler ve açıklamaları *http://www.vbtutor.net/VBA/vba_tutorial.html* adresinden bulunabilir.

Özet

VBA tasarım mantığını açıklamak.

Yazılım geliştiricilerinin temel işlevi sorunları çözecek sistem tasarımı yapmaktır. Bu nedenle hangi verilerin kullanıcıdan alınacağına, bu verilerin hangi süreçler ile elde edileceğine ve çıktıların neler olacağına karar vermek gerekir. Yazılım geliştiricinin tasarlayacağı sistemin büyüklüğü, karmaşıklığı ve diğer sistemler ile ilişkisi sonsuzdur. VBA programlamasında girdiler Excel programı hücrelerinden alınır, fonksiyon ve yordamlarla süreçlenir, hücreler veya mesaj kutuları ile çıktı üretir. VBA programlamasında yapılan işin süreçlerine göre kullanıcılar tarafından tasarım yapılır. Tasarımda 5 aşamalı Yazılım Geliştirme Yaşam Döngüsü (SDLC) sıklıkla kullanılan yöntemlerden biridir. SDLC modelinin birinci adımında sorun analiz edilir. İkinci adımda sorunu çözecek bir tasarım geliştirilir. Üçüncü adımda tasarıma uygun kodlama yapılır. Son aşama olan entegrasyona geçmeden önce yazılım güvenlik, kullanışlılık vb. gibi farklı testlerden geçirilir. Entegrasyon aşamasında yazılım kullanıcıların hizmetine sunulur.

Değişkenleri ve verileri tanımlamak.

Girdilerin yordam ve fonksiyonlarda kullanılması için "Değişken"ler kullanılır. Değişkenler yazılım içerisindeki paketlere benzer. Değişken paketlerinin içine veriler konulur ve yazılımda gerektiği zaman bu paketler açılarak verilerin kullanılması sağlanır. Değişken paketlerinin kullanılmadan önce içerisinde hangi tür verilerin gireceği ve bu değişken paketinin adının ne olacağı bilgisi VBA yazılımına tanıtılmalıdır. Değişken paketi tanımlandıktan sonra paketin içerisine veri ataması için = (eşit) simgesi kullanılır. Değişken paketlerinin VBA yazılımında verileri taşıyabileceği yerlere yaşam alanı denilir. Yaşam alanları değişkenlerin tanımlanma biçimine göre genel (global) seviye, modül (module) seviye ve yordam (procedure) seviye olabilir. VBA yazılımında da verilerin özelliğine uygun değişken paketlerinin kullanılması gereklidir. Bu veri tipleri sayısal, metin, sabit, mantıksal ve esnek veri tipleri olabilir. Sayısal tanımlanmış değişken paketleri; toplama, çıkartma, bölme ve çarpma gibi çeşitli aritmetik işlemler için rakamlardan oluşan verileri kapsar. Metin veri tipi karakterlerden (genellikle harflerden) oluşur. Sabit veri tipi VBA yazılımı içerisinde yer alacak ve değişmemesi gereken verilerdir. Tarih veri tipi Excel hücrelerindeki takvim ve/veya saat şeklinde yazılmış tarih verileri için kullanılır. Mantıksal veri tipleri VBA yazılımında sadece doğru/yanlış (true/false) verilerini alabilir. VBA yazılımında tipini ön göremediğimiz veriler için esnek veri tipi kullanılır.

Fonksiyon ve yordamları açıklamak.

Yordamlar VBA yazılımında tasarlanan sistemi tanımlamakta kullanılan özel bir tür dildir. Yordamlar tıpkı insanların kullandığı dil gibi özel bir yapıdaki söz dizimi içinde belirtilir. VBA yazılımında fonksiyonların temel işlevi girdileri süreçlemek ve fonksiyonun amacına yönelik çıktılar üretmektir. VBA yazılımında fonksiyon çalıştırıldığında, çalıştırılan fonksiyon amacına uygun şekilde veriler ile işlem yaparak sonuç üretir. Sık kullanılan fonksiyonların bir bölümü Excel VBA tabanında hazır olarak bulunur ve komutlar şeklindedir. Açık sistem yaklaşımı ile tanımlanmış bir VBA yazılımında fonksiyonların ürettiği çıktı diğer fonksiyonlar tarafından kullanılır. VBA'da sıkça kullanılan yordamlar operatörler, deyimler, zaman fonksiyonları, karakter fonksiyonları, matematik fonksiyonlarıdır.

Kendimizi Sınayalım

1. Aşağıdakilerden hangisi bilgi sistemleri tasarımında kullanılan yazılım geliştirme yaşam döngüsünün bir aşaması **değildir?**

a. Analiz
b. Tasarım
c. Kodlama
d. Test
e. Geri Bildirim

2. Excel hücresindeki bir rakam veya yazının hücrelere atanması için aşağıdaki simgelerden hangisi kullanılır?

a. *
b. +
c. $
d. =
e. /

3. VBA'da değişken tanımlama ile ilgili aşağıdaki ifadelerden hangisi **yanlıştır?**

a. Değişken paketlerinin tanımlama işlemi değişken paketinin taşıyacağı verilerin özelliklerine göre yapılır.
b. Değişken paketi adı tanımlanırken bir rakam ile başlanmalıdır.
c Değişken paketlerinin adı nokta içeremez.
d. Değişken paketlerinin adı boşluk içeremez.
e. Değişken paketlerinin adı 255 karakterden fazla olamaz.

4. Aşagıdakilerden hangisi değişken paketlerinin VBA yazılımında verileri taşıyabileceği yaşam alanlarından biridir?

a. Giriş (input) seviye
b. Karakter (char) seviye
c. Hücre (cell) seviye
d. Fonksiyon (function) seviye
e. Genel (global) seviye

5. Rakamlardan oluşan verileri Excel hücrelerinden alıp VBA yazılımındaki fonksiyon ve yordamlarda yeri geldiği zaman kullanmak üzere taşıyan veri tipi aşağıdakilerden hangisidir?

a. Sabit veri tipi
b. Sayısal veri tipi
c. Tarih veri tipi
d. Mantıksal veri tipi
e. Esnek veri tipi

6. Aşağıdakilerden hangisi VBA yazılımda kullanılan döngülerden biri **değildir?**

a. If… Then… Else
b. For…Next
c. For Each...Next
d. Do...Loop
e. While…Wend

7. Bilgisayarın sistem saati ile bağlantı kurarak bu günün tarihini alan zaman fonksiyonu aşağıdakilerden hangisidir?

a. DateDiff
b. Round
c. DateAdd
d. Date
e. Val

8. Bir karakterin içerisindeki tanımlanmış karakterleri yenileriyle değiştiren karakter fonksiyonu aşağıdakilerden hangisidir?

a. Replace
b. Public
c. Trim
d. Dim
e. Abs

9. Aşagıdakilerden hangisi ondalık sayıyı yuvarlamak için kullanılan matematik fonksiyonudur?

a. Val
b. Tan
c. Round
d. Date
e. Abs

10. Matematik fonksiyonlarından "Val" komutunun temel işlevi nedir

a. Sayısal veri tipinde verilen bir verinin mutlak değerini almak.
b. Bilgisayar saati ile bağlantı kurarak Single veri tipinde (0 ile 1 arasında pozitif) rassal sayı üretmek.
c. İki karakter katarını bir birleriyle karşılaştırmak.
d. String tipinde verilen bir yazının içerisindeki rakamları bulmak.
e. Karakter katarının başlangıcından itibaren belirtilen miktarda katar almak.

Kendimizi Sınayalım Yanıt Anahtarı

1. e	Yanıtınız yanlış ise "Fonksiyon ve Yordamların Tasarımı" konusunu yeniden gözden geçiriniz.
2. d	Yanıtınız yanlış ise "Değişkenler" konusunu yeniden gözden geçiriniz.
3. b	Yanıtınız yanlış ise "Değişkenlerin Tanımlanması" konusunu yeniden gözden geçiriniz.
4. e	Yanıtınız yanlış ise "Değişkenlerin Tanımlanması" konusunu yeniden gözden geçiriniz.
5. b	Yanıtınız yanlış ise "Veri Tipleri" konusunu yeniden gözden geçiriniz.
6. a	Yanıtınız yanlış ise "Deyimler" konusunu yeniden gözden geçiriniz.
7. d	Yanıtınız yanlış ise "Zaman Fonksiyonları" konusunu yeniden gözden geçiriniz.
8. a	Yanıtınız yanlış ise "Karakter Fonksiyonları" konusunu yeniden gözden geçiriniz.
9. c	Yanıtınız yanlış ise "Matematik Fonksiyonları" konusunu yeniden gözden geçiriniz.
10. d	Yanıtınız yanlış ise "Matematik Fonksiyonları" konusunu yeniden gözden geçiriniz.

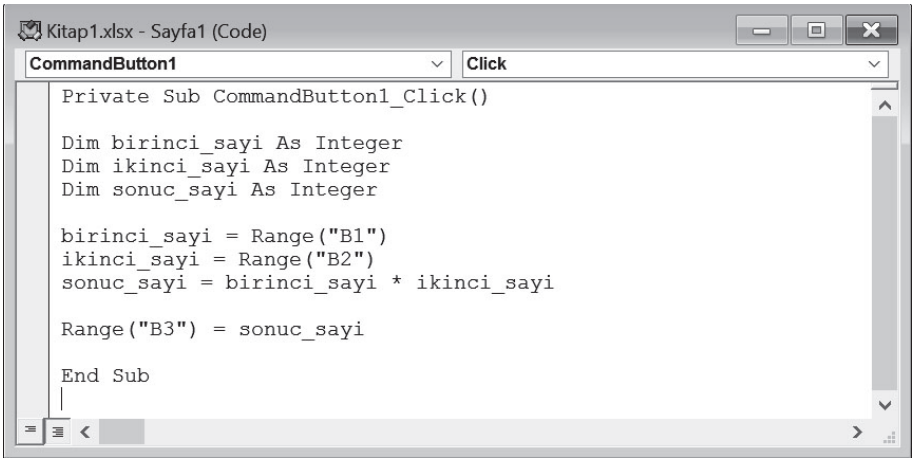
Yararlanılan ve Başvurulabilecek Kaynaklar

Katz, A. (2011). Excel 20110 Made Simple. New York: Apress.

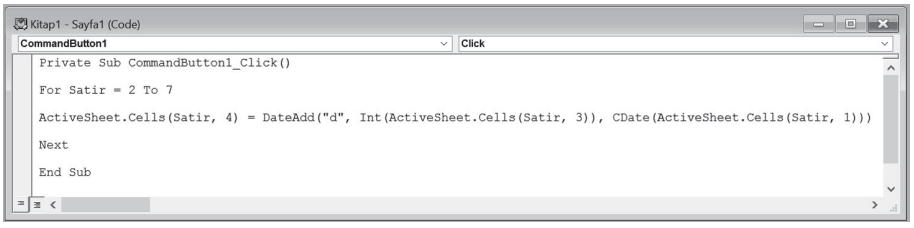
Kiong, L. V. (2009). Made Easy. http://www.vbtutor.net/VBA/vba_tutorial.html.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1



Sıra Sizde 2



İŞLEM TABLOSU PROGRAMLAMA

6

Amaçlarımız

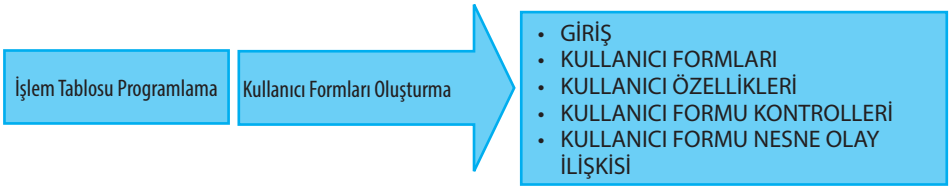
Bu üniteyi tamamladıktan sonra;

- Kullanıcı formlarının nasıl oluşturulduğunu tanımlayabilecek,
- Kullanıcı formlarındaki kontrol nesnelerinin özelliklerini açıklayabilecek,
- Kullanıcı formu kontrol nesnelerini tanıyabilecek,
- Kullanıcı formu nesne olay ilişkisini oluşturabileceksiniz.

Anahtar Kavramlar

- Kullanıcı Formu Kontrol Nesneleri
- Sınıflandırılmış ve Alfabetik Özellik Ögeleri
- Kontrol Nesneleri Olayları

İçindekiler



Kullanıcı Formları Oluşturma

GİRİŞ

Kullanıcıların önceden belirlenmiş bir formatta işlem yapabilmesi için Excel VBA yazılımında formlar tasarlanabilir. Böylece kullanıcılar Excel hücreleri yerine tasarlanan formları kullanır. Bu durum fonksiyon ve yordamların tetiklenmesinde ve verilerin girişinde kolaylık sağlar.

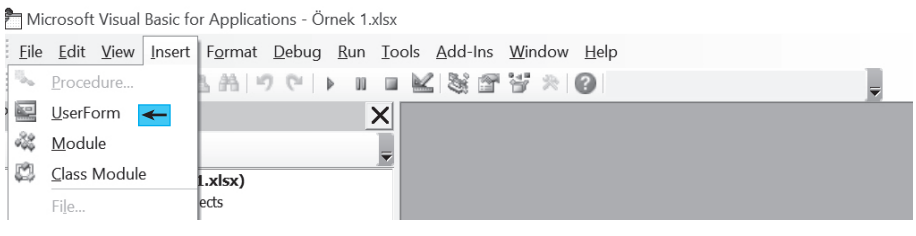
Kullanıcı formlarında özellikler penceresi kullanılarak formun görünümü düzenlenebilir. Düzenlemeler ile kullanıcının ihtiyacına göre tasarımlar hazırlanır ve görsel uyum sağlanabilir. Görsel tasarımın yanında formların davranışları da özellikler penceresinden ayarlanabilir. Örneğin bir butonun tıklandığında tetikleyeceği fonksiyon veya hangi durumda hangi yordamın tetikleneceği davranış kategorisindeki öğeler ile ayarlanabilir. Ayrıca açılan pencerelerin kaynağının belirlenmesi vb. işlemlerde de form özledikleri kullanılır.

KULLANICI FORMLARI

Excel VBA yazılımı, girdileri süreçleyerek çıktılara dönüştüren fonksiyonları içermektedir. Önceki bölümlerde VBA yazılımında girilen ve fonksiyonlar tarafından üretilen verilerin çıktıları için Excel hücrelerinin kullanımı anlatılmıştır. Kullanıcı tarafından VBA yazılımında veri girişi için Excel hücreleri dışında önceden tasarlanmış formlar da kullanılabilmektedir. Kullanıcı formal bir yapıda verilerin girilmesini istiyor ise form tasarlayabilir. Kullanıcı tasarlanan form ile verileri Excel sayfalarından alabilir veya VBA yazılımında fonksiyonların ürettiği sonuçları Excel hücreleri ve/veya kullanıcı formlarına yazdırılabilir.

Kullanıcı formu eklemek için öncelikle Excel sayfasında *Geliştirici* sekmesindeki *Visual Basic* butonuna tıklanarak VBA yazılım platformuna geçmelidir. Şekil 6.1'de görüldüğü gibi *Insert* sekmesine tıklandığında menü açılacaktır. Açılan menü içerisinden *UserForm* seçeneğine tıklanır.

Alt ve F11 kısa yol tuşları kullanılarak da VBA yazılım platformuna geçilebilir.



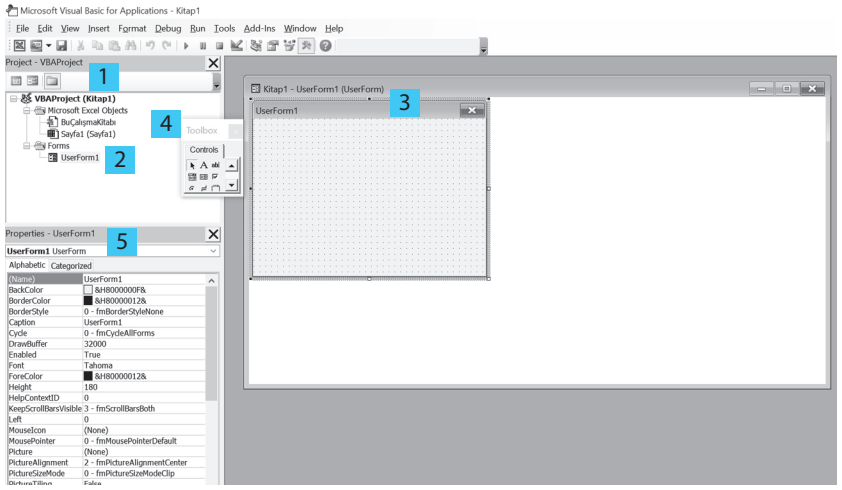
Şekil 6.1

Kullanıcı Formu Ekleme

UserForm seçeneğine tıklandığında VBA kullanıcısının önüne Şekil 6.2'de görülen ekran gelir. Şekil 6.2'de kullanıcı tarafından oluşturulan VBA projesi *Project* penceresinde görülmektedir (1). *Project* penceresinde *Forms* klasörü ve içerisinde yeni bir kullanıcı formu oluşmuştur (2). Kullanıcı formunun tasarımının yapılabilmesi için VBA yazılımı; *UserForm* (3), *Toolbox* (4) ve *Properties* (5) pencerelerini varsayılan olarak açar.

Şekil 6.2

Kullanıcı Form Nesnesi



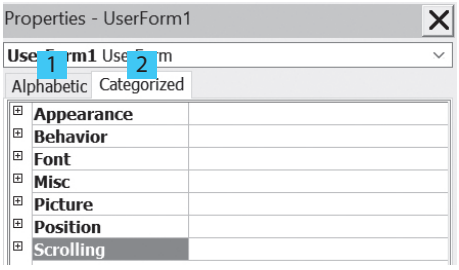
Kullanıcı form öğelerinin üzerinde F1 tuşuna basıldığında özellik ögesi hakkında ayrıntılı bilgi edinilebilir.

KULLANICI FORMU ÖZELLİKLERİ

Kullanıcı formunun tasarım aşamasında görsel düzenlemeler yapılabilmesi için kullanıcı formu özellikleri *(Properties)* penceresi kullanılır. Şekil 6.3'te görüldüğü gibi kullanıcı formu özellikleri penceresi içerisinde kontrol nesnelerinin özellik öğeleri yer alır. Kullanıcı formu kontrollerinin özellik öğeleri, kullanım kolaylığı açısından alfabetik *(1)* veya sınıflandırılmış *(2)* şekilde görüntülenebilir. Alfabetik ve sınıflandırılmış özellik öğeleri aynı içeriğe sahiptir. Fakat pencere içerisindeki öğelerin sıralaması bir birlerinden farklıdır.

Şekil 6.3

Kullanıcı Formu Penceresi



SIRA SİZDE 1

Kullanıcı formu özellik öğelerinin sınıflandırılmış görüntülediğinizde kategori ana başlıklar nelerdir?

Görünüm Kategorisi Öğeleri

Kullanıcı formunda yer alan özelliklerin sınıflandırılmış içeriğinde ilk olarak görünüm *(appearance)* kategorisi yer alır. Kullanılan kontrollerin özelliklerine göre görünüm kategorisindeki öğeler değişir. Görünüm kategorisinin içerdiği sık kullanılan öğeler ve kullanım yöntemleri Tablo 6.1'de görüldüğü gibidir.

Tablo 6.1
Görünüm (Appearance) Kategorisi Öğeleri

GÖRÜNÜM (APPEARANCE) KATEGORİSİ	
	Name Kullanıcı form ve kontrollerinin isimlendirilmesinde kullanılır. VBA yazılımındaki fonksiyon ve yordamlarda işlem yapılabilmesi için İsim *(Name)* öğesi kullanılır. *Name* öğesi fonksiyon ve yordamlarda kullanılacağından Türkçe Karakterli olmaması tavsiye edilir. İsim öğeleri VBA kontrolleri yerleştirildiğinde otomatik olarak üretilir. Karışıklığın önüne geçmek için form ve kontrollerin tümünün isim öğeleri düzenlenmelidir. Mümkün olduğu kadar kontrolün işlev ve türüne uygun isimler seçilmelidir. Örneğin *TextBox* türündeki kontrol nesnelerinin isimleri tb_ ile başlayabilir.
	BackColor, BorderColor, BorderStyle, ForeColor Kullanıcı form ve kontrollerinin görsel tasarımında kullanılan öğelerdir. Form ve kontrollerin arka alan rengini *BackColor*, kenarlık rengini BorderColor, kenarlık stilini BorderStyle, kontrolün ön rengini ise *ForeColor* düzenler.
	Caption Kullanıcı formlarında *UserForm*, *Label, Frame, CheckBox, TabStrip, MultiPage* gibi kontrollere başlık verilmesi için *Cation* öğesi kullanılır. Kontrollerin isimlendirmesinde kullanılan *Name* ile *Caption* öğeleri arasındaki en önemli fark; *Caption* öğesinin son kullanıcı tarafından VBA çalışırken görünmesi, *Name* öğesinin ise sadece VBA fonksiyon ve yordamlarında kullanılmasıdır.
	ControlTip Text VBA yazılımına genellikle veri girişi sağlamakta kullanılan *TextBox, ComboBox, ListBox, CheckBox, CommandButton* gibi kontrollerde bulunan bir öğedir. İçerisine yazılan metin VBA yazılımı çalışırken kullanıcı tarafından fare üzerine gelindiğinde görünür. İlgili kontrolün kullanımı ile ilgili bilgi içerir. Örneğin tarih girilecek bir metin kutusuna tarih formatı ile ilgili açıklama *ControlTip Text* içerisine yazılabilir.
	PasswordChar Sadece *TextBox* türündeki kontrollerde yer alır. VBA yazılımı çalıştırıldığında kullanıcı tarafından metin kutusuna girilen şifre karakterlerinin görünümünü * şeklinde düzenler.
	SpecialEffect Form ve kontroller için görsel efektler uygulanmasında *SpecialEffect* öğesi kullanılır.
	Value Genellikle veri girişinde kullanılan kontrollerde varsayılan bir metnin kontrolün içine otomatik olarak yazdırılmasında *Value* öğesi kullanılır. Açılan menü uygulamalarında *(ComboBox, LisBox vb.)* menünün içereceği liste *Value* öğerine girilebilir.
	Visible Kontrolün kullanıcı tarafından varsayılan olarak görünür olup olmaması *Visible* öğesi ile belirlenir.

Davranış Kategorisi Öğeleri

Kullanıcı formunda yer alan özelliklerin sınıflandırılmış içeriğinin ikincisi Davranış *(Behavior)* kategorisidir. VBA form ve kontrollerinin çeşitli koşullar altında nasıl bir davranış sergileyeceği davranış kategorisi altında yer alan öğeler ile belirlenir. Tab →| tuşuna basıldığında olacak eylem, kullanıcının metin içerisine yazı yazması durumunda otomatik kayma özelliği gibi öğeler Davranış kategorisi öğelerine örnek olarak verilebilir. Davranış kategorisinin içerdiği öğelerden sık kullanılanları Tablo 6.2'de açıklandığı gibidir.

Tablo 6.2
Davranış (Behavior) Kategorisi Öğeleri

DAVRANIŞ (BEHAVIOR) KATEGORİSİ	
	AutoSize, Auto Tab, AutoWordSelect, MaxLength Genellikle veri girişi yapılan metin kutusu gibi kontrollerin otomatik ölçülendirilmesi *AutoSize* öğesi ile *True* veya *False* seçeneklerinden biri seçilerek yapılır. *True* seçildiğinde metin kutusu, içerisine yazılan karakterle doğru orantılı olarak büyür. *MaxLength* veri girişinde kullanılan kontrollerin alabileceği en fazla karakter uzunluğunu belirlemekte kullanılır. *Auto Tab* öğesi; *ComboBox* gibi klavyeden karakter girişi yapılan kontrollerde, maksimum karakter uzunluğu *(MaxLength)* girildiğinde bir sonraki öğeye otomatik geçişi sağlar. *AutoWordSelect* form başladığında daha önceden belirlenmiş bir kelime veya karakterin otomatik seçimini sağlar. Örneğin, VBA'da tasarlanmış bir takvim açıldığında bu günün tarihinin otomatik seçili gelmesi *AutoWordSelect* komutu ile yapılabilir.
	Cycle *UserForm* veya *frame* gibi öglerin içerisine kontroller yerleştirilir. *Cycle* öğesi ile klavyede Tab tuşuna →| basılınca imlecin gideceği kontroller belirlenir. Tüm *UserForm* veya *frame* öğelerinin başına imlecin gitmesi için *0 – fmCycleAllForms* seçilir. İçerisinde işlem yapılan *UserForm* veya *frame*'in başına imlecin gitmesi için *1- fmCycleCurrentForm* seçilir.
	Enabled, Locked Kontrollerin etkin olup olmaması *Enable* öğesi ile sağlanır. Örneğin, VBA'da tasarlanmış bir anket formunda bir sonraki seçeneğin aktif olması açılan kutudaki cevaba bağlı ise *Enabled* seçeneği kullanılır. *Locked* öğesi *Enable* öğesine benzer şekilde kontrolün kitlenmesini sağlar. *Locked* öğesi *True* seçildiğinde imleç ile kontrolün içerisine girilebilir fakat işlem yapılamaz. *Enable* öğesi için *false* seçildiğinde ise imleç metin kutusunun içerisine giremez.
	MatchEntry, MatchRequired *MatchEntry; ListBox, ComboBox* gibi açılan menü öğelerindeki liste içeriğine kullanıcının kolay erişimi için arama yapma imkânı sağlayan öğedir. İlk harfe göre arama yapmak için *0-fmMatchEntryFirstLetter,* tüm karakterler ile arama yapmak için *1-fmMatchEntryComplete* seçeneği seçilir.
	TextAlign Veri girişinde kullanılan kontrollerde karaterlerin hizalama yeri *TextAlign* öğesi ile belirlenir. Karakterlerin sola yaslı olması için *1-fmTextAlignLeft,* ortalı olması için *2-fmTextAlignCenter,* sağa yaslı olması için *3-fmTextAlignRight* öğesi kullanılır.

SIRA SİZDE 2

Bir hesap makinası uygulamasında VBA kullanıcı formunda yer alan sayıların girildiği *TextBox* öğesinin hizalama *(TextAlign)* öğesi nasıl olmalıdır?

Veri Kategorisi Öğeleri

Sadece veri girişine yarayan *TextBox, ComboBox, ListBox, CheckBox, OptionButton, ToggleButton* gibi kontrollerde veri kategorisi *(Data)* bulunur. Açılan bir menü içerisindeki metnin kaynağı ve sırası ile ilgili bilgiler veri kategorisi içerisinde yer alır. Veri kategorisinin içerdiği öğelerden sık kullanılanları Tablo 6.3'te açıklandığı gibidir.

VERİ (DATA) KATEGORİSİ	
	BoundColumn, ColumnCount, ColumnHeads, ColumnWidths Sadece *ComboBox* ve *ListBox* kontrollerinde *BoundColumn, ColumnCount, ColumnHeads, ColumnWidths* öğeleri yer alır. Liste şeklinde gelen seçeneklerde *ColumnCount* açılan kutuda görülecek kolon sayısını belirler. Örneğin, tatil rezervasyonu için kullanılan bir *ListBox* öğesinde ilk kolon şehir, ikinci kolon otel, üçüncü kolon giriş tarihi, dördüncü kolon çıkış tarihi olabilir. *BoundColumn* bağlı kolon sayısını, *ColumnHeads* kolon başlığını, *ColumnWidths* kolon genişliklerinin belirlemede kullanılır.
	ControlSource Kullanıcı formlarında daha önce kayıt edilmiş verilerin kullanılması durumunda kontrol kaynağının yeri *ControlSource* ile belirtilmelidir. Kaynak değiştiğinde kontrol içeriği de otomatik değişir. Örneğin, stok tutan kaydı tutan bir VBA uygulamasında stok bittiğinde açılan menüde biten ürünler otomatik olarak menüden silinebilir.
	ListRows, ListStyle, ListWidth Liste şeklinde gelen seçeneklerde, listedeki veri sayısı maksimum sayıyı aşıyorsa yanlarda kaydırma çubuğu çıkar. Bu durumda listedeki satır sayısı *ListRows* arttırılarak bu durumun önüne geçilebilir. *ListStyle,* görsel olarak *ListBox* ve *ComboBox* kontrollerinin düzenlenmesinde kullanılır. *ListWidth* ise *ComboBox* kontrolü için özel genişlik belirler.
	RowSource *ComboBox* veya *ListBox* kontrolleri için liste kaynağı sağlamakta kullanılır.
	Text Metin kutusu gibi kontrollerin içerisine varsayılan bir metin atamak için kullanılır. VBA yazılımı çalıştırıldığında metin yazılı olarak gelir. *ListBox* ve *ComboBox* gibi liste içeren kontrollerde *Text* içerisine yazılan metin listede olmaması durumunda VBA yazılımı hata uyarısı vermektedir.

Tablo 6.3
Veri (Data) Kategorisi Öğeleri

Yazı Kategorisi Öğeleri

Kullanıcı formu ve kontrolleri içerisindeki yazıların özellikleri Yazı Kategorisi *(Font)* altında yer alan aynı isimli *(Font)* öğe tarafından belirlenir.

Diğer Kategori Öğeleri

Kullanıcı formu ve kontrolleri içerisinde diğer sınıflar arasına konulamayan öğeler için Diğer Kategori Öğeleri *(Misc.)* oluşturulmuştur. Diğer Kategori Öğelerinin içeriği ve kullanımı Tablo 6.4'te özetlenmiştir.

DİĞER (MISC.) KATEGORİ	
	DrawBuffer Kullanıcı form ve kontrollerinin görsel işlem sürecinde ekran kartı hafızasındaki piksel sayısı *DrawBuffer* öğesi ile ayarlanır.
	MouseIcon, MousePointer Kullanıcı form ve kontrolleri üzerine gelindiğinde fare ikonunu *MouseIcon*, fare işaretleyicisi ise *MousePointer* ile ayarlanır.
	TabIndex, TabStop Klavyede Tab tuşuna basıldığında kullanıcı kontrolü üzerine gelmesi için *TabStop* öğesi *True* ayarlanır. Tab tuşuna kaçıncı basışta kontrol üzerine geleceği ise *TabIndex* öğesi ile belirlenir.
	Tag Kullanıcı form ve kontrollerine etiket vermek için *Tag* öğesi kullanılır.
	WhatsThisButton, WhatsThisHelp Kullanıcı formlarına yardımcı olmak ve yardım menüleri oluşturmak için *WhatsThisButton* ve *WhatsThisHelp* öğeleri kullanılabilir.

Tablo 6.4
Diğer (Misc.) Kategori Öğeleri

Resim Kategorisi Öğeleri

Kullanıcı formu ve kontrolleri içerisine resim eklemek için Resim *(Picture)* Kategorisi Öğeleri ile işlem yapılır. Resim kategorisi öğeleri içerisinde yer alan *Picture* öğesi resmin bilgisayardaki yerinin belirlenmesi, *PicturePosition* ise resmin kontrol içerisine nasıl yerleştirileceğine karar verilmesi için kullanılır. VBA'da tasarlanmış yazılımda kaydet butonu içine metin yazmak yerine disket resmi konulması resim kategorisi öğelerinin kullanımına örnek verilebilir.

SIRA SİZDE 3

VBA'da silme işlemi için Resim Kategorisi Öğelerinin kullanımı örneği veriniz.

Pozisyon Kategorisi Öğeleri

Kullanıcı kontrollerinin kullanıcı formu içerisindeki pozisyonunun belirlenmesi için Pozisyon *(Position)* Kategorisi Öğeleri ile işlem yapılır. *Height* kontrolün yüksekliğininin, *Width* ise kontrolün genişliğinin ayarlamasında kullanılır. Kontrolün form içerisindeki hizalamasında *Left* ve *Top* öğeleri ile işlem yapılır. *Left* kontrolün kullanıcı formunda soldan pozisyonunun ayarlanmasına, *Top* ise kontrolün kullanıcı formunda yukarıdan pozisyonunun ayarlanmasına yarar.

Kaydırma Çubuğu Kategorisi Öğeleri

Kullanıcı formunda kaydırma çubuğu özelliklerinin belirlenmesi için Kaydırma *(Scrolling)* Çubuğu Kategorisi Öğeleri kullanılır. *ScrollBars* seçeneği kullanılarak form içerisine dikey veya yatay kaydırma çubuğu konulabilir.

KULLANICI FORMU KONTROLLERİ

Kullanıcı formlarının işlevsel tasarımında araç kutusu *(Toolbox)* penceresi içerisinde yer alan kontrol menüsü *(Controls)* kullanılır. Kontrol menüleri formların içeriğinin belirlenmesinde kullanılan temel araçlardan sadece biridir. Kontrol menüsü şekil 6.4'te görüldüğü gibidir.

Şekil 6.4
Araç Kutusu Kontrolleri

Kontrol menüsü içerisinde yer alan nesneler 3 bölümde özetlenecektir. Kontrol nesnelerinin 1. bölümünde *Select Objects, Label, TextBox, ComboBox, ListBox* nesnelerinin kullanımı Tablo 6.5'te verilmiştir.

Tablo 6.5
Kullanıcı Formu Kontrolleri 1. Bölüm

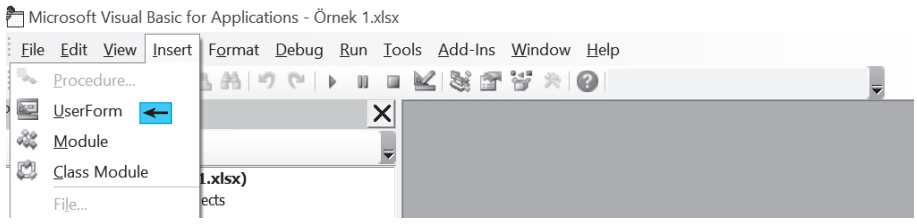
KULLANICI FORMU KONTROLLERİ 1. BÖLÜM	
	Select Objects Simgesi ile gösterilir. Kullanıcı formunun üzerindeki kontrollerin özellik menüsünün aktif olabilmesi için *Select Objects* ile seçilmesi gerekir.
	Label Simgesi ile gösterilir. Sadece tasarım ekranında düzenlenebilen bir metni görüntüler. Diğer kontrol nesnelerinin işlevlerinin etiketlenmesinde kullanılır. VBA yazılımı çalıştırıldığında bu metin kullanıcı tarafından değiştirilemez.
	TextBox Simgesi ile gösterilir. VBA yazılımı çalıştırıldığında kullanıcı tarafından veri girişinde kullanılan bir tür metin kutusudur. Varsayılan ayarı değiştirilmediği sürece tek satırlık veri girişinde kullanılır. Örneğin, açık uçlu sorularda kullanılabilir.
	ComboBox Simgesi ile kullanılır. Veri girişi için kullanılan bir diğer metin kutusudur. Kullanıcılar için birden fazla seçim yapılabilecek liste menüsü *ComboBox* ile oluşturulur. Örneğin, kullanıcı hobilerinin alınmasında *ComboBox* nesnesi kullanılabilir.
	ListBox Simgesi ile kullanılır. Kullanıcıya sadece bir tane seçim yapabileceği açılan liste menüsü hazırlamakta kullanılır. *ListBox ComboBox*'tan farklı olarak sadece bir metnin seçimine izin verir. Örneğin, kullanıcının cinseyitinin alınmasında *ListBox* nesnesi kullanılabilir.

Tablo 6.1'de yer alan kontrol nesnelerinin kullanımı için örnek anket formu hazırlanacaktır. Örnek anket formunun hazırlanması için ilk olarak Excel sayfası açılır. Geliştirici sekmesi veya *Alt* ve *F11* tuşlarına basılarak VBA yazılımına geçilir.

Birinci aşamada VBA yazılımında *Insert* menüsüne basılarak *UserForm* tıklanır (Şekil 6.5).

Şekil 6.5

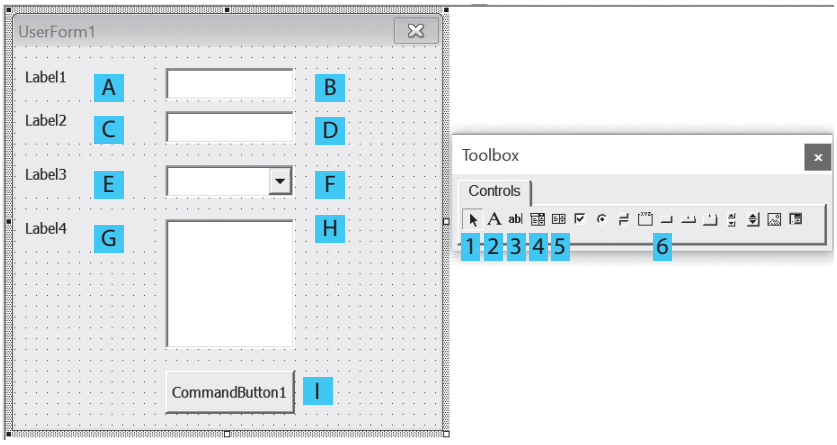
VBA Yazılımında UserForm Ekleme



İkinci aşamada UserForm üzerine kullanıcının anket girişi için kullanacağı kontrollerden *Label, TextBox, ComboBox* ve *ListBox* nesneleri eklenir. Şekil 6.6'da *UserForm* üzerinde tıklanacak yerler alfabetik ve *ToolBox* menüsünde eklenecek kontrol nesneleri ise sayı ile kodlanmıştır. Sırasıyla önce *ToolBox* üzerinde *2* numaralı *Label* kontrol nesnesine sonra *UserForm* üzerinde *A* harfi ile kodlanan yere tıklanır. Böylece form üzerinde *Label* kontrol nesnesi eklenebilir *(2-A)*. Metin kutusu eklemek için *3-B* işlemi yapılır. Sonraki aşamada Şekil 6.6'da görülen formda *Label* eklemek için *2-C, 2-E, 2-G* tuşlarına basılır. *TextBox* eklemek için *3-D, ComboBox* için *4-F* ve son olarak *ListBox* eklemek için *5-H*'ye tıklanır. Kontrollerin yerinin ayarlanması için *ToolBox* üzerinde *1* numara ile kodlanan *Select Objects* tıklanır ve düzenlenmek istenen kontrol nesnesinin yeri ayarlanabilir.

Şekil 6.6

Örnek 1 Araç Kutusu Kontrolleri

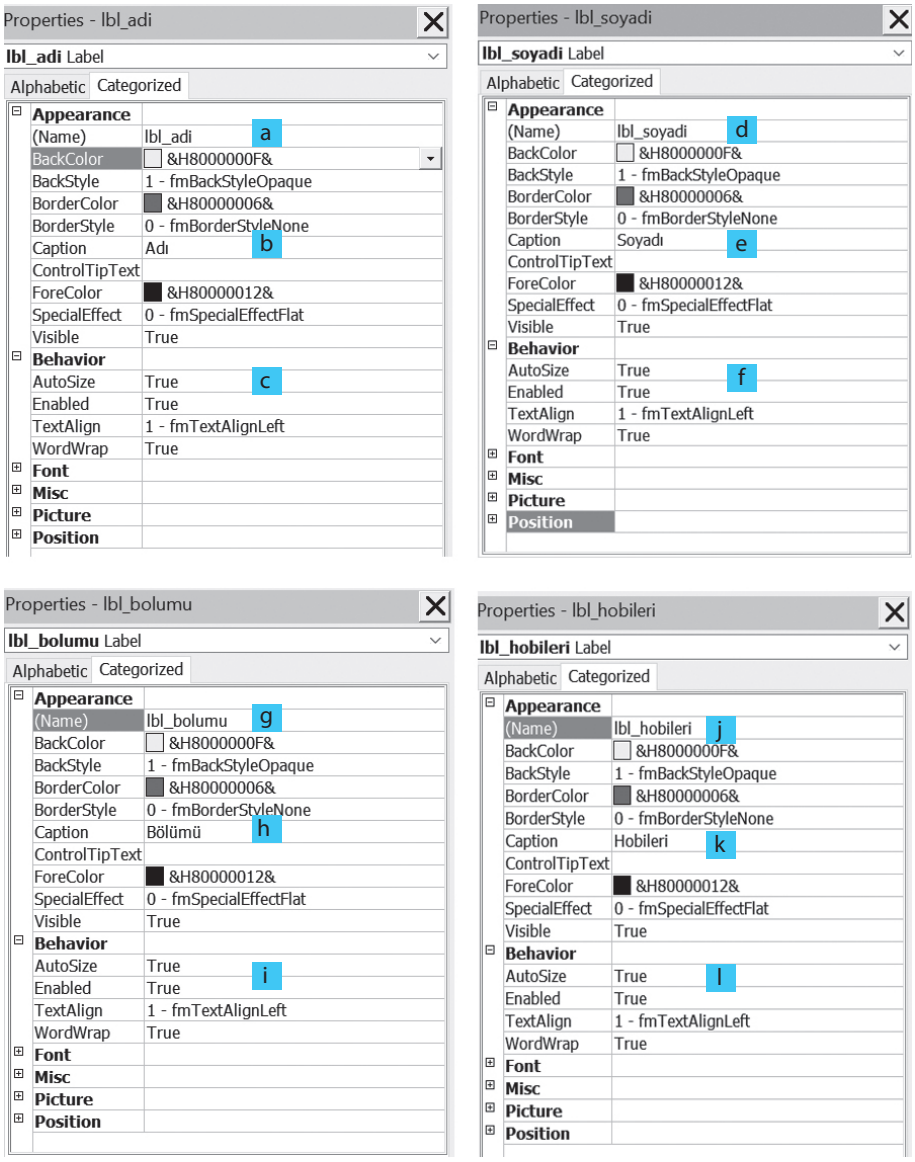


Üçüncü aşamada Label türündeki kontrol nesnelerinin özelliklerinin düzenlenmesi için *Proporties* penceresinde işlem yapılır. Şekil 6.6'da *UserForm* üzerinde *A* harfi ile kodlanmış *Label* özelliklerinin Şekil 6.7'de *a* harfi ile görünen *Name* öğesi *lbl_adi* şeklinde, *b* harfi ile görünen *Caption* öğesi *Adı* şeklinde, *c* harfi ile görünen *AutoSize* öğesi ise *True* olacak şekilde düzenlenir. Özetle *A-a* lbl_adi, *A-b* Adı, *A-c* True yapılır. Kullanıcı formu üzerinde *C, E, G* ile kodlanan *Label* nesneleri de Şekil 6.7 görüldüğü gibi düzenlenir. Şekle göre; *C* ile kodlanan *Label* için *C-d* lbl_soyadi, *C-e* Soyadı, *C-f* True şeklinde; E ile kodlanan *Label* için *E-g* lbl_bolumu, *E-h* Bölümü, *E-i* True şeklinde, G ile kodlanan *Label* için *G-j* lbl_hobileri, *G-k* Hobileri, *G-l* True şeklinde düzenlenir.

Özellik (Properties) öğelerinde değişiklik yapılabilmesi için her seferinde ToolBox üzerinde Select Objects (1) nesnesine tıklanması gereklidir.

Şekil 6.7

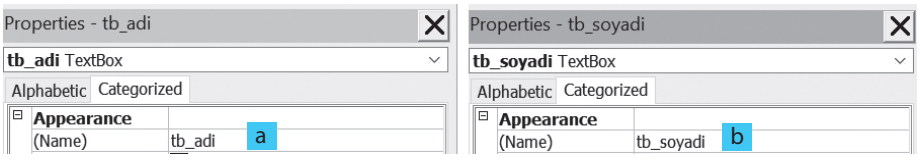
Örnek 1 Properties Penceresi Label İşlemleri



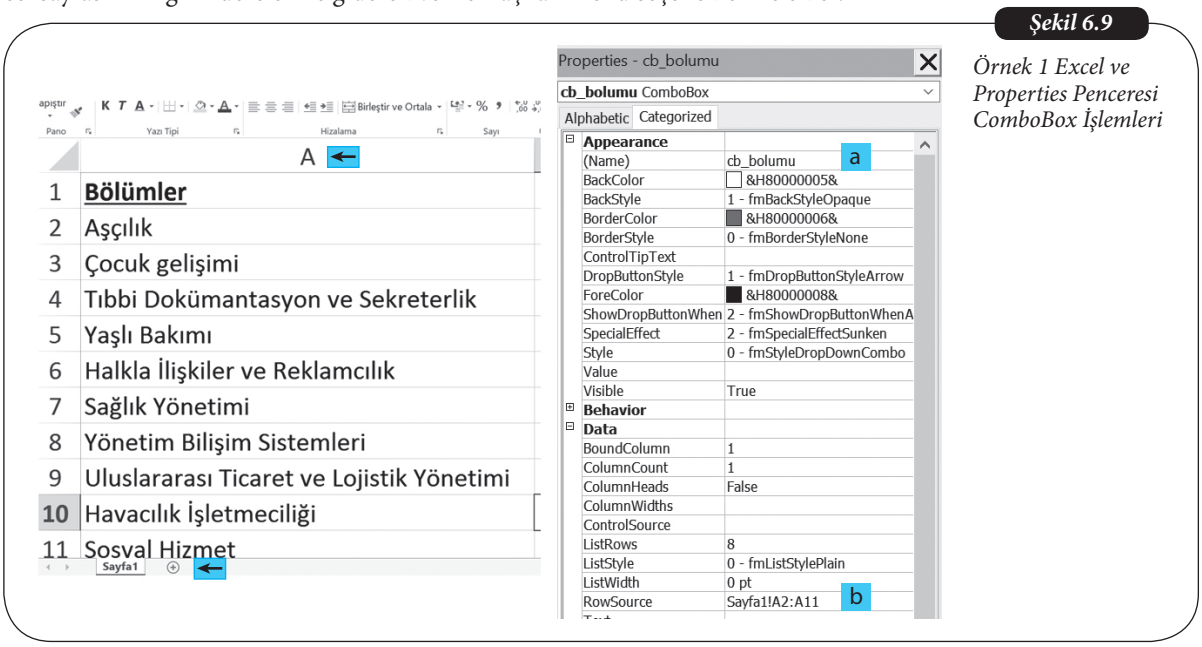
Dördüncü aşamada kullanıcı formu üzerinde *TextBox* kontrol nesnelerinin özelliklerinin ayarlanması için *Properties* penceresinde işlem yapılır. Daha önce tasarladığımız Şekil 6.6'da görülen Form üzerindeki B ve D harfleri ile kodlanan *TextBox*'larda, Şekil 6.8'de görüldüğü gibi *B-a* tb_adi, *D-b* tb_soyadi olacak şekilde isimlendirilir.

Şekil 6.8

Örnek 1 Properties Penceresi TextBox İşlemleri



Beşinci aşamada açılan bir menü şeklinde bölümlerin sorulacağı *ComboBox* tasarımı yapılacaktır. *ComboBox* içerisindeki veriler VBA kodları ile *AddItem* komutu ile eklenebilir veya Excel Hücrelerindeki belirli bir aralıktan aldırılabilir. Örnekte, Şekil 6.9'da ok işaretleri ile gösterildiği gibi Excel sayfasında Sayfa1'de A2'den A11'e kadar bölüm isimleri yazılıdır. VBA sayfasında *UserForm* üzerindeki ComboBox (Şekil 6.6'da F harfi ile gösterilmişti) kontrol nesnesi seçilerek *Name* öğesi cb_bolumu *(F-a)* şeklinde isimlendirilir. Açılan menünün Excel sayfasındaki A2'den A11'e kadar yazılı bölümler ile *ComboBox* arasında bir köprü oluşturmak için *RowSource* öğesi kullanılarak *Sayfa1!A2:A11* yordamı özelliğe yazılır (F-b) Köprüleme işleminden sonra, VBA yazılımı, her çalıştırıldığında Excel sayfasının ilgili hücrelerine giderek verileri açılan menü seçeneklerine ekler.



Şekil 6.9

Örnek 1 Excel ve Properties Penceresi ComboBox İşlemleri

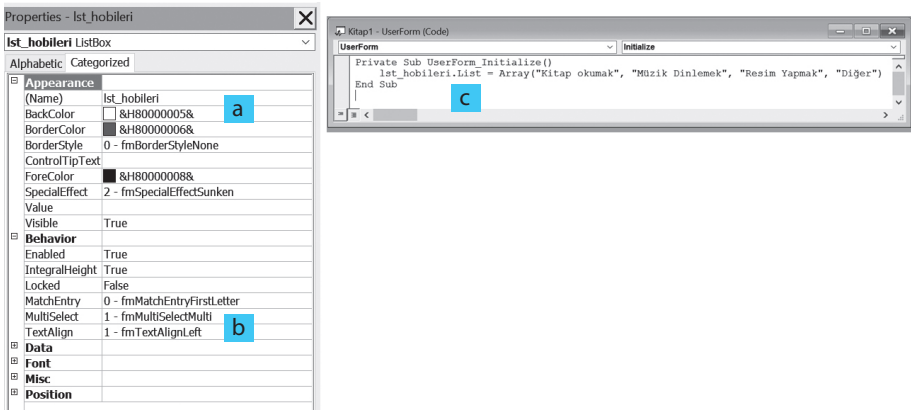
Altıncı aşamada öğrenci hobilerinin yazılabileceği Şekil 6.6'da *H* harfi ile kodlanmış *ListBox* ile ilgili işlem yapılacaktır. *a* harfi ile kodlanan *Name* öğesi *lst_hobileri* şeklinde düzenlenir. *ListBox* içerisinden birden fazla seçim yapılabilecek liste sunabilmesi için Şekil 6.10'da *b* harfi ile kodlanan *MultiSelect* öğesi *1-fmMultiSelectMulti* seçilir. Kullanıcı formu çalıştırdığında *ListBox* nesnesinin içerisinde göreceği hobi seçeneklerin *beşinci aşamada* olduğu gibi hücrelerden alınabilir veya VBA kod kısmına geçilerek Şekil 6.10 *c*'de görüldüğü gibi yazılımda dizi şeklinde hazırlanmış bir karakter katarı *ListBox* ile köprülenebilir. Köprüleme işleminin başlatılması için bir tetikleyici gereklidir. Fonksiyonun çalıştırılması için formun yüklenmesi *Initialize()* alınabilir. Bunun için hazırlanacak fonksiyon ve yordam aşağıdaki gibidir:

```
Private Sub UserForm_Initialize()
lst_hobileri.List = Array("Kitap okumak", "Müzik Dinlemek", "Resim Yapmak", "Diğer")
End Sub
```

Oluşturulan kod kullanıcı yüklenirken *lst_hobileri* şeklinde isimlendirilmiş *ListBox* içerisindeki listeye tırnak işaretleri ve virgüllerle ayrılmış karakter katarını atar.

Şekil 6.10

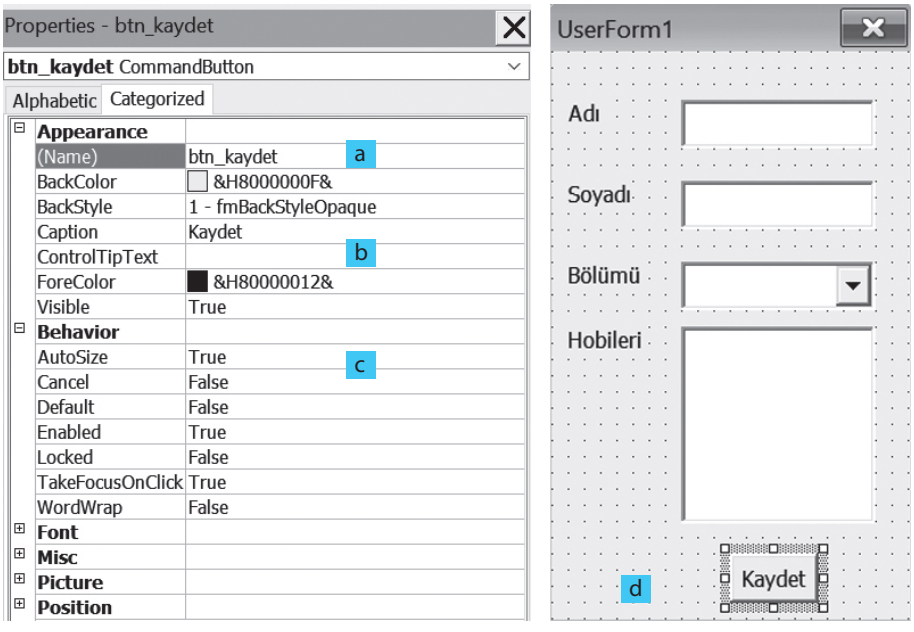
Örnek 1 Excel ve Properties Penceresi ListBox İşlemleri



Yedinci aşamada Şekil 6.6'da *I* harfi ile gösterilen *CommandButton* nesnesinin özellikleri ve VBA kodları hazırlanacaktır. *CommandButton* üzerine yüklenen işlemi yapması için kullanılan bir butondur. Tasarlanan anketin Excel sayfalarına yazdırılmasında *CommandButton* nesnesi bir tetikleyici olarak kullanılacaktır. Bu butona basan kişinin bilgileri Excel sayfasına yazdırılması için hazırlanan VBA kodunu çalıştıracaktır. Öncelikle *CommandButton* özelliklerini kullanarak butonun hangi işlevi olduğunun kullanıcıya aktarılması gerekir. Bunun için Şekil 6.11 butonun ismi *btn_kaydet* şeklinde *(bkz. a)*, formda görünümü için başlığı *Kaydet* şeklinde *(bkz. b)* ve son olarak görsel olarak büyük olmaması için *AutoSize True* şeklinde *(bkz. c)* düzenlenmiştir. Tasarlanan kullanıcı formunun son hâli *d* harfi ile gösterildiği gibi olmuştur.

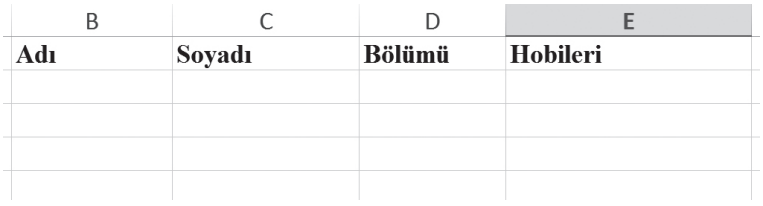
Şekil 6.11

Örnek 1 Excel ve Properties Penceresi CommandButton İşlemleri



Sekizinci aşamada kaydet butonun işlevi tasarlanacaktır. Butona tıklandığında buton bir dizi fonksiyonu çalıştırarak kullanıcı formundaki bilgileri Excel hücrelerine aktaracaktır. Excel sayfasındaki hücreler için Şekil 6.12'de görüldüğü gibi tasarım yapılmıştır.

Şekil 6.12



B	C	D	E
Adı	**Soyadı**	**Bölümü**	**Hobileri**

Örnek 1 Excel Hücreleri Tasarımı

Dozkuncu aşamada VBA kodlarına geçilerek *btn_kaydet* isimli *CommandButton* fonksiyonları hazırlanacaktır. Bunun için bir ana fonksiyon ve 2 alt fonksiyon tasarlanacaktır. Alt fonksiyonlardan birincisi Excel hücrelerinde dolu satırları kontrol ederek ilk boş satırı bulma fonksiyonudur. Excel sayfasında B sütununda *Ctrl* + *Shift* + *End* tuşuna basma işlemi VBA kodu kullanılarak yapılacaktır. Böylece kullanıcı formundan alınan bilgiler alt alta boş satır olmadan kayıt edilebilecektir.

Şekil 6.13

```
Public Function son_satiri_bul() As Integer

son_satiri_bul = ActiveSheet.Cells(ActiveSheet.Rows.Count, "B").End(xlUp).Row + 1

End Function
```

Örnek 1 VBA'da son_satiri_bul Fonksiyonu

Alt fonksiyonlardan ikincisi *ListBox* içerisinden seçilen hobileri tek satıra dönüştüren ve *String* formatında çıktı üreterek çıktı üreten bir fonksiyondur. *seçilen_hobileri_bul* fonksiyonu Şekil 6.14'te a ile gösterilen kısımda *ListBox* içerisinde kullanıcı tarafından ilk seçilen veri fonksiyondan geri dönüş için hazırlanır. *ListBox* nesnesi içerisinde kullanıcı tarafından seçilen diğer veriler için *For Next* döngüsü kullanılmıştır. İlk veriden sonra seçilen verilerin aralarına virgül konularak veriler bir birlerinin arkasına eklenir.

Şekil 6.14

```
Public Function secilen_hobileri_bul() As String

secilen_hobileri_bul = CStr(lst_hobileri.List(0))

For i = 1 To lst_hobileri.ListCount - 1
          If lst_hobileri.Selected(i) Then   b
              secilen_hobileri_bul = secilen_hobileri_bul + ", " + CStr(lst_hobileri.List(i))   c
          End If
Next i

End Function
```

Örnek 1 VBA'da seçilen_hobileri_bul Fonksiyonu

btn_kaydet isimli butona basıldığında VBA kodunun tetiklenmesi için *btn_kaydet_Click()* fonksiyonu oluşturulmuştur. Şekil 6.15'te a bölümünde kullanıcı formundan alınacak verilerin konulması için değişken paketleri hazırlanır. b ile gösterilen bölümde VBA'daki hazır fonksiyonlardan *IsEmpty()* komutuyla formun içerisindeki verilerin boş olup olmadığı kontrol edilir. *ListBox*'ta kullanıcının seçim yapıp yapmadığı ise *ListBox* nesnesinde seçilen satır sayısı 0'a eşit ise *(lst_hobileri.ListCount=0)* deyimi ile kontrol edilir. Eğer form eksik doldurulduysa bir mesaj kutusu çıkarak *(MsgBox)* uyarı verecektir. Form tam olarak doldurulduysa *c* ile görüldüğü gibi formda yazılan veriler *a* bölümünde hazırlanan değişken paketlerinin içerisine konulacaktır.

Şekil 6.15

Örnek 1 VBA'da seçilen_hobileri_bul Fonksiyonu

```
Private Sub btn_kaydet_Click()

Dim adi As String
Dim soyadi As String
Dim bolumu As String
Dim hobileri As String
Dim son_satir As Integer

If IsEmpty(tb_adi.Value) = True Or IsEmpty(tb_soyadi.Value) = True Or _
   IsEmpty(cb_bolumu.Value) = True Or lst_hobileri.ListCount = 0 Then
     MsgBox "Lütfen Tüm Formu Doldurun"
Else

adi = CStr(tb_adi.Value)
soyadi = CStr(tb_soyadi.Value)
bolumu = CStr(cb_bolumu.Value)
hobileri = secilen_hobileri_bul()

son_satir = son_satiri_bul()

ActiveSheet.Cells(son_satir, 2).Value = adi
ActiveSheet.Cells(son_satir, 3).Value = soyadi
ActiveSheet.Cells(son_satir, 4).Value = bolumu
ActiveSheet.Cells(son_satir, 5).Value = hobileri

tb_adi.Value = ""
tb_soyadi.Value = ""
cb_bolumu.Value = Null

For i = 0 To lst_hobileri.ListCount - 1
          lst_hobileri.Selected(i) = False
Next i

End If
End Sub
```

a b c d e f

Fonksiyonlardan *secilen_hobileri_bul* fonksiyonu bu aşamada çalıştırılır ve *hobileri* değişken paketine formda seçilen verilerin aktarılması sağlanır. *d* bölümünde *son_satiri_bul* fonksiyonu çalıştırılarak son satır numarası *son_satir* isimli değişken paketinin içerisine atılır. *e bölümünde* Şekil 6.12'de görülen Excel sayfalarının içerisine değişken paketlerinin içerisindeki veriler yazdırılır. Son olarak f aşamasında kullanıcı formu içerisindeki metin kutuları, açılan kutu ve liste kutusu temizlenerek yeni veri girişi için kullanıcı formu hazır hâle getirilir.

Kontrol nesnelerinin 2. bölümünde *CheckBox, OptionButton, ToggleButton, Frame, CommandButton* nesnelerinin kullanımı Tablo 6.6'da verilmiştir.

Tablo 6.6
Kullanıcı Formu Kontrolleri 2. Bölüm

KULLANICI FORMU KONTROLLERİ 2. BÖLÜM

CheckBox
Simgesi ile gösterilir. Kullanıcı tarafından *Evet-Hayır, Açık-kapalı* gibi iki değerler arasında seçim yapması için *CheckBox* kullanılır.

OptionButton
Simgesi ile gösterilir. Kullanıcı tarafından bir grup veri içerisinden bir tane veri seçecek form tasarımı yapılırken *OptionButton* kullanılır. Çoktan seçmeli sorularda seçenekler Excel hücrelerinden veya VBA kodu ile atanabilir.

ToggleButton
Simgesi ile gösterilir. Bir maddenin seçilip seçilmediğini *ToggleButton* nesnesi ile gösterilebilir. Aç-Kapa, Evet-Hayır, Doğru-Yanlış gibi ikili verilerden birinin seçilmesinde kullanılabilir.

Frame
Simgesi ile kullanılır. *Frame* nesnesi kullanıcı formlarında grup oluşturmak için kullanılır. Bir tür çerçeve olan *Frame* nesnelerin gruplandırılmasında da kullanılabilmektedir.

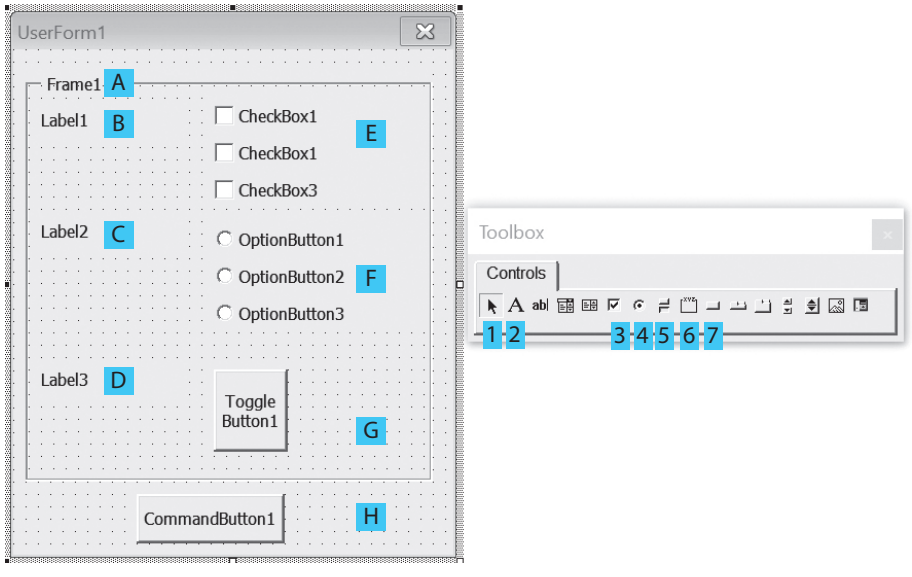
CommandButton
Simgesi ile kullanılır. Makro veya olay prosedürünü tetiklemekte kullanılır. VBA'da hazırlanmış fonksiyon ve yordamlara komut vermek için kullanılır. Varsayılan olarak *Click* olayı ile kullanılır.

Önceki bölümde VBA'da hazırlanan anket örneğinin devamı niteliğinde bir örnek hazırlanacaktır. Örnek 1'deki anket hobiler arasında kitap okumak seçilmesi durumunda doldurulacak anketin devamıdır. Geliştirilen örnekte *CheckBox, Frame, OptionButton, ToggleButton, CommandButton* ve *Label* kullanılmıştır. Şekil 6.16'da *UserForm* üzerinde tıklanacak yerler bir önceki örnekte olduğu gibi alfabetik ve *ToolBox* menüsünde eklenecek kontrol nesneleri ise sayı ile kodlanmıştır.

Birinci aşamada sırasıyla önce *ToolBox* üzerinde 6 numaralı *Frame* kontrol nesnesine sonra *UserForm* üzerinde A harfi ile kodlanan yere tıklanarak form üzerinde bir çerçeve oluşturulur (6-A). *Label* kontrol nesnesi eklemek için sırasıyla 2-B, 2-C, 2-D işlemleri yapılır. *CheckBox* eklemek için 3-E işlemi ve *OptionButton* için 4-F işlemi 3 defa tekrarlanır. Böylece 3 seçenekli *CheckBox* ve *OptionButton* oluşturulabilir. Seçenek sayısının arttırılması istenirse işlemin tekrar sayısı arttırılır. *ToggleButton* için *ToolBox* üzerinde 5, *UserForm* üzerinde G harfi ile kodlanan yere tıklanır. Son olarak VBA yazılımda kodların tetiklenebilmesi için 7-H işlemi yapılır. Kontrollerin yerinin ayarlanması için *ToolBox* üzerinde 1 numara ile kodlanan Select Objects tıklanır ve düzenlenmek istenen kontrol nesnesinin yeri ayarlanabilir.

Şekil 6.16

Örnek 2 Araç Kutusu Kontrolleri

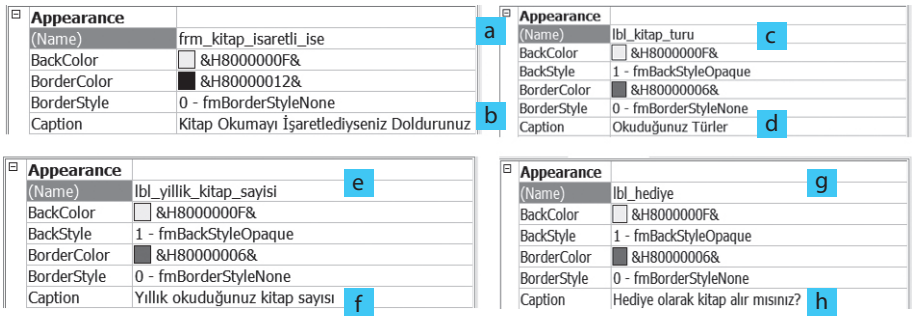


İkinci aşamada Frame ve Label türündeki kontrol nesne özelliklerinin düzenlenmesi için *Proporties* penceresinde işlem yapılır. Şekil 6.16'da *UserForm* üzerinde *A* harfi ile kodlanmış *Frame* özelliklerinin Şekil 6.17 *a* harfi ile görünen *name* öğesi *frm_kitap_isaretli_ise* şeklinde, *b* harfi ile görünen *Caption* öğesi *Kitap Okumayı İşaretlediyseniz Doldurunuz* şeklinde düzenlenir. Özetle A-a *frm_kitap_isaretli_ise,* A-b *Kitap Okumayı İşaretlediyseniz Doldurunuz* yapılır. Kullanıcı formu üzerinde B, C, D ile kodlanan *Label* nesneleri de Şekil 6.17'de görüldüğü gibi düzenlenir. B ile kodlanan *Label* için B-c *lbl_kitap_turu,* B-d *Okuduğunuz Türler* şeklinde; C ile kodlanan *Label* için C-e *lbl_yillik_kitap_sayisi,* C-f *Yıllık Okuduğunuz Kitap Sayısı* şeklinde, D ile kodlanan *Label* için D-g *lbl_hediye,* D-h *Hediye Olarak Kitap Alır mısınız?* şeklinde düzenlenir.

Özellik (Properties) öğelerinde değişiklik yapılabilmesi için her seferinde ToolBox üzerinde Select Objects (1) nesnesine tıklanması gereklidir.

Şekil 6.17

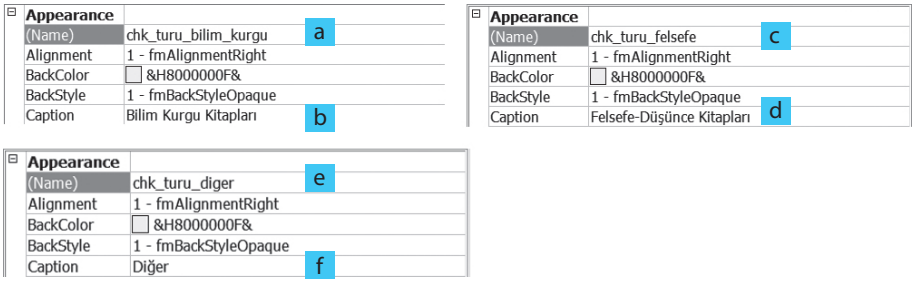
Örnek 2 Properties Penceresi Label İşlemleri



Üçüncü aşamada CheckBox türündeki kontrol nesnelerinin özellikleri ayarlanır. Şekil 6.16'da *UserForm* üzerinde E ile kodlanan 3 *CheckBox* kontrol nesnesinin özellik öğeleri Şekil 6.18'deki gibi düzenlenir. *Name* öğeleri *chk_turu_bilim_kurgu (bkz. a), chk_turu_felsefe (bkz. c)* ve *chk_turu_diger (bkz. e)* şeklinde düzenlenir. Caption öğeleri ise *Bilim Kurgu Kitapları (bkz.b), Felsefe-Düşünce Kitapları (bkz.d)* ve *Diğer (bkz.f)* şeklinde ayarlanır.

Şekil 6.18

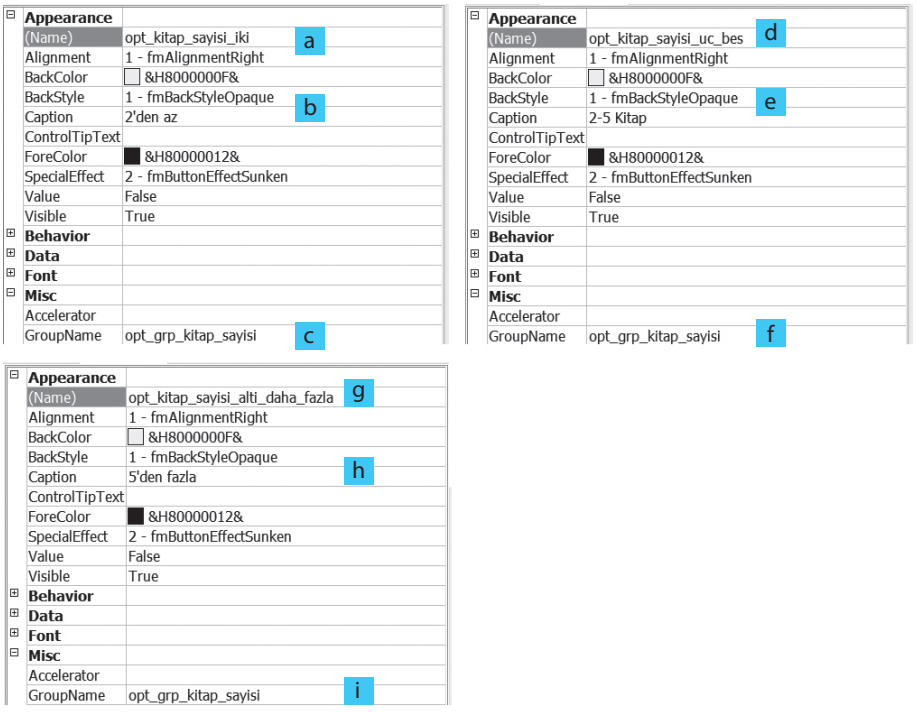
Örnek 2 Properties Penceresi CheckBox İşlemleri



Dördüncü aşamada kullanıcıdan yıllık anket sayısını öğrenmek için kullanılan Şekil 6.16'da F harfi ile kodlanan *OptionButton* kontrol nesnesinin özellik öğeleri ayarlanacaktır. Kullanıcı *OptionButton* seçeneklerinden sadece birini seçilebileceği çoktan seçmeli bir yapı kurmak için *GroupName* özelliği kullanılmaktadır. Şekil 6.19'da *c, f, e* harfleri ile gösterildiği gibi üç nesne için *GroupName* öğesi *opt_grp_kitap_sayisi* şeklinde, *Name* ve *Caption* öğeleri de *a, b, d, e, g, h* harfleri görüldüğü gibi düzenlenmiştir.

Şekil 6.19

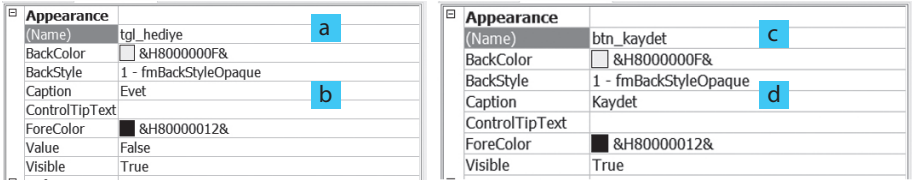
Örnek 2 Properties Penceresi OptionButton İşlemleri



Beşinci aşamada Şekil 6.16'da G harfi ile gösterilen *ToggleButton* ve H harfi ile gösterilen *CommandButton* kontrol nesnelerinin özellik öğeleri düzenlenmiştir. Şekil 6.20'de *ToggleButton* nesnesi *Name* öğesi *tgl_hediye* şeklinde *(bkz. a), Caption* öğesi ise *Evet* şeklinde girilmiştir *(bkz. b). CommandButton* nesnesi *Name* özellik öğeleri ise *btn_kaydet (bkz. c)* ve *Caption* öğesi ise *Kaydet* şeklinde düzenlenmiştir.

Şekil 6.20

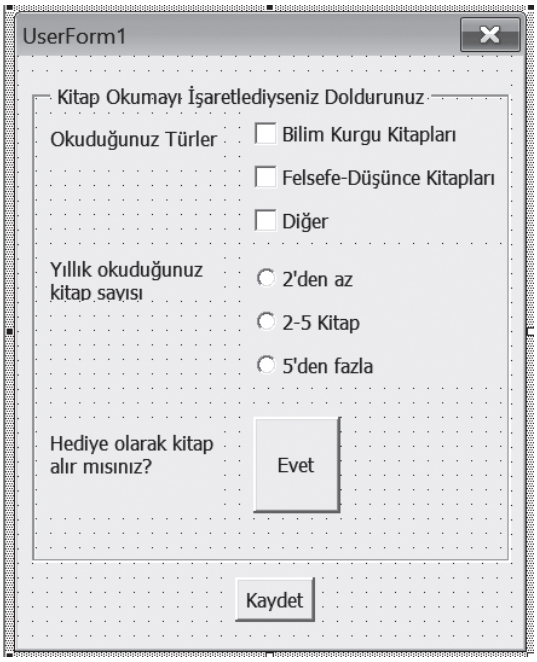
Örnek 2 Properties Penceresi ToggleButton İşlemleri



Altıncı aşamada Şekil 6.21'de görüldüğü gibi *UserForm* tasarımı nesneleri uygun şekilde sıralanır ve hizalanır.

Şekil 6.21

Örnek 2 UserForm Tasarım İşlemleri



Altıncı aşamada VBA yazılım kodlarına geçilerek *Kaydet* butonuna basıldığında tetiklenecek fonksiyon ve yordamlar hazırlanır.

Şekil 6.22'de *a* ile kodlanan bölümde *turler, kitap_sayisi* ve *hediye* olarak isimlendirilen *String* tipinde değişken paketleri hazırlanır. Ayrıca kullanıcı formlarına eklenen kontrollerin tek seferde ve hızlı işlem yapılması için *MSForms.Control* türünde *ctrl* isminde bir değişken paketi de tanımlanmıştır.

Şekil 6.22'de *b* harfi ile gösterilen bölümde okunan kitapların sorulduğu 3 *CheckBox* nesnesinin sonuçlarının değişken paketlerine aktarılma yöntemlerinden biri gösterilmiştir. Koşul fonksiyonu *(If…Then)* kullanılarak *CheckBox* işaretlendiyse daha önceki bölümde tanımlanan *turler* verisi değişken paketine aktarılır. Aktarma sırasında kullanıcı formunda görsel başlıkların verilmesinde kullanılan *Caption* öğesinden faydalanılabilir. *CheckBox* nesnelerinin kullanıcı formu üzerinde görülen başlıkları *(Caption)* değişken paketlerine aktarılmıştır. Böylece aynı verilerin tekrar edilmesine gerek kalmamıştır.

Kullanıcı formunda yıllık okunan kitap verisinin alındığı *OptionButton* işlemleri Şekil 6.22'de *c* harfi ile gösterilmiştir. *b* harfi ile gösterilen *CheckBox* nesnesinde her bir nesne için farklı koşul fonksiyonu kullanılmıştır. Özellikle çok fazla seçeneğin olduğu işlemlerde her bir seçenek için işlem yapmak VBA'da kodlanan satır sayısını artırmaktadır. Bu nedenle *c* harfi ile gösterilen *OptionButton* işlemlerinde Şekil 6.16'da A harfi ile kodlanan *Frame* nesnesinin içerisindeki tüm kontroller tek seferde *ctrl* değişken paketine atılmıştır. Bu işlem için *For Each* döngüsü kullanılarak *ctrl* değişken paketine *frm_kitap_isaretli_ise Controls* deyimi kullanılmıştır. Frame içerisindeki kontrollerden *OptionButton* nesneleri koşul ve *TypeName()* fonksiyonları kullanılarak seçilmiştir. Grup ismi *opt_grp_kitap_sayisi* olan ve son kullanıcı tarafından işaretlenmiş olan *OptionButton* nesnelerinin başlığı bir başka değişle *kitap_sayisi* verisi değişken paketine atılmıştır. Tek bir döngü ile yapılan işlem karmaşık olmasına rağmen *CheckBox* nesnesindeki verilerin aktarılmasına göre *(bkz. b)* daha az satırda hazırlanmıştır. *OptionButton* verilerinin aktarılmasında kullanılan yapıda ihtiyaca bağlı olarak mevcut *OptionButton* seçeneklerinde değişiklik yapılması veya yeni *OptionButton* eklenmesi durumunda VBA kodlarında değişiklik yapılmasına gerek kalmaz. Çünkü kod esnek bir yapıdadır.

Şekil 6.22'de *d* harfi ile gösterilen bölümde kullanıcı formunda *Hediye olarak kitap alır mısınız?* sorusunun yanıtı *ToggleButton* kontrol nesnesine verilen cevap değişken paketine aktarılmıştır. Bunun için koşul fonksiyonu kurularak soruya verilen cevaba bağlı olarak *hediye* değişken paketinin içerine *Evet* veya *Hayır* verilerinden biri atanmıştır.

Şekil 6.22

Örnek 2 VBA Kaydet Fonksiyonu

```
Private Sub btn_kaydet_Click()
Dim turler As String
Dim kitap_sayisi As String                    a
Dim hediye As String
Dim ctrl As MSForms.Control

If chk_turu_bilim_kurgu.Value = True Then
   turler = chk_turu_bilim_kurgu.Caption + " "
End If
If chk_turu_felsefe.Value = True Then
   turler = turler + chk_turu_felsefe.Caption + " "
End If                                                    b
If chk_turu_diger.Value = True Then
   turler = turler + chk_turu_diger.Caption + " "
End If

For Each ctrl In frm_kitap_isaretli_ise.Controls
    If TypeName(ctrl) = "OptionButton" Then
        If ctrl.GroupName = "opt_grp_kitap_sayisi" And ctrl.Value = True Then   c
            kitap_sayisi = ctrl.Caption
        End If
    End If
Next

If tgl_hediye.Value = True Then
hediye = "Evet"
Else
hediye = "Hayır"                              d
End If
                                  e
son_satir = son_satiri_bul()
ActiveSheet.Cells(son_satir, 6).Value = turler
ActiveSheet.Cells(son_satir, 7).Value = kitap_sayisi   f
ActiveSheet.Cells(son_satir, 8).Value = hediye

For Each ctrl In frm_kitap_isaretli_ise.Controls
    If TypeName(ctrl) = "OptionButton" Or TypeName(ctrl) = "CheckBox" Or   g
    TypeName(ctrl) = "ToggleButton" Then
        ctrl.Value = False
    End If
Next
```

Şekil 6.22'de *e* harfi ile gösterilen bölümde *Örnek 1*'de Şekil 6.13'te açıklanan *son_satiri_bul()* fonksiyonu kullanılarak son satır bilgisi *son_satir* değişken paketine atılmıştır.

Şekil 6.22'de *f* harfi ile gösterilen bölümde ise önceki bölümlerde *turler, kitap_sayisi* ve *hediye* değişken paketlerine atılmış veriler Excel hücrelerine yazdırılmıştır.

Excel hücrelerine aktarıldıktan sonra kullanıcı formu yeni kayıt girmesi için verilerin temizlenmesi gerekmektedir. Şekil 6.22'de *g* harfi ile gösterilen bölümde formun ilk çalıştırıldığı hale dönüştürülmesi için *CheckBox, OptionButto*n ve *ToggleButton* nesnelerinin üzerinde işaretlemelerin kaldırılması gerekir. Veri temizleme işlemi için *For Each* döngüsü ve *ctrl* değişken paketi kullanılmıştır.

Kontrol nesnelerinin 3. bölümünde *TabStrip, Multipage, ScrollBar, SpinButton, Image, RefEdit* nesnelerinin kullanımı Tablo 6.6'da verilmiştir.

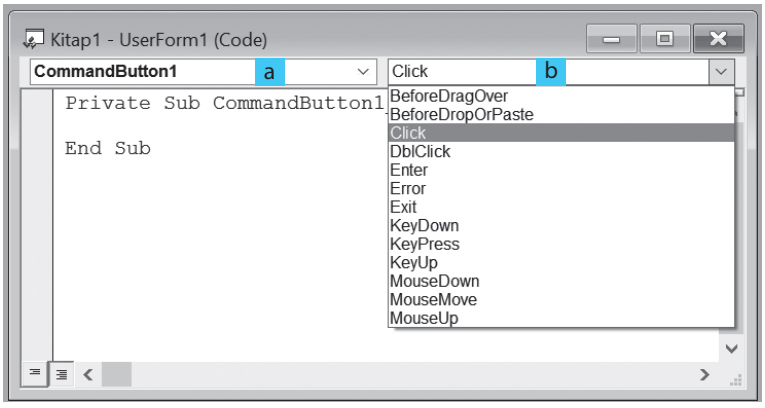
Tablo 6.7
Kullanıcı Formu Kontrolleri 3. Bölüm

KULLANICI FORMU KONTROLLERİ 3 BÖLÜM	
TabStrip	Simgesi ile gösterilir. *TabStrip* kontrol nesnesini kullanarak farklı özellikteki veriler için bir tür fihrist oluşturmada kullanılır.
MultiPage	Simgesi ile gösterilir. Tek bir kontrol nesnesi ile birden fazla sayfa yapılmasında kullanılır. Büyük sayıda veri ile çalışırken bilgilerin çeşitli kategorilere göre sınıflandırılmasına yarar.
ScrollBar	Simgesi ile gösterilir. *ListBox* gibi birden fazla satırlı nesneler ile beraber kullanılır. Kaydırma çubuğudur.
SpinButton	Simgesi ile kullanılır. Sayısal verilerin kademeli olarak fare tuşu ile arttırılmasında veya azaltılmasında kullanılır. Örneğin takvimlerde bir sonraki aya veya güne geçmek için *SpinButton* kullanılır.
Image	Simgesi ile kullanılır. Formun içerisinde resim koymak için kullanılan nesnedir. Örneğin personel kayıtlarının tutulmasında kullanılan bir kullanıcı formunda personelin resmi forma *Image* nesnesi ile bağlanabilir.
RefEdit	Simgesi ile kullanılır. 2013 ve sonrasında kullanılan bir nesnedir. Kullanıcı formunda bir blok üzerinde işlem yapmada kullanılır.

KULLANICI FORMU NESNE OLAY İLİŞKİSİ

Kullanıcı formları üzerine eklenen kontrol nesnelerinin VBA yazılımındaki fonksiyon ve alt yordamlara bağlantısı olaylar ile sağlanır. Örneğin, kullanıcı formuna konulmuş bir *CommandButton* nesnesi VBA'da tanımlanmış bir fonksiyonu fare üzerine geldiğinde, tıklandığında, çift tıklandığında, klavyede belirli *Enter* tuşuna basıldığında vb. durumlarda çalıştırabilir. Tasarlanan kullanıcı formları ile VBA'daki fonksiyonlar arasında etkileşim ve etkileşimin başlama biçimi olaylar ile belirlenir. Bu anlamda olaylar bir tür tetik mekanizması görevi görür. Şekil 6.23'te kontrol nesneleri *(bkz. a)* ile VBA fonksiyonları arasındaki ilişkiyi sağlayan olay tanımlanması *(bkz. b)* işlemi gösterilmiştir.

Şekil 6.23
Kullanıcı Formu Olay İlişkisi



Şekil 23'te *b* ile gösterilen olaylar nesnelerin özelliklerine göre değişiklik gösterir. Örneğin, *CommandButton* bir tür komut düğmesidir. Bu komut düğmesine kullanıcı klavyeden karakter yazmaz. Kullanıcı fare veya klavye ile seçer veya tıklar. Bu nedenle komut düğmelerinde *Change* olayı bulunmaz.

Tablo 6.8

OLAYLAR	
	Change Kontrol nesnelerinde değişiklik olması durumunda VBA'daki fonksiyon *Change* olayı ile tetiklenir. Değişiklik olma durumu kontrol nesnelerine göre farklılaşır. Bir başka deyişle *Change olayı* kontrol nesnelerine bağlı olarak farklı koşullarda VBA fonksiyonlarını tetikler. Örneğin, *TextBox* veya *ComboBox* gibi bir metin kutusundaki değişiklik durumunda tanımlanmış olayın başlangıcı sağlanır. Bunun yanında *ComboBox* ve *ListBox* nesnesinde yeni bir seçenek seçildiğinde olay tetiklenir. *ScrollBar* kontrol nesnesinde kaydırma çubuğunun hareketi, *MultiPage* kontrol nesnesinde ise farklı sayfaya geçiş *Change* olayını tetikler.
	Click Bilgisayarın temel çevre bileşenlerinden biri faredir. Kontrol nesnelerinin üzerine fare ile tıklandığında *Click* olayı başlar. Birçok kontrol nesnesinde varsayılan olay olarak *Click* gelir. *CommandButton, Frame, Imange, Label, ScrollBar* ve *SpinButton* ile kullanılabilir.
	DblClick Kullanıcı formunda tanımlanmış bir kontrol nesnesinin üzerinde çift tıklanınca VBA fonksiyonu tetiklemek isteniyorsa *DblClick* olayı kullanılır. Çift tıklama arasındaki zaman boşluğu kullanılan bilgisayarın sistem ayarlarındaki çift tıklama hızı ayarları (double-click speed setting) ile yapılır.
	Enter, Exit *Enter* olayı kontrol nesnesine girilmesi ile tetiklenir. Örneğin, bir form üzerindeki *CheckBox* kontrol nesnesi işaretlendiğinde *Enter* olayı tetiklenir. Form üzerindeki farklı bir nesneye geçildiğinde *Exit* olayı tetiklenir.
	KeyDown, KeyUp Kullanıcı kontrol nesnesi üzerinde bir tuşa bastığında *KeyDown* olayı tetiklenir. Tuşu bıraktığında ise *KeyUp* nesnesi tetiklenir. Genellikle bir birlerinin alternatifi olarak kullanılırlar. Özellikle *Home, End* gibi fonksiyon tuşları ile çalışacak şekilde düzenlenirler.
	KeyPress Kullanıcı belirli tuşlara bastığında tetiklenen olaylar *KeyPress* ile hazırlanır. *Ctr* tuşu ile beraber hazırlanan kısayol tuşları *KeyPress* olayına örnek olarak verilebilir.

Kullanıcı formunda kullanılan nesnelerde yer alan diğer olaylara https://msdn.microsoft.com/en-us/library/office/jj692788.aspx adresinden ulaşılabilir.

Özet

Kullanıcı formlarının nasıl oluşturulduğunu tanımlamak.
Kullanıcı formal bir yapıda verilerin girilmesini istiyor ise form tasarlayabilir. Kullanıcı tasarlanan form ile verileri Excel sayfalarından alabilir veya VBA yazılımında fonksiyonların ürettiği sonuçları Excel hücreleri ve/veya kullanıcı formlarına yazdırabilir. Kullanıcı formu eklemek için öncelikle Excel sayfasında Geliştirici sekmesindeki Visual Basic butonuna tıklamalı veya alt ve F11 kısa yol tuşları kullanılarak VBA yazılım platformuna geçmelidir. *Insert* sekmesine tıklandığında menü açılacaktır. Açılan menü içerisinden *UserForm* seçeneğine tıklanır. VBA yazılımı; *UserForm, Toolbox* ve *Properties* pencerelerini varsayılan olarak açar.

Kullanıcı formlarındaki kontrol nesnelerinin özelliklerini açıklamak.
Kullanıcı formunun tasarım aşamasında görsel düzenlemeler yapılabilmesi için kullanıcı formu özellikleri penceresi kullanılır. Kullanıcı formunda yer alan özelliklerin sınıflandırılmış içeriğinde ilk olarak görünüm (Appearance) kategorisi yer alır. Görünüm kategorisi içerisindeki *Name, BackColor, BorderColor, BorderStyle, ForeColor, Caption, ControlTip Text, PasswordChar, SpecialEffect, Value ve Visible* öğeleri sıklıkla kullanılan öğelerdir. VBA form ve kontrollerinin çeşitli koşullar altında nasıl bir davranış sergileyeceği Davranış (Behavior) kategorisinde yer alır. Davranış kategorisi altında yer alan öğelerden en sık kullanılanları; *AutoSize, Auto Tab, AutoWordSelect, MaxLength, Cycle, Enabled, Locked, MatchEntry, MatchRequired, TextAlign* dır. Sadece veri girişine yarayan *TextBox, ComboBox, ListBox, CheckBox, OptionButton, ToggleButton* gibi kontrollerde veri kategorisi (Data) bulunur. Bu kategori altında *BoundColumn, ColumnCount, ColumnHeads, ColumnWidths, ControlSource, ListRows, ListStyle, ListWidth, RowSource, Text* öğeleri yer alır. Yazı Kategorisi (Font) altında sadece aynı isimli *Font* öğesi vardır. Kullanıcı formu ve kontrolleri içerisine resim eklemek için Resim (Picture) Kategorisi Öğeleri ile işlem yapılır. Kullanıcı kontrollerinin kullanıcı formu içerisindeki pozisyonunun belirlenmesi için Pozisyon *(Position)* Kategorisi Öğeleri kullanılır. Pozisyon kategorisi öğeleri; *Height, Width, Left ve Top* öğeleridir. Kullanıcı formunda kaydırma çubuğu özelliklerinin belirlenmesi için Kaydırma (Scrolling) Çubuğu Kategorisi Öğeleri kullanılır. Kullanıcı formu ve kontrolleri içerisinde bu sınıflar arasına konulamayan diğer öğeler için Diğer Kategori Öğeleri (Misc.) oluşturulmuştur. Sık kullanılan Diğer kategori öğeleri *DrawBuffer, MouseIcon, MousePointer, TabIndex, TabStop, Tag, WhatsThisButton, WhatsThisHelp* öğeleridir.

Kullanıcı formu kontrol nesnelerini tanımak.
Kullanıcı formlarının işlevsel tasarımında araç kutusu (Toolbox) penceresi içerisinde yer alan kontrol menüsü (Controls) ile kullanılır. Kontrol menüleri formların içeriğinin belirlenmesinde kullanılan temel araçlardan sadece biridir. Kontrol formu içerisinde *Select Objects, Label, TextBox, ComboBox, ListBox, CheckBox, OptionButton, ToggleButton, Frame, CommandButton, TabStrip, Multipage, ScrollBar, SpinButton, Image, RefEdit* öğeleri yer alır. Select Objects; kullanıcı formunun üzerindeki kontrollerin özellik menüsünün aktif olabilmesi için Select Objects ile seçilmesi gerekir.
Label; Sadece tasarım ekranında düzenlenebilen bir metni görüntüler. Diğer kontrol nesnelerinin işlevlerinin etiketlenmesinde kullanılır. VBA yazılımı çalıştırıldığında bu metin kullanıcı tarafından değiştirilemez.
TextBox; VBA yazılımı çalıştırıldığında kullanıcı tarafından veri girişinde kullanılan bir tür metin kutusudur.
Kullanıcılar için birden fazla seçim yapılabilecek liste menüsü *ComboBox* ile oluşturulur.
Kullanıcıya sadece bir tane seçim yapabileceği açılan liste menüsü hazırlamakta *ListBox* nesnesi kullanılır.
Kullanıcı tarafından Evet-Hayır, Açık-kapalı gibi iki değerler arasında seçim yapması için *CheckBox* kullanılır.
Kullanıcı tarafından bir grup veri içerisinden bir tane veri seçecek form tasarımı yapılırken *OptionButton* kullanılır. Çoktan seçmeli sorularda seçenekler Excel hücrelerinden veya VBA kodu ile atanabilir.
Bir maddenin seçilip seçilmediğini *ToggleButton* nesnesi ile gösterilebilir. Aç-Kapa, Evet-Hayır, Doğru-Yanlış gibi ikili verilerden birinin seçilmesinde kullanılabilir.
Frame nesnesi kullanıcı formlarında grup oluşturmak için kullanılır. Bir tür çerçeve olan *Frame* nesnelerin gruplandırılmasında da kullanılabilmektedir.
Makro veya olay prosedürünü tetikleyecek nesne oluşturmak için *CommandButton* nesnesi kullanılır.

Kullanıcı formu nesne olay ilişkisini oluşturmak.

Kullanıcı formları üzerine eklenen kontrol nesnelerinin VBA yazılımındaki fonksiyon ve alt yordamlara bağlantısı olaylar ile sağlanır. Tasarlanan kullanıcı formları ile VBA'daki fonksiyonlar arasında etkileşim ve etkileşimin başlama biçimi olaylar ile belirlenir. Kontrol nesnelerinde değişiklik olması durumunda VBA'daki fonksiyon *Change* olayı ile tetiklenir. Kontrol nesnelerinin üzerine fare ile tıklandığında *Click* olayı başlar. Kullanıcı formunda tanımlanmış bir kontrol nesnesinin üzerinde çift tıklanınca VBA fonksiyon tetiklenmek isteniyorsa *DblClick* olayı kullanılır. *Enter* olayı kontrol nesnesine girilmesi ile tetiklenir. Form üzerindeki farklı bir nesneye geçildiğinde ise *Exit* olayı tetiklenir. Kullanıcı kontrol nesnesi üzerinde bir tuşa bastığında *KeyDown* olayı tetiklenir. Tuşu bıraktığında ise *KeyUp* nesnesi tetiklenir. Kullanıcı belirli tuşlara basıldığında tetiklenen olaylar *KeyPress* ile hazırlanır. *Ctr* tuşu ile beraber hazırlanan kısa yol tuşları *KeyPress* olayına örnek olarak verilebilir.

Kendimizi Sınayalım

1. Excel VBA yazılım platformunda UserForm eklendiğinde aşağıdaki pencerelerden hangisi varsayılan olarak açılır?

a. Local
b. ToolBox
c. Object Browser
d. Watches
e. Immediate

2. Kullanıcı formunun tasarım aşamasında görsel düzenlemeler yapılabilmesi için VBA pencerelerinden hangisi kullanılır?

a. Object Browser
b. List Constants
c. Immediate
d. Properties
e. Local

3. Aşağıdaki öğelerden hangisi kullanıcı formu özelliklerindeki "Görünüm Kategorisi Öğeleri"nden biri **değildir?**

a. Caption
b. PasswordChar
c. ControlTip Text
d. Visible
e. Cycle

4. VBA'da tasarlanmış bir anket formunda bir sonraki seçeneğin aktif olması açılan kutudaki cevaba bağlı ise "Davranış Kategorisi" öğelerinden hangisi kullanılır?

a. MatchEntry
b. MatchRequired
c. Enabled
d. MaxLength
e. AutoSize

5. Kullanıcı formlarında daha önce kayıt edilmiş verilerin kullanılması durumunda, kontrol kaynağının yeri hangisi "Veri Kategorisi Öğesi" ile belirlenir?

a. ControlSource
b. ListRows
c. Text
d. BoundColumn
e. ColumnWidths

6. Klavyede Tab tuşuna basıldığında imlecin kullanıcı kontrol nesnesi üzerine gelmesi için *True* değeri ayarlanan özellik öğesi aşağıdakilerden hangisidir?

a. MouseIcon
b. Tag
c. WhatsThisHelp
d. TabStop
e. MousePointer

7. VBA'da tasarlanmış yazılımda "Kaydet" butonu içine metin yazmak yerine disket resmi konulması için kullanılan özellik kategorisi aşağıdakilerden hangisidir?

a. Resim Kategorisi Öğeleri
b. Pozisyon Kategorisi Öğeleri
c. Kaydırma Çubuğu Kategorisi Öğeleri
d. Yazı Kategorisi Öğeleri
e. Veri Kategorisi Öğeleri

8. Kullanıcı formunun üzerindeki kontrollerin özellik menüsünün aktif olabilmesi için "ToolBox" içinde seçilmesi gereken kullanıcı formu kontrol nesnesi aşağıdakilerden hangisidir?

a. ListBox
b. Select Objects
c. ComboBox
d. CommandButton
e. Frame

9. Kullanıcı tarafından bir grup veri içerisinden bir tane veri seçecek şekilde form tasarımı yapılırken aşağıdaki kontrollerden hangisi kullanılır?

a. MultiPage
b. Image
c. CommandButton
d. ToggleButton
e. OptionButton

10. Kontrol nesnelerinde değişiklik olması durumunda VBA'daki olaylardan hangisi çalışmaktadır?

a. Click
b. Enter
c. Change
d. DblClick
e. KeyPress

Kendimizi Sınayalım Yanıt Anahtarı

1. b	Yanıtınız yanlış ise "Kullanıcı Formları" konusunu yeniden gözden geçiriniz.
2. d	Yanıtınız yanlış ise "Kullanıcı Formu Özellikleri" konusunu yeniden gözden geçiriniz.
3. e	Yanıtınız yanlış ise "Görünüm Kategorisi Öğeleri" konusunu yeniden gözden geçiriniz.
4. c	Yanıtınız yanlış ise "Davranış Kategorisi Öğeleri" konusunu yeniden gözden geçiriniz.
5. a	Yanıtınız yanlış ise "Veri Kategorisi Öğeleri" başlıklı konusunu yeniden gözden geçiriniz.
6. d	Yanıtınız yanlış ise "Diğer Kategori Öğeleri" başlıklı konusunu yeniden gözden geçiriniz.
7. a	Yanıtınız yanlış ise "Resim Kategorisi Öğeleri" konusunu yeniden gözden geçiriniz.
8. b	Yanıtınız yanlış ise "Kullanıcı Formu Kontrolleri" konusunu yeniden gözden geçiriniz.
9. e	Yanıtınız yanlış ise "Kullanıcı Formu Kontrolleri" konusunu yeniden gözden geçiriniz.
10. c	Yanıtınız yanlış ise "Kullanıcı Formu Nesne Olay İlişkisi" konusunu yeniden gözden geçiriniz.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1

Kullanıcı formu özellik öğelerini sınıflandırılmış görüntülediğimizde 7 ana kategori görünmektedir. Bu kategoriler; *Appearance, Behavior, Font, Misc, Picture, Position, Scrolling* şeklindedir.

Sıra Sizde 2

Hesap makinelerinde işlem kolaylığı sağlanması için rakamlar sağa dayalı olarak gelmektedir. Bu nedenle hesap makinesinde rakamların gösterileceği bir *TextBox* nesnesinin *TextAlign* özellik öğesi *3-fmTextAlignRight* şeklinde olmalıdır.

Sıra Sizde 3

Genellikle çöp kutusu gibi resimler yazılımlarda silme işlemi için kullanılan butonlarda resim olarak kullanılmaktadır.

Yararlanılan ve Başvurulabilecek Kaynaklar

Katz, A. (2011). **Excel 20110 Made Simple.** New York: Apress.

Kiong, L. V. (2009). **Made Easy.** http://www.vbtutor.net/VBA/vba_tutorial.html.

İŞLEM TABLOSU PROGLAMLAMA

7

Amaçlarımız

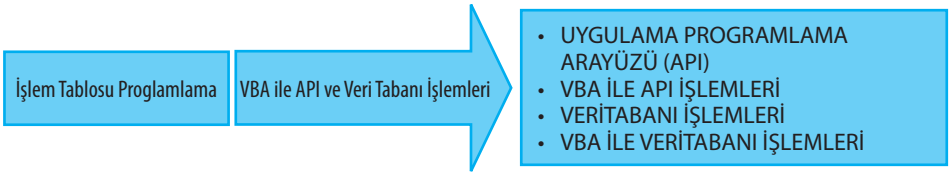
Bu üniteyi tamamladıktan sonra;

- Uygulama programlama arayüzü kavramını açıklayabilecek,
- VBA ile API işlemlerini tasarlayabilecek,
- Veri tabanı yönetim sistemlerini tanımlayabilecek,
- VBA ile veritabanı işlemlerini ilişkilendirebileceksiniz.

Anahtar Kavramlar

- Windows Software Development Kit (SDK)
- Dinamik Link Kütüphanesi (dll)
- Veritabanı Yönetim Sistemleri
- Microsoft Access

İçindekiler



VBA ile API ve Veri Tabanı İşlemleri

GİRİŞ

Excel VBA kullanıcılarının fonksiyon ve yordam hazırlarken hazır komutlar kullanabildiğini veya kendi hazırlamış olduğu fonksiyonları kullandıkları daha önceki bölümlerde anlatılmıştı. Excel VBA'da hazır komutların yetersiz olduğu durumlar için tüm yazılım dilleri ile çalışabilecek köprü vazifesi görecek yazılım dilleri kullanılır. Bu ihtiyacı karşılamak için Microsoft Windows yapısında sık kullanılan yapılardan bir tanesi dinamik link kütüphanesidir. VBA ile Uygulama Programlama Arayüzü işlemlerinde genellikle Windows işletim sistemi yapısında varsayılan olarak bulunan kütüphaneler kullanılır. Bu işlemler sırasında Dinamik link kütüphaneleri ile bağlantı kurulur ve bu kütüphanelerde tanımlanmış fonksiyonlar kullanılır.

Veritabanları büyük miktardaki bilgileri depolamak için geliştirilen birbirleriyle ilişkili bilgilerin kayıt edilmesi ve çağrılabilmesi için hazırlanmış saklama alanlarıdır. Bilgisayar temelli kayıtların tutulması için hazırlanan veritabanları, veritabanı yönetim sistemleri aracılığı ile oluşturulur ve yönetilir. VBA yazılım platformunda veritabanlarına erişim için kullanılan yöntemler örnekleri ile bu bölümde açıklanmıştır.

UYGULAMA PROGRAMLAMA ARAYÜZÜ (API)

Günümüzde insanlar ihtiyaçları doğrultusunda geliştirdikleri farklı niteliklerdeki yazılımları bilgisayarlarında kullanmaktadır. Tüm yazılımlar; girdi, süreç ve çıktıdan oluşan fonksiyonlarla çalışmasına rağmen fonksiyonların tanımlandığı yazılım dilleri farklılık göstermektedir. Yazılım dillerinin tümünün bilgisayar yazılımı yapanlar tarafından öğrenilmesi ve bu yazılım dillerinde uzmanlaşmak para, zaman ve diğer maliyetler açısından mümkün değildir.

Bilgisayar yazılımlarının geliştirildiği ilk yıllarda yazılımcılar farklı yazılım dillerinde geliştirilmiş olan fonksiyonları kullanamamakta ve tekrar hazırlamak zorunda kalmaktaydı. Örneğin, bir kasiyerin meyveleri tartmakta kullandığı A dilinde hazırlanmış yazılım ile yazar kasa fişinin yazdırılmasını sağlayan B dilinde yazılım eş güdümlü çalıştırılamadığından her iki donanımı beraber kullanmak için üçüncü bir yazılım hazırlanmak zorunda kalınmaktaydı. Yazılım fonksiyonlarını farklı dillerde tekrar yazma maliyeti, bankacılık gibi uluslararası ve geniş bir yelpazede çalışan işletmeler için oldukça yüksekti. Birbirlerinden farklı programlama dillerinde hazırlanmış yazılımların fonksiyonlarını birbirlerine adapte edecek (modüler) şekilde kullanma ihtiyacı bilgisayarların ve yazılımların artmasıyla daha da arttı. Bunun yanında birden fazla bilgisayarın eş güdümlü çalışabilmesi, yazılım tasarım süresinin kısaltılabilmesi, yazılım hatalarının daha kolay bulunabilmesi, veri akışlarının sadeleştirilerek yönetilebilmesi ve veri güvenliğinin sağlanabilmesi gibi ihtiyaçların da giderilmesi gündeme geldi. Bu nedenle yazılım dillerinin tümü ile çalışabilecek, köprü vazifesi gören üst diller geliştirilmeye başlandı. Örneğin, Microsoft

Yazılım yapıları (framework) yazılım projelerinin bir türüdür. Bir yazılım projesi kapsamında oluşturan yapı içerisinde beraber çalışan farklı nitelikteki yazılımlar ve kod kütüphaneleri bulunur. Örneğin, Windows işletim sistemi, Mac işletim sistemi birer yapı örneğidir. Bu işletim sistemleri farklı nitelikteki destekleyici programları, kod kütüphanelerini ve diğer yazılımları tek çatı altında toplar ve beraberce çalıştırır.

firması Windows yazılımlarının geliştirme sürecinde evlerinde birbirlerinden bağımsız çalışan mühendislere fonksiyon hazırlama görevleri verip bir merkezde bu fonksiyonları birleştirerek tek bir program çatısı altında çalıştırılabilecek bir yapıya ihtiyaç duydu. Böylece Microsoft Windows Software Development Kit (SDK) dilini geliştirdi. Bunun yanında 2000'li yıllarda İnternet tabanlı yazılım mimarisindeki gelişim sonucunda web tabanlı olarak çalışabilecek köprü vazifesi görecek bir dile olan ihtiyaç doğdu. İnternetin yaygınlaşması ve kullanımının artmasıyla üretici, tedarikçi, e-ticaret yapan işletme, kargo işletmesi ve bankaların yazılımlarının birbirleri ile eş güdümlü şekilde çalışması gerekti.

Tüm bu ihtiyaçları karşılamak üzere geliştirilen çözümlerden biri kütüphanelerdir. Kütüphaneler bir yazılımda kullanılmak üzere bir araya getirilmiş nesne, fonksiyon ve metotların genel adıdır. Kütüphanelerin temel kullanım amacı bir yazılım dili içerisinde tekrar kullanılabilir fonksiyonları toplayarak yazılımcının hizmetine sunmaktır. Kütüphaneler başka programlar tarafından çağırılarak çalıştırılan yapılardır.

DİKKAT

Yazılım Kütüphanesi genellikle aynı veya benzer yazılım dilinde hazırlanmış programlar tarafından kullanılabilir.

Excel VBA yazılımlarındaki komutlar varsayılan olarak yüklenen kütüphanede yer alır. İstenmesi durumunda Excel VBA'ya diğer kütüphaneler de bağlanabilir. Genellikle birden fazla kişinin beraberce üzerinde çalıştığı yazılım projelerinde geliştirilen fonksiyonlar ayrı bir kütüphanede tutulmaktadır. Böylece tekrarlanan fonksiyonların önüne geçilir. Gerektiğinde kütüphane farklı bir projeye bağlanarak daha önce hazırlanan fonksiyonlar üzerinde değişiklik yapılmadan kullanılır. Bir yazılıma birden fazla kütüphane eklenebilir. Örneğin, araç bakımı yapan bir işletme için geliştirilen bir yazılımda stokların takip edilmesinde kullanılan fonksiyonlar stok kütüphanesinde; garanti kapsamındaki araç bilgilerinin sorgulanması, kaydı vb. işlemler için üretilmiş fonksiyonlar garanti kütüphanesinde yer alabilir. Bakım yapan işletmede satın alma ile ilgili yeni bir yazılım hazırlanırken stok kütüphanesi yeni yazılımın içerisine dinamik link kütüphanesi (Dynamic Link Library-dll) ile bağlanabilir. Bunun yanında, yeni yazılımda satın almaya ilişkin yeni bir kütüphane de oluşturulabilir. Böylece bir yazılım birden fazla kütüphane ile çalıştırılmış olur.

Resim 7.1

Araç Bakımı Yapan İşletmede Kullanılan Yazılım Örneği

Uygulama Programlama Arayüzü (Application Programming Interface-API) bilgisayar programlamasında yazılım uygulamalarının birbirleri ile bütünleştirmek için oluşturulmuş rutinleri, protokolleri ve araçları içerir. API'ler yazılımları birbirlerine bağlayarak

beraber çalışmalarına imkân sağlamaktadır. Örneğin, pos cihazı kullanan işletmede yazar kasada belirtilen tutar pos cihazı yazılımına gönderilebilmektedir. Müşteri banka kartını cihazdan geçirip şifresini girdiğinde pos cihazı bilgileri sim kart ile İnternet'e bağlanarak bankaya yollamaktadır. Bankadaki yazılım; tutar bilgisini, kart bilgisi ve şifreyi kontrol etmektedir. Bilgilere göre müşteri hesabından tutar miktarı düşülerek bilgi pos cihazına geri gönderilmektedir. Pos cihazına bankadan onay geldikten sonra bu bilgiyi yazar kasa cihazına ileterek fişin yazdırılmasını sağlamaktadır. Bu işlemler, yazılımlar arasında köprü vazifesi gören API'ler ve iletişimi sağlayan katmanlar sayesinde (communication foundation) bir dakikadan kısa bir süre içerisinde gerçekleştirilebilmektedir.

Resim 7.2

POS Cihazlarında API Kullanımı

Yazılımları bütünleştirmenin yanında bilgisayar donanımları da API sayesinde kolay bir şekilde çalıştırılır. Örneğin, bilgisayar donanımına takılan *USB* bellek, *plug-in API* sayesinde bilgisayar tarafından tanınır ve bir depolama aygıtı olarak kullanılabilir. API'ler çalışırken diğer yazılımlar içerisinde yer alan fonksiyonlar yazılıma yüklenmez. Sadece API aracılığı ile çağrıldıklarında bellekte (RAM) yer alırlar. Bu özellik yazılımların daha hızlı çalışmasını sağlar. API fonksiyonları aynı yapı içerisindeki bütün programlama dillerinde ve geliştirme ortamlarında çağrılıp kullanılabilmektedir. API ile yazılımlar birbirlerine adapte edildiği için yazılımcılar fonksiyonları hazırlamak için kullanacağı dili seçmekte özgürdür. Böylece farklı yazılım dillerinde geliştirilen program blokları bir bütün gibi çalıştırılabilir.

VBA İLE API İŞLEMLERİ

Kullanıcılar Excel VBA'daki hazır komutlardan veya oluşturdukları fonksiyon ve yordamlar ile ihtiyaçlarını giderecek yazılımları yaparlar. Excel VBA fonksiyonlarının yetersiz kaldığı durumlarda ise Windows işletim sistemi yapısındaki kütüphanelerde bulunan fonksiyonların kullanılması daha verimli olacaktır. Örneğin, Excel VBA'da hazırlanan programda çıktı almak için kullanılacak pencerenin VBA kullanıcı formları ile yapılması zor ve zaman alıcıdır. Bu ve bunun gibi örneklerde VBA ile API işlemleri yapılır. Genellikle Windows işletim sistemi yapısında varsayılan olarak bulunan kütüphaneler ile bağlantı kurularak bu kütüphanelerde tanımlanmış fonksiyonlar kullanılır.

Windows İşletim Sistemindeki VBA API Kütüphaneleri

Windows işletim sisteminde VBA API ile kullanılan API kütüphanelerinden Advapi32, Windows GDI+, Comdl32, Kernel32, Shell32, User32, Netapi32 ve Winspool sıkça kullanılan kütüphanelerdir. Bu API kütüphanelerinin temel kullanım alanları ve içerdiği fonksiyonlardan bazıları Tablo 7.1'de verilmiştir.

Tablo 7.1
Windows İşletim Sistemi Yapısındaki VBA API Kütüphaneleri

Advapi32 Advapi32 (Advanced Services) Windows işletim sistemi yapısı içerisinde çekirdek kayıtların (Windows registry) ve NT güvenlik sistemlerinin (*NT Security*) fonksiyonlarını içeren kütüphanedir. VBA yazılımında bu kütüphanede bulunan fonksiyonları hazırlamak mümkün değildir. Bu nedenle API ile köprü kurularak Windows işletim sistemi yapısından alınır. *RegOpenKey, RegCloseKey, RegCreateKey, GetUserName, GetComputerName* gibi Advapi32 fonksiyonları VBA API ile kullanılmaktadır.
Windows GDI+ Windows GDI+ (Graphics Device Interface) Windows işletim sistemi yapısı içerisinde grafik temelli işlemler için hazırlanmış bir kütüphanedir. VBA kullanıcı formlarında resim temelli işlemlerde Excel VBA içerisindeki fonksiyonlar ve komutlar yeterli olmayabilir. Excel VBA'da resmin bir kaynaktan alınması, resmin tekrar boyutlandırılması, kırpma işlemi vb. işlemler Windows GDI+ ile yapılabilir. Büyük boyutlu resimlere hızlı erişim için *Imaging*, netlik gibi resim topolojisi işlemleri için *Typography*, resim üzerinde fırça vb. ile işlem yapılabilmesi için *2-D vector grapics* kullanılır. *GdipCreateBitmapFromFile, GdipCreateHBITMAPFromBitmap, GdipDisposeImage* gibi Windows GDI+ fonksiyonları VBA API ile sıkça kullanılmaktadır.
Comdl32 Comdl32 (Common Dialog Box Library) Windows işletim sistemi yapısındaki diyalog pencerelerinin yönetilmesi için kullanılan kütüphanedir. Dosya açılması, renk paletindeki renk değerlerinin seçilmesi, doküman yazdırılması gibi işlemlerde Windows işletim sistemi varsayılan olarak açılan diyalog pencerelerini Comdl32 kütüphanesinde bulundurur. Excel VBA içerisinde kullanıcı formları ile renk paleti hazırlanması zordur. Diyalog kutusu gerektiği durumlarda API ile Comdl32 arasında köprü kurularak Comdl32 kütüphanesindeki diyalog pencereleri kullanılabilir. *GetOpenFileName, GetSaveFileName, ChooseColor* gibi fonksiyonlar VBA API ile sıkça kullanılmaktadır.
Kernel32 Windows işletim sistemi yapısında, dosya ve sistem ile ilgili fonksiyonlar Kernel32 kütüphanesi ile yönetilir. Excel VBA'da genellikle dosya ve sistem fonksiyonları yer almaz. Bu nedenle dosya ve sistem işlemleri ile ilgili VBA işlemlerinde API aracılığı ile Kernel32 kütüphanesindeki fonksiyonlar kullanılır. *CreateProcess, OpenProcess, GetDiskFreeSpaceEx, GetDriveType, GetExitCodeProcess, GetTempPath, GetWindowsDirectory, SetCurrentDirectory* gibi fonksiyonlar VBA API ile sıkça kullanılan fonksiyonlardır.
Shell32 Windows işletim sistemi yapısının ana kabuğu Shell32 kütüphanesinde yer almaktadır. Programların başlatılması, yazılım ikonlarının görüntülenmesi, dosya gezgini gibi fonksiyonların büyük bir bölümü Shell32 kütüphanesi altındadır. *SHBrowseForFolder, SHFileOperation, ShellExecute, SHGetPathFromIDList, SHGetSpecialFolderLocation* gibi fonksiyonlar VBA API ile sıkça kullanılan fonksiyonlardır.
User32 Windows işletim sistemi yapısında kullanıcı arayüzü olarak User32 kütüphanesi kullanılır. Excel VBA programı hazırlanırken pencerelerin yönetilmesi, klavyedeki kısa yol işlemlerinin yapılması, bir metnin kopyalama ve yapıştırma işlemlerine ihtiyaç duyulabilir. Excel VBA fonksiyonlarının ihtiyaca cevap vermediği durumlarda Windows altındaki User32 kütüphanesi API ile köprü kurularak istenilen fonksiyonlar çalıştırılabilir. User32 kütüphanesi altında yer alan *FindWindow, FindWindowEx, GetClassName, GetDesktopWindow, GetKeyState, GetLastInputInfo, GetWindow* gibi fonksiyonlar VBA API ile sıkça kullanılan fonksiyonlardır.
Netapi32, Winspool Windows işletim sistemi yapısında ağ fonksiyonları için Netapi32 Kütüphanesi, yazdırma fonksiyonlarının yönetilmesi için ise Winspool kullanılmaktadır. VBA API ile ağ üzerinde dosya işlemeri için *NetFileClose, NetFileGetInfo, NetUseAdd, NetUseGetInfo, NetUserAdd, NetUserGetInfo* gibi Netapi32 fonksiyonları kullanılır. Yazdırma işlemlerinde ise *AddPrinter, OpenPrinter, SetPrinter* gibi Winspool kütüphanesindeki fonksiyonlar VBA API işlemlerinde sıkça kullanılmaktadır.

Windows işletim sistemi için hazırlanmış bir yazılımın MAC işletim sisteminde çalışmamasının nedeni ne olabilir?

SIRA SİZDE 1

API Deklarasyon Deyimi

VBA ile API işlemlerinde fonksiyon kütüphanelerinin kullanılabilmesi için kütüphanelerin deklare edilmesi gerekmektedir. Deklarasyon deyimi, ihtiyaç duyulan fonksiyonun bulunduğu kütüphanenin ve fonksiyonun yol ve içeriğinin tanımlandığı bir yordamdır. Aşağıda örnek olarak bir deklarasyon deyimi verilmiştir:

Public Declare PtrSafe Function GetUserName Lib "advapi32.dll" Alias _ "GetUserNameA" (ByVal lpBuffer As String, nSize As Long) As Long

- *Public:* Deklarasyon deyiminin geçerli olduğu yaşam alanını ifade etmektedir. *Public* ifadesi projenin tamamında erişime izin verildiğini gösterir.
- *Declare:* Deyimin Deklarasyon deyimi olduğunu göstermektedir
- *PtrSafe:* API köprüleme işleminin 64-bitlik işletim sistemi yapısında çalıştırılacağını göstermektedir. 32-bitlik işletim sisteminde bu deyim yer almaz. Hazırlanan yazılımın 64 veya 32 bitlik işletim sistemi yapısında çalıştırılıp çalıştırılmayacağı bilinmiyorsa basit bir koşul cümlesi kurulabilir. Böylece API hem 32 hem de 64 bitlik işletim sistemi yapılarında çalıştırılabilir. Bunun için *#If VBA7 Then* ile koşul cümlesi kurulur. Bu durumda VBA7 yani 64 bitlik bir yapı ile çalışıyorsa, deklarasyonun *PtrSafe* ifadeli olan deklarasyon deyimi çalıştırılabilir. Değilse 32 bitlik yapı ile kurulan deklarasyon deyimi çalıştırılır.
- *Function:* API ile bağlantı kurulan kütüphane içerisindeki hangi fonksiyonun kullanılacağını gösterir.
- *GetUserName:* API ile bağlantı kurulan kütüphane içerisindeki fonksiyonu gösterir. Örnekte bilgisayarın kullanıcı adını getirmeye yarayan fonksiyon yani *GetUserName* fonksiyonu için deklarasyon deyimi hazırlanmıştır.
- *Lib:* Kütüphane bağlantısı yapılacağını gösterir.
- *"advapi32.dll":* API ile bağlantı kurulacak dinamik kütüphanenin ismi verilir. *advapi32* kütüphanesi kayıt ve NT güvenlik işlemleri için kullanılan kütüphanelerden biridir.
- *Alias: advapi32* kütüphanesi içerisinden çağrılan *GetUserName* fonksiyonunun kullanıcı tarafından geliştirilen VBA yazılımındaki takma adını ifade eder. Kullanıcı yazılımda Türkçeleştirilmiş şekilde fonksiyon isimleri kullanabilir. Kütüphaneye bağlantı yapıldıktan sonra *GetUserName* fonksiyonu VBA yazılımının içerisine taşınır ve *Alias* ile belirtilen takma isimle kullanılır. Karışıklık olmaması ve API fonksiyonu olduğunun gösterilmesi için genellikle kütüphanedeki fonksiyon isminin sonuna *A* harfi eklenerek kullanılır. Örnekte takma adı *"GetUserNameA"* şeklinde ifade edilmiştir.
- *(ByVal lpBuffer As String, nSize As Long):* API fonksiyonundaki girdileri göstermektedir. Fonksiyona String türünden *lpBuffer* ve *Long* veri türünden iki değişken gönderilebilir.
- *As Long:* API fonksiyonu *Long* türünden geri dönüş sağlar.

32 bitlik Windows işletim sistemi için hazırlanmış programların 64 bitlik Windows işletim sisteminde çalışmamasının temel sebebi API kütüphaneleri arasındaki iletişimi sağlayan deklarasyon deyiminin uyumsuzluğundan kaynaklanmaktadır.

API deklarasyon deyimi sayesinde bağlantı kurulan kütüphanenin tümü değil sadece ihtiyaç duyulan fonksiyon yazılım içine çekilir. Böylece VBA yazılımının sade ve hızlı çalışması sağlanır.

Deklarasyon deyiminde kullanılan 32-64 bitlik değimlerdeki farklılıkları https://msdn.microsoft.com/en-us/library/office/ee691831(v=office.14).aspx İnternet sayfasından öğrenebilirsiniz.

İNTERNET

Hem 32 hem de 64 bitlik Windows işletim sistemi için API deklarasyon deyimi nasıl hazırlanabilir?

SIRA SİZDE 2

VBA ile API Uygulama Örneği

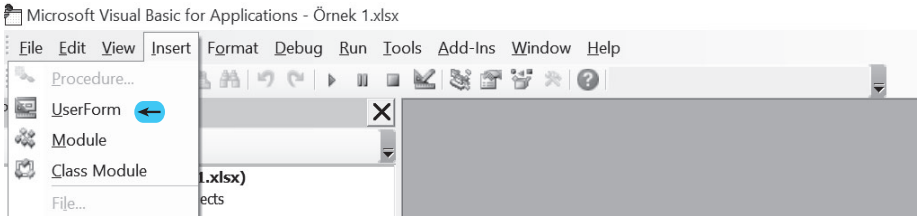
Alt ve F11 kısa yol tuşları kullanılarak da VBA yazılım platformuna geçilebilir.

Windows işletim sistemi yapısı içerisinde çekirdek kayıtların tutulduğu Advapi32 (Advanced Services) kütüphanesi ile bağlantı kurularak bilgisayar kullanıcı adı ve bilgisayar adını alacak bir örnek geliştirilecektir. VBA kullanıcı formunda bir tuşa tıklandığında API ile Advapi32 fonksiyonları çalıştırılacaktır. Kullanıcı adı ve bilgisayar adı Windows işletim sistemi yapısından alınarak kullanıcı formunda *TextBox* nesnesinin içerisine yazdırılacaktır.

Birinci aşamada Excel sayfasında *Geliştirici* sekmesindeki *Visual Basic* butonuna tıklanarak VBA yazılım platformuna geçilir. Şekil 7.1'de görüldüğü gibi *Insert* sekmesine tıklanarak açılan menü içerisinden *UserForm* seçeneğine tıklanır.

Şekil 7.1

VBA ile API Uygulama Örneği İçin Kullanıcı Formu Ekleme

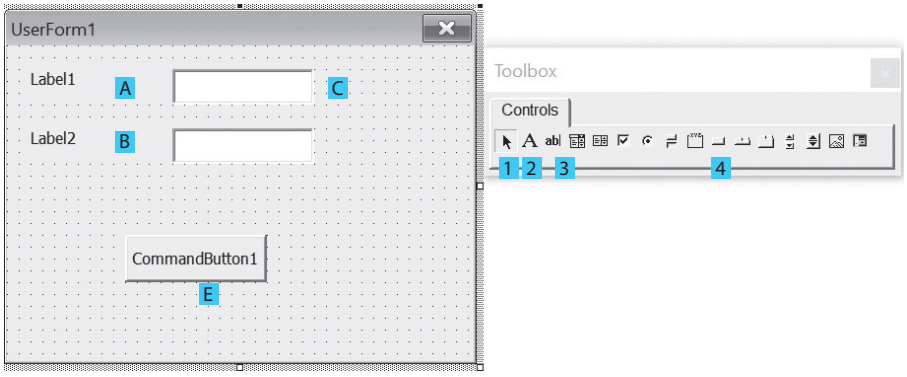


İkinci aşamada UserForm üzerine kullanıcı adı ve bilgisayar isminin yazdırılması için kullanılacak kontrollerden *Label, TextBox, CommandButton* nesneleri eklenir. Şekil 7.2'de *UserForm* üzerinde tıklanacak yerler alfabetik ve *ToolBox* menüsünde eklenecek kontrol nesneleri ise sayı ile kodlanmıştır. Sırasıyla önce *ToolBox* üzerinde *2* numaralı *Label* kontrol nesnesine sonra *UserForm* üzerinde *A* ve *B* harfi ile kodlanan yere tıklanır. Böylece form üzerinde *Label* kontrol nesnesi eklenir *(2-A* ve *2-B).* Metin kutusu eklemek için *3-C* ve *3-D* işlemi yapılır. Son olarak *4-E* işlemi yapılarak komut butonu eklenir. Kontrollerin yerinin ayarlanması için *ToolBox* üzerinde *1* numara ile kodlanan *Select Objects* tıklanır ve düzenlenmek istenen kontrol nesnesinin yeri ayarlanabilir.

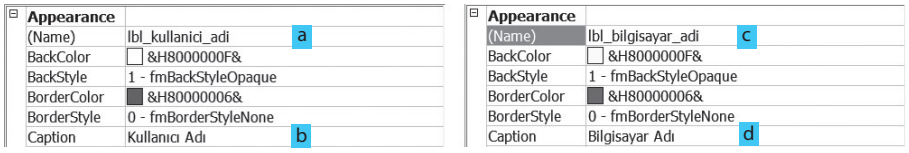
Özellik (Properties) öğelerinde değişiklik yapılabilmesi için her seferinde ToolBox üzerinde Select Objects (1) nesnesine tıklanması gereklidir.

Şekil 7.2

VBA ile API Uygulama Örneği İçin Kullanıcı Formu Tasarımı



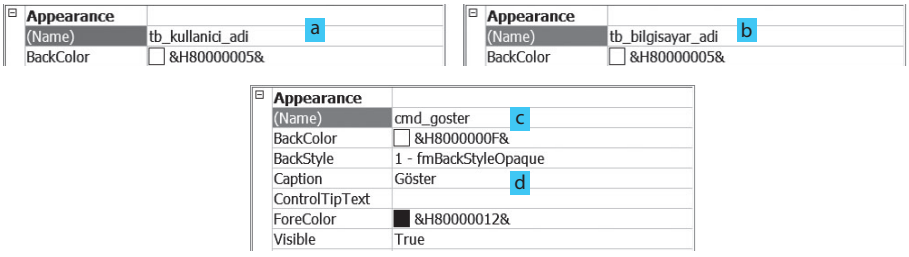
Üçüncü aşamada Label türündeki kontrol nesnelerinin özellikleri düzenlenmesi için *Proporties* penceresinde işlem yapılır. Şekil 7.2'de *UserForm* üzerinde *A* harfi ile kodlanmış *Label* özelliklerinin Şekil 7.3'te *a* harfi ile görünen *name* öğesi *lbl_kullanici_adi* şeklinde, *b* harfi ile görünen *Caption* ögesi *Kullanıcı Adı* şeklinde düzenlenir. *UserForm* üzerinde *B* harfi ile kodlanmış *Label* için *name* öğesi *lbl_bilgisayar_adi* şeklinde, *Caption* öğesi ise *Bilgisayar Adı* şeklinde düzenlenir (bkz. Şekil 7.3 *c* ve *d*).



Şekil 7.3

VBA ile API Uygulama Örneği İçin Properties Penceresi Label İşlemleri

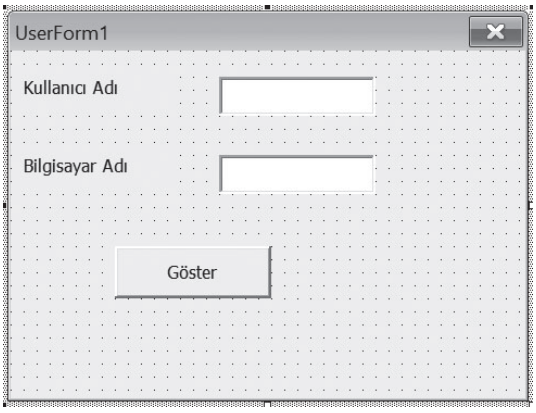
Dördüncü aşamada TextBox ve *CommandButton* türündeki kontrol nesnelerinin özellikleri düzenlenmesi için *Proporties* penceresinde işlem yapılır. Şekil 7.2'de *UserForm* üzerinde *C* harfi ile kodlanmış *TextBox* kontrol nesnesinin *Name* öğesi *tb_kullanici_adi* şeklinde, Şekil 7.2'de *D* harfi ile kodlanmış *TextBox* kontrol nesnesinin *Name* öğesi ise *tb_bilgisayar_adi* şeklinde düzenlenir (bkz. Şekil 7.4 *a* ve *c)*. Son olarak *UserForm* üzerinde *E* harfi ile kodlanan *CommandButton* türündeki kontrol nesnesinin özelliklerinden *Name* öğesi *cmd_goster* (bkz. Şekil 7.4 *c*), *Caption* öğesi ise *Göster* (bkz. Şekil 7.4 *d*) yapılır.



Şekil 7.4

VBA ile API Uygulama Örneği İçin Properties Penceresi TextBox İşlemleri

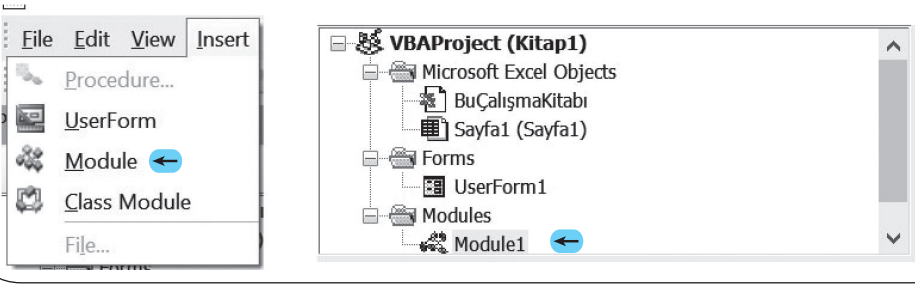
Kullanıcı formunun son hali Şekil 7.5'te görüldüğü gibi olmalıdır.



Şekil 7.5

VBA ile API Uygulama Örneği Kullanıcı Formu Son Hâli

Beşinci aşamada API deklarasyon deyiminin tanımlanması için projeye bir modül eklenecektir. VBA yazılım platformunda Şekil 7.6'da görüldüğü gibi *Insert* sekmesine tıklanarak açılan menü içerisinden *Module* seçeneğine tıklanır.



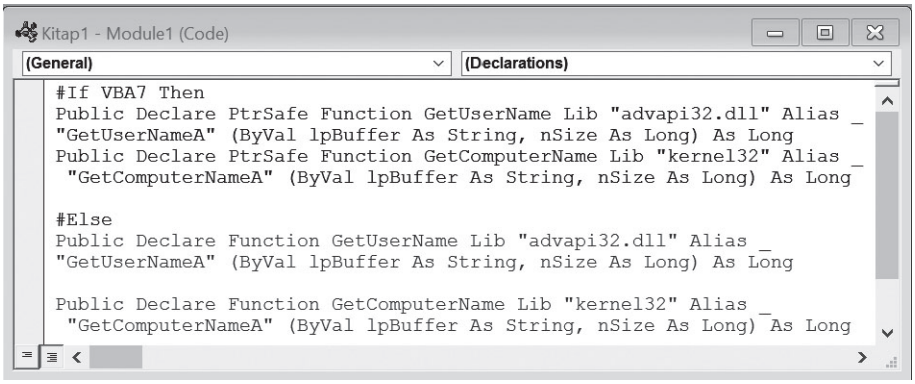
Şekil 7.6

VBA ile API Uygulama Örneği İçin Modül Ekleme

Altıncı aşamada API deklarasyon deyiminin tanımlanması VBA proje penceresinin içerisinde *Module1* nesnesinin üzerine çift tıklanır. Şekil 7.7'de görüldüğü gibi ve açılan pencere içerisine *advapi32.dll* kütüphanesinden *GetUserName* ve *GetComputerName* fonksiyonları ile hazırlanan VBA yazılımı arasında köprü oluşturulur.

Şekil 7.7

VBA ile API Uygulama Örneği İçin API Deklarasyon Deyimi

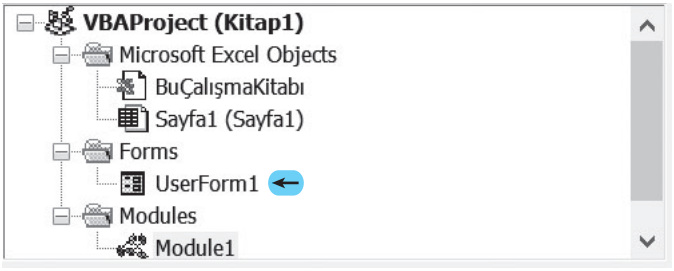


Hazırlanan VBA yazılımının çalıştırılacağı bilgisayar bilinmediğinden koşul cümleciği ile deklarasyon deyimi kullanılmıştır. Böylece advapi32 içerisinden alınacak fonksiyonlar hem 32 hem de 64-bitlik işletim sistemi yapısında çalıştırılabilir.

Yedinci aşamada UserForm penceresine ulaşmak için VBA proje penceresinde Şekil 7.8'de görülen *UserForm1* formunun üzerine çift tıklanır.

Şekil 7.8

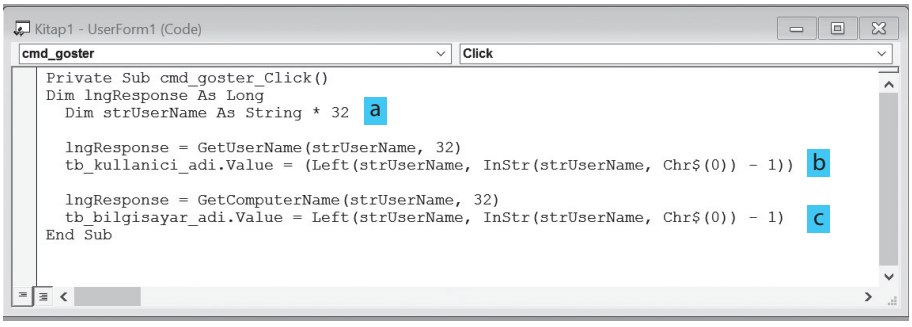
VBA ile API Uygulama Örneği İçin UserForm İşlemleri



Sekizinci aşamada UserForm'da Şekil 7.2'de *E* harfi ile gösterilen *CommandButton* kontrol nesnesinin üzerine tıklanarak *cmd_goster* nesnesinin *click* olayı oluşturulmuştur. Şekil 7.9'da *a* harfi ile kodlanan bölümde fonksiyonlara gönderilecek *String* ve *Long* türünde değişken paketleri tanımlanmıştır. *b* ile kodlanan bölümde fonksiyon çalıştırılarak *Long* türündeki veri paketine kullanıcı adı atanmış ve *tb_kullanici_adi* isimli *TextBox* nesnesine *Long* türündeki değişken string türüne çevrilerek atanmıştır. Şekil 7.9 *c* harfi ile kodlanan bölümde ise aynı işlemler bilgisayar ismi ile ilgili yapılmıştır.

Şekil 7.9

VBA ile API Uygulama Örneği İçin UserForm İşlemleri



VERİ TABANI İŞLEMLERİ

Veritabanları birbirleriyle ilişkili bilgilerin ihtiyaç duyulduğunda tekrar çağrılabilmesi için hazırlanmış bir tür saklama alanıdır. Veritabanları; büyük miktardaki bilgileri depolamak için geliştirilmiştir. Cep telefonlarında kullanılan rehber, gündelik hayatta sıkça kullandığımız veri tabanı uygulamalarından sadece biridir. Günümüzde lojistik, telekomünikasyon, idari işlemler, kamu hizmetleri gibi birçok alanda veri tabanı altyapısını kullanan bilgisayar sistemleri ile hizmet verilmektedir. Bilgisayar temelli kayıtların tutulması için hazırlanan veritabanları veri tabanı yönetim sistemleri (Database Management System-DBMS) aracılığı ile oluşturulur ve yönetilir. Microsoft Access, Microsoft SQL Server, MySQL, Oracle, IBM DB2, Informix, PostgreSQL, Interbase ve Sysbase gibi yazılımlar veri tabanı yönetim sistemlerine örnek olarak verilebilir.

Veri tabanlarında genellikle tablolar bulunur ve bu tablolar bir birleri ile belirli alanlarda ilişkilidir. Veri tabanı ilişkileri birden bire, birden çoğa, çoktan bire ve çoktan çoğa olabilmektedir. Vatandaşlık numarasını kullanarak bankacılık, sağlık, nüfus ve eğitim gibi hizmetlerin yapılabilmesi birden çoğa ilişkili veri tabanı sistemine örnek olarak verilebilir. Bu yapıda vatandaşlık numarası veri tabanında diğer bilgilerin bağlandığı bir anahtar olarak çalışır. Bilgisayar sisteminde bu anahtar girildiğinde anahtara bağlanmış diğer tablolardaki verilerde otomatik olarak çağrılabilmektedir. Veritabanlarının bu özelliği veri tekrarlarını en aza indirilebilmektedir. Veritabanlarındaki bilgiler sorgu cümleleri ile çağrılır. Sorgu cümleleri büyük miktardaki veriyi klasik dosya sistemlerine göre çok daha hızlı süzülebilmektedir. Fakat veritabanlarının kurulum ve bakımı klasik dosya sisteminden daha pahalıdır. Bunun yanında veri tabanı bileşenleri iyi tasarlanmazsa veri tabanı sorgu ve kayıt işlemleri başarısız sonuçlar üretebilmektedir.

VBA İLE VERİ TABANI İŞLEMLERİ

VBA yazılım platformunda veritabanlarına erişim için 3 farklı arayüz kullanılabilir.

- ADO (ActiveX Data Objects),
- RDO (Remote Data Objects),
- DAO (Data Access Objects).

Bu erişim arayüzleri VBA yazılımı ile veritabanları arasında bağlantı sağlamak için kullanılan bir tür kütüphanelerdir. Bu kütüphanelerde bağlantı, erişim, kontrol ve veri geri dönüşünü içeren fonksiyonlar vardır. Bir birlerinden farklı 3 arayüz olmasının temel sebebi farklı gelişim evrelerinde kullanılmasıdır. Örneğin, son bağlantı yöntemi ADO diğer bağlantı arayüzlerine göre daha basit ve esnek bir yapıda geliştirilmiştir.

VBA veri tabanı örnekleri ADO arayüzü ile verilecektir. Çünkü ADO kolay kullanımı olan ve Microsoft'un en yeni ve güçlü veri tabanı bağlantı arayüzüdür. Bunun yanında veri tabanı yönetim sistemlerinin birçoğuna ADO arayüzü ile erişilebilir.

ADO ile veri tabanı bağlantısı yapılabilmesi için öncelikle referans olarak *Microsoft ActiveX Data Objects X.X Library* kütüphanelerinden biri eklenmelidir. Bunun için VBA yazılım platformunda *Tools* üst menüsü ve *Referances* seçeneği kullanılarak kütüphane ekleme işlemi yapılır.

VBA ile veri tabanı farklı yazılımlar olduğundan ADO arayüzü aracılığı ile veri tabanı bağlantısı yapılabilir. VBA ile veri tabanı bağlantısını günlük hayattan bir örnekle açıklanabilir. Adnan (ADO) adında çok yakın bir arkadaşınız varsa ve Adnan' ın sınıf arkadaşı olan fakat sizin doğrudan tanımadığınız Deniz' in (Database-DB) çalıştığı işletmede yarı zamanlı bir personel alımı yapılacağını duyuruldu. Başvuru yapmak istiyorsunuz fakat iş ilanına doğrudan erişemiyorsunuz. Duyduğunuz iş ilanına erişmek için Adnan' dan yardım istiyorsunuz. Bunun için Adnan size aracı olarak Deniz ile bir görüşme ayarlıyor. Deniz'e durumu anlatıp öz geçmişinizi veriyorsunuz. Ardından Deniz, öz geçmişinizi per-

sonel arayan birime iletiyor. VBA ile ADO aracılığı ile veri tabanı bağlantısı bu örneğe benzer şekilde gerçekleşir. Öncelikle bir bağlantı olması gerekir. Bağlantının türü ve yolu tanımlanır ve bağlantı yolu açılır. Örnekteki bağlantıda Adnan ve bağlantı yolu Adnan'ın sınıf arkadaşlığıdır. Bağlantının açılması ise işe başvuru yapmak istediğinizin Adnan'a anlatılmasıdır. ADO işlemlerinde daha sonra tablo işlemleri gerçekleştirilir. Tabloya bağlantı atanır, tablonun tanımı yapılır ve tablo açılarak tablo işlemi gerçekleştirilir. Örnekte, bu Adnan'ın Deniz ile buluşma ayarlaması ve sizin Deniz' den öz geçmişinizi iş ihtiyacı olan birime iletmesini rica etmeniz ile oluyor. Tablo işlemleri ise Deniz' in özgeçmişinizi müdürüne iletmesine benzetilebilir. VBA ile ADO aracılığı ile veri tabanı işlemlerini aşağıdaki gibi özetlemek mümkündür:

- Bağlantı:
  - Yeni bağlantı tanımı,
  - Veri tabanı bağlantı yolu atanması,
  - Bağlantının açılması.
- Tablo:
  - Yeni tablo tanımı,
  - Tabloya veri tabanı bağlantısının atanması,
  - Tablo kaynağının tanımlanması,
  - Tablonun açılması.
- Tablo işlemleri:
  - Veri tabanından okuma,
  - Veri tabanına yazma,
  - Veri tabanından silme.
- Kapatma:
  - Tablonun kapatılması,
  - Tablo yolunun temizlenmesi,
  - Bağlantının kapatılması,
  - Bağlantı yolunun temizlenmesi.

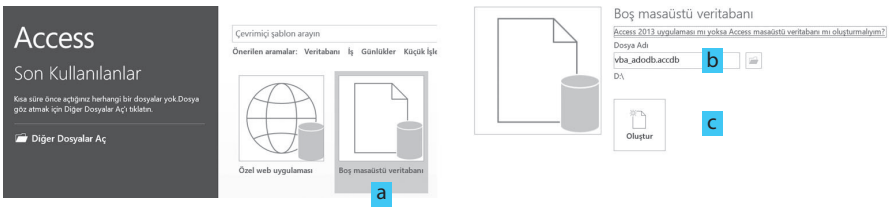
VBA ile Veri Tabanı Uygulama Örneği

VBA ile veri tabanı uygulama örneği kapsamında Excel VBA ile hazırlanmış bir kullanıcı formu ile Microsoft Access veri tabanına bağlanacak bir yazılım hazırlanacaktır. Bağlantı ADO arayüzü ile gerçekleştirilecektir. VBA kullanıcı formu ile kullanıcıdan alınacak adı, soyadı ve yaş verileri veri tabanına eklenecektir. Ayrıca eklenen verileri düzeltecek veya silecek butonlar da tasarlanacaktır. Adı, soyadı ve yaş verileri *TextBox* kontrol nesnesinden alınacaktır. Veri tabanından okunan veriler *ListBox* içinde görüntülenecektir. Kullanıcı *ListBox* içerisinden seçtiği veriyi *TextBox* kontrol nesnesinin içerisine aktararak silme veya düzenleme yapabilecektir.

Birinci aşamada Microsoft Office 2013 programları arasında yer alan Microsoft Access programı çalıştırılır. Şekil 7.10' da *a* harfi ile gösterilen *Boş masaüstü veri tabanı* seçeneğine tıklanır. Boş masaüstü veri tabanı ekranında veri tabanının oluşturulacağı yer ve adı seçilir *(bkz. Şekil 7.10 b)* ve *c* harfi ile gösterilen *Oluştur* butonuna tıklanır.

DİKKAT

Kullandığınız kişisel bilgisayarın güvenlik ve paylaşım ayarları nedeniyle örneğin çalışmama olasılığına karşı veri tabanının fat32 formatında biçimlendirilmiş bir taşınılabilir bellekte oluşturulmasında fayda vardır.



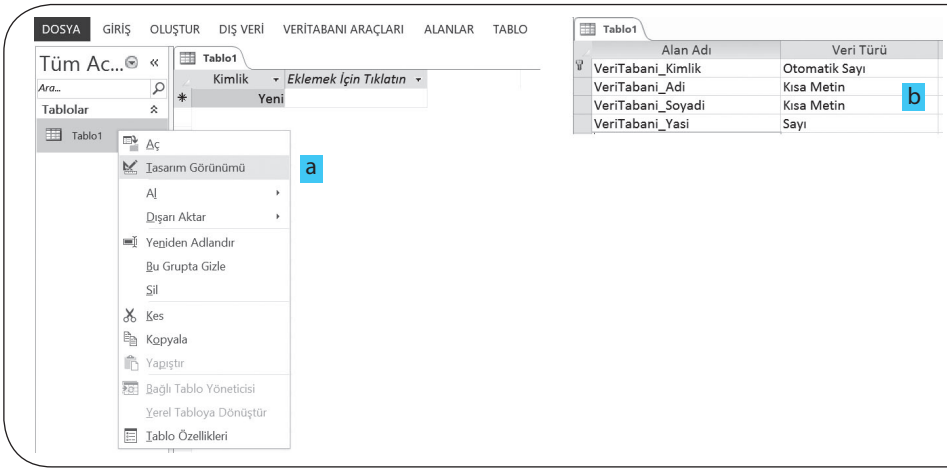
Şekil 7.10

Access Veritabanı Açma İşlemleri

İkinci aşamada Şekil 7.11'de *a* harfi ile gösterilen pencere görüntülenir. Microsoft Access programı, varsayılan olarak bir tablo oluşturmuştur. Tablo üzerinde sağa tıklanarak *Tasarım Görünümü* seçeneği seçilir. *Tasarım Görünümü* seçeneği seçildiğinde *Tablo1* içerisinde yer alacak alan adı ve veri türü bilgilerinin girileceği Şekil 7.11'de *b* harfi ile gösterilen pencere açılır. Sırasıyla *VeriTabani_Kimlik-Otomatik Sayı, VeriTabani_Adi-Kısa Metin, VeriTabani_Soyadi-Kısa Metin, VeriTabani_Yasi-Sayı* bilgileri tabloya girilerek kayıt edilir. Microsoft Access programı kapatılır.

Kullanılan yazılım diline bağlı olarak Microsoft Access veri tabanında oluşturulan tablo isminde değişiklik olabilir. Türkçe Microsoft Access programında Tablo1 varsayılan olarak açılacaktır.

DİKKAT



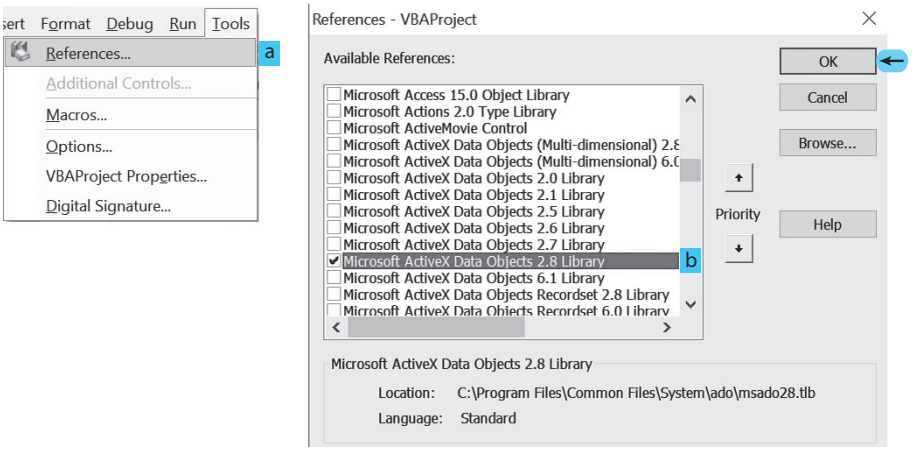
Şekil 7.11

Access Veritabanı Tablo İşlemleri

Üçüncü aşamada Microsoft Excel programı açılarak *Geliştirici* sekmesindeki *Visual Basic* butonuna tıklanır ve VBA yazılım platformuna geçilir. VBA yazılım platformunda Şekil 7.12'de *a* harfi ile gösterildiği gibi üst menüden *Tools* menüsü açılarak *References* menüsüne tıklanır. *References* menüsü VBA yazılım platformuna kütüphane eklemede kullanılan yöntemlerden biridir. ADO kütüphanesinin eklenebilmesi için Microsoft ActiveX Data Objects 2.x Library seçeneklerinden birisi seçilir ve OK tuşuna basılır.

Şekil 7.12

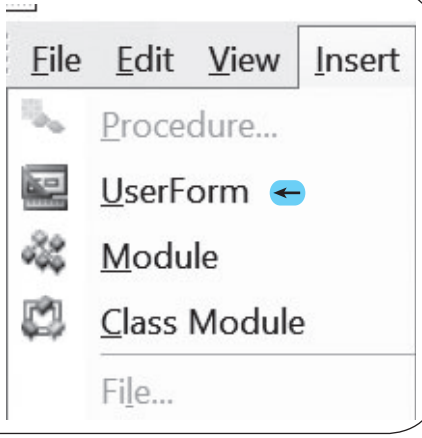
Excel VBA Referans Ekleme İşlemleri



Dördüncü aşamada VBA yazılım platformunda *Insert* açılan menüsünden *UserForm* seçeneği seçilerek projeye bir kullanıcı formu eklenir (Şekil 7.13).

Şekil 7.13

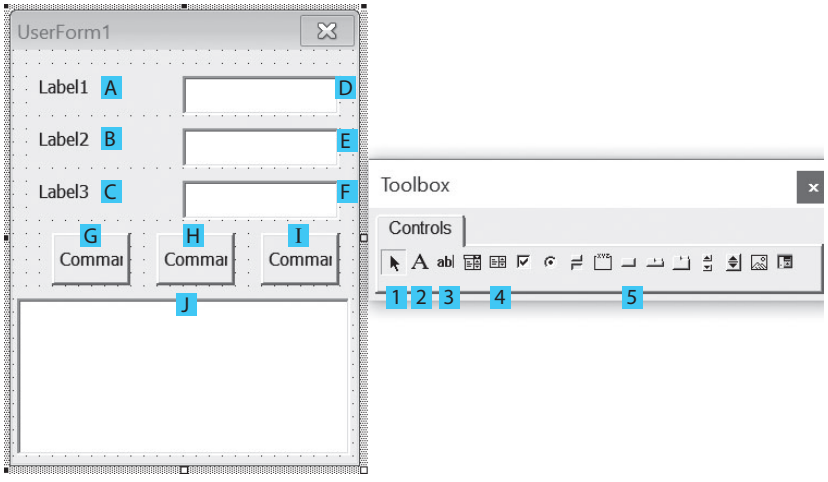
Excel VBA Referans Ekleme İşlemleri



Beşinci aşamada UserForm üzerine adı, soyadı ve yaş verilerinin kayıt, silme ve düzeltme işlemleri için kullanılacak *Label, TextBox, CommandButton* ve *ListBox* nesneleri eklenir. Şekil 7.14'te *UserForm* üzerinde tıklanacak yerler alfabetik ve *ToolBox* menüsünden eklenecek kontrol nesneleri ise sayı ile kodlanmıştır. Sırasıyla önce *ToolBox* üzerinde *2* numaralı *Label* kontrol nesnesine sonra *UserForm* üzerinde *A*, *B* ve *C* harfi ile kodlanan yerlere tıklanır. Böylece form üzerinde *Label* kontrol nesnesi eklenir *(2-A, 2-B, 2-C).* Metin kutusu eklemek için *3-D, 3-E* ve *3-F* işlemleri yapılır. Komut düğmeleri *5-G, 5-H ve 5-I* işlemleri yapılarak eklenir. Son olarak *4-J* işlemi ile *ListBox* kontrol nesnesi eklenir. Kontrollerin yerinin ayarlanması için *ToolBox* üzerinde *1* numara ile kodlanan *Select Objects* tıklanır ve düzenlenmek istenen kontrol nesnesinin yeri ayarlanabilir.

Şekil 7.14

Veritabanı Uygulama Örneği İçin Kullanıcı Formu Tasarımı



Altıncı aşamada Label türündeki kontrol nesnelerinin özellikleri düzenlenmesi için *Proporties* penceresinde işlem yapılır. Şekil 7.14'te *UserForm* üzerinde *A* harfi ile kodlanmış *Label* özelliklerinde Şekil 7.15'te *a* harfi ile görünen *Name* öğesi *lbl_adi*, *b* harfi ile görünen *Caption* öğesi ise *Adı* şeklinde düzenlenir (*A-a lbl_adi, A-b Adı)*. Kullanıcı formu üzerindeki *B, C* harfi ile kodlanan *Label* nesneleri için sırasıyla *B-c lbl_soyadi, B-d Soyadı* ve *C-e lbl_yasi, C-f Yaşı* şeklinde düzenlenir.

Şekil 7.15

Veritabanı Uygulama Örneği İçin Label İşlemleri

Appearance	
(Name)	lbl_adi a
BackColor	&H8000000F&
BackStyle	1 - fmBackStyleOpaque
BorderColor	&H80000006&
BorderStyle	0 - fmBorderStyleNone
Caption	Adı b

Appearance	
(Name)	lbl_soyadi c
BackColor	&H8000000F&
BackStyle	1 - fmBackStyleOpaque
BorderColor	&H80000006&
BorderStyle	0 - fmBorderStyleNone
Caption	Soyadı d

Appearance	
(Name)	lbl_yasi e
BackColor	&H8000000F&
BackStyle	1 - fmBackStyleOpaque
BorderColor	&H80000006&
BorderStyle	0 - fmBorderStyleNone
Caption	Yaşı f

Yedinci aşamada Şekil 7.14'te *UserForm* üzerinde *D, E ve F* harfleri ile kodlanmış *TextBox* türündeki kontrol nesnelerinin *Name* öğeleri Şekil 7.16'da görüldüğü gibi *tb_adi* (*D-a*), *tb_soyadi* (*E-b*), *tb_yasi* (*F-c*) şeklinde düzenlenir.

Şekil 7.16

Veritabanı Uygulama Örneği İçin TextBox İşlemleri

Appearance	
(Name)	tb_adi a

Appearance	
(Name)	tb_soyadi b

Appearance	
(Name)	tb_yasi c

Sekizinci aşamada kayıt, silme ve düzeltme işlemleri için kullanılacak komut düğmelerinin (Şekil 7.14 *G, H ve I*) *Name* ve *Caption* öğeleri ile ilgili işlem yapılır. Şekil 7.17'de görüldüğü gibi *G-a cmd_kaydet, G-b Kaydet, H-c cmd_duzelt, H-d Düzelt, I-e cmd_sil ve I-f Sil* şeklinde düzenlenir.

Şekil 7.17

Veritabanı Uygulama Örneği İçin CommandButton İşlemleri

Appearance	
(Name)	cmd_kaydet a
BackColor	&H8000000F&
BackStyle	1 - fmBackStyleOpaque
Caption	Kaydet b

Appearance	
(Name)	cmd_duzelt c
BackColor	&H8000000F&
BackStyle	1 - fmBackStyleOpaque
Caption	Düzelt d

Appearance	
(Name)	cmd_sil e
BackColor	&H8000000F&
BackStyle	1 - fmBackStyleOpaque
Caption	Sil f

Dokuzuncu aşamada Şekil 7.14'te *J* harfi ile gösterilen *ListBox* kontrol nesnesinin özellik öğesi Şekil 7.18'de görüldüğü gibi düzenlenir.

Şekil 7.18

Veritabanı Uygulama Örneği İçin ListBox İşlemleri

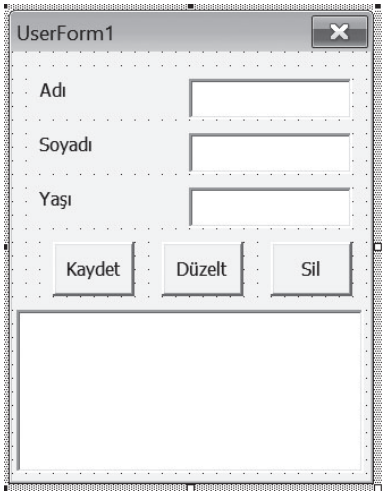
Appearance	
(Name)	list_kayit

Kullanıcı formu tasarımı bittiğinde Şekil 7.19'da görüldüğü gibi olacaktır.

Onuncu aşamada kullanıcı formu tasarımı tamamlanmasının ardından kod yazımına geçilir. Bu aşamada bağlantı yolu atamasının kolay yapılabilmesi ve esnek olması için *baglantiyolu* şeklinde tanımlanmış ADO arayüzü için kullanılacak bir fonksiyon tanımlanır (Şekil 7.20).

Şekil 7.19

Veritabanı Uygulama Örneği Kullanıcı Formu Tasarımı



"Provider=Microsoft.ACE.OLEDB.12.0; Data Source=D:\vba_adodb.accdb; Persist Security Info=False;"

Bağlantı yolu tanımlamasında ADO kütüphanesinde kullanılacak veri tabanı bağlantısını sağlayacak protokol tanımlanarak (*Microsoft. ACE.OLEDB.12.0*), veri tabanının dosya ve dizin içerisindeki yeri verilir (*D:\vba_adodb.accdb*) ve güvenlik protokolleri ayarlanır (*Persist Security Info=False;"*).

On birinci aşamada veri tabanından veri okuyarak *ListBox* içerisine okunan verileri atayacak

Şekil 7.20

VBA ile Veritabanları Uygulama Örneği İçin Bağlantı Yolu Tanımlama İşlemleri

```
(General)                                    kayit_oku
Public Function baglantiyolu() As String

baglantiyolu = "Provider=Microsoft.ACE.OLEDB.12.0; Data Source=D:\vba_adodb.accdb;Persist Security Info=False;"

End Function
```

alt yordam tasarlanacaktır. Öncelikle yeni bağlantı tanımları yapılır (*bkz.7.21 a*). Onuncu adımda tanımlanmış bağlantı yolu bağlantı deyimi (*ConnectionString)* içerisine atanır ve bağlantı açılır (*bkz. 7.21 b)*. Bağlantı açıldıktan sonra veri tabanı ile bir köprü kurulmuş olur. Veri tabanındaki tablodan okuma yapılabilmesi için tablo ile aktif bağlantının ilişkilendirilmesi, tablo kaynağının *(Source)* tanımlanması ve açılması gerekir (*bkz. 7.21 c)*. Veri tabanından okunacak tablo *ListBox* içerisine atılacaktır. Bu nedenle okuma işleminden önce *Clear* komutu kullanılarak *list_kayit* isimli *ListBox* içeriği temizlenir (*bkz. 7.21 d*). Veri tabanındaki tablonun sonuna kadar okunması için *Do… Until* döngüsü ile *EOF* (End Of File) yapısı kullanılır. Veri tabanında tanımlanmış alanlar toplam 4 adettir. Bu nedenle tablo alanları 0'dan başlayarak 3'e kadar alınarak *CStr* komutu ile metine dönüştürülür ve *list_kayit* isimli kontrol nesnesinin içinde *AddItem* komutu ile eklenir. Okunan her bir alanın birbirinden ayrılabilmesi için aralarına tire işareti konulur (*bkz.7.21 e)*. Son olarak tablo kapatılır, tablo yolu temizlenir, bağlantı kapatılır ve bağlantı yolu temizlenir (*bkz. 7.21 f*).

Şekil 7.21

VBA ile Veri Tabanları Uygulama Örneği için Kayıt Okuma Alt Yordamı

```
Public Sub kayit_oku()

    Dim baglanti As New ADODB.Connection        a
    Dim tablo As New ADODB.Recordset

    baglanti.ConnectionString = baglantiyolu()   b
    baglanti.Open

    tablo.ActiveConnection = baglanti
    tablo.Source = "Select * From Tablo1"        c
    tablo.Open

    list_kayit.Clear     d

    Do Until tablo.EOF
        list_kayit.AddItem CStr(tablo.Fields(0).Value) + "-" + _
                           CStr(tablo.Fields(1).Value) + "-" + _   e
                           CStr(tablo.Fields(2).Value) + "-" + _
                           CStr(tablo.Fields(3).Value)
        tablo.MoveNext
    Loop

    tablo.Close
    Set tablo = Nothing
                                 f
    baglanti.Close
    Set baglanti = Nothing

End Sub
```

On ikinci aşamada on birinci aşamada tanımlanan veri tabanından veri okuma alt yordamını tetikleyecek kullanıcı formu nesnesi bağlantısı Şekil 7.22'de görüldüğü gibi yapılır.

Şekil 7.22

VBA ile Veri Tabanları Uygulama Örneği için Kayıt Okuma Yordamını Çağırma

```
Private Sub UserForm_Initialize()

    Call kayit_oku

End Sub
```

On üçüncü aşamada kayıt yapma alt yordamı tasarlanacaktır. Kullanıcı formunda *tb_adi, tb_soyadi ve tb_yasi* şeklinde isimlendirilen metin kutularına yazılanların veri tabanına kayıt edilmesi için bu alt yordam kullanılacaktır. Öncelikle yeni bağlantı tanımlanır (*bkz.7.23 a*). Onuncu adımda tanımlanmış bağlantı yolu, bağlantı deyimi içerisine atanır ve bağlantı açılır (*bkz. 7.23 b)*. Veri tabanındaki tablo ile ilgili yazma işlemi yapılabilmesi için tablo ile aktif bağlantı ilişkilendirilir. Bir sonraki aşamada tablo yazılmak üzere açılır (*bkz. 7.23 c*). Tabloya yeni değerlerin yazdırılması için *AddNew* komutu kullanılır. Tırnak içerisine belirtilen tablo alanlarına metin kutusundan alınan değerler veri türü değişimi yapılarak atanır (*bkz. 7.23 d*). Veri tabanı içerisindeki değerlerin kalıcı olarak saklanması için veri tabanı güncellenir (*bkz. 7.23 e*). Veri tabanı ve tablo bağlantısı kapatılarak metin kutularındaki değerler silinir (*bkz. 7.23 f ve g*).

Şekil 7.23

VBA ile Veri Tabanları Uygulama Örneği için Kayıt Yapma Alt Yordamı

```
Public Sub kayit_yap()
    Dim baglanti As New ADODB.Connection                    a
    Dim tablo As New ADODB.Recordset

    baglanti.ConnectionString = baglantiyolu()    b
    baglanti.Open

    tablo.ActiveConnection = baglanti
    tablo.Open "Tablo1", baglanti, adOpenKeyset, adLockOptimistic, adCmdTable  c

    tablo.AddNew
    tablo("VeriTabani_Adi").Value = CStr(tb_adi.Value)
    tablo("VeriTabani_Soyadi").Value = CStr(tb_soyadi.Value)  d
    tablo("VeriTabani_Yasi").Value = CDbl(tb_yasi.Value)

    tablo.Update  e

    tablo.Close
    Set tablo = Nothing
    baglanti.Close          f
    Set baglanti = Nothing

    tb_adi.Value = ""
    tb_soyadi.Value = ""  g
    tb_yasi = ""

End Sub
```

On dördüncü aşamada kayıt yapma alt yordamının *cmd_kaydet* isimli *CommandButton* kontrol nesnesi içerisinden tetiklenmesi için VBA kodu hazırlanacaktır. Buton içerisinde ilk olarak metin kutularının tümünün dolu olup olmadıkları Şekil 7.24' te *a* harfi ile gösterilen alanda kontrol edilir. Metin kutularından boş olan varsa kullanıcıya tüm formu doldurmalarına yönelik uyarı kutusu açılır. Metin kutuları doluysa önce kayıt yapmak için hazırlanmış yordam çalıştırılır ve ardından kayıtların okunduğu yordam çalıştırılır (*Şekil 7.24 b)*.

Şekil 7.24

VBA ile Veri Tabanları Uygulama Örneği için Kaydetme Alt Yordamı

```
Private Sub cmd_kaydet_Click()

    If IsEmpty(tb_adi.Value) = True Or _
       IsEmpty(tb_soyadi.Value) = True Or _
       IsEmpty(tb_yasi.Value) = True Then    a

        MsgBox "Lütfen Tüm Formu Doldurun"
    Else
        Call kayit_yap                        b
        Call kayit_oku
    End If

End Sub
```

On beşinci aşamada kullanıcı formunda *ListBox* kontrol nesnesi içerisinde verilere tıklanması durumunda çalışacak alt yordam hazırlanacaktır. *ListBox* kontrol nesnesi içerisinde veri tabanından okunmuş olan kayıtlar vardır. Veri tabanından okunan kayıt alanları arasında tire (-) işareti vardır. *ListBox* kontrol nesnesi içerisinde yer alan verinin üzerine tıklandığında tire (-) işaretinden veriler bölünerek adı, soyadı ve yaşı bilgileri metin kutularının içerisine yazdırılır. Bu işlem için Şekil 7.25 *a* harfi ile gösterildiği gibi *Variant* veri tipinde *secilen_kayit* isminde değişken paketi tanımlanır. For döngüsü ile *ListBox* içerisindeki kayıtlar tek tek kontrol edilir ve seçilen kayıt tespit edilir. Seçilen kayıt *Split* komutu ile tire – işaretinden bölünerek *secilen_kayit* ismindeki değişken paketine atanır (*Şekil 7.25 b*). Değişken paketinin (*secilen_kayit*) içerisinde veri olup olmadığı son kez kontrol edilerek metin kutularının içerisine seçilen kayıt bilgileri aktarılır (*Şekil 7.25 c*).

Şekil 7.25

VBA ile Veri tabanları Uygulama Örneği İçin Liste Tıklandığında Çalışacak Alt Yordam

```
Private Sub list_kayit_Click()

    Dim secilen_kayit As Variant   a

    For i = 0 To list_kayit.ListCount
        If list_kayit.Selected(i) Then
            secilen_kayit = Split(CStr(list_kayit.List(i)), "-")  b

        End If
    Next i

    If IsEmpty(secilen_kayit) = True Then
        MsgBox "Lütfen listeden Seçim Yapınız"
    Else                                          c
        If UBound(secilen_kayit) = 3 Then
            tb_adi.Value = secilen_kayit(1)
            tb_soyadi.Value = secilen_kayit(2)
            tb_yasi.Value = secilen_kayit(3)
        End If
    End If
End Sub
```

On altıncı aşamada ListBox kontrol nesnesi içerisinde veri tabanından okunup *TextBox* kontrol nesnelerinin içerisine *on beşinci aşamada* atılmış olan verilerin düzeltilme işlemi için işlem yapılacaktır. *cmd_duzelt* şeklinde isimlendirilmiş *CommandButton* kontrol nesnesine tıklandığında öncelikle yeni bağlantı tanımları yapılır (*Şekil 7.26 a*). Bir sonraki adımda *LisxBox* içerisinde seçilmiş olan veri Şekil 7.26 *b*'de gösterildiği gibi okunur, tire (-) işareti olan yerlerden bölünerek *secilen_kayit* değişken paketi içerisine atılır. Değişken paketinin içerisinin boş olup olmadığı kontrol edilerek bağlantı yolu ataması yapılır ve bağlantı açılır (*Şekil 7.26 c* ve *d*). Şekil 7.24'te *e* harfi ile gösterildiği gibi tabloya bağlantı atanır ve sadece düzeltilmek istenen veri, tablo içerisinden *Where* deyimi ile seçilerek açılır. Tablo içerisindeki alanlara düzeltilmek istenen metin kutusundaki kayıtlar atanarak tablo güncellenir (*Şekil 7.26 f*). Şekil 7.26'da *g* harfi ile gösterildiği gibi tablo ve bağlantı kapatılır. Metin kutusundaki değerler silinir ve *ListBox* içerisinde seçilmiş veri ilk hâline dönüştürülerek *kayit_oku* alt yordamı ile güncellenmiş veriler tablodan tekrar okunur (*Şekil 7.26 i* ve *j*).

Şekil 7.26

VBA ile Veritabanları Uygulama Örneği İçin Düzelt Butonuna Tıklandığında Çalışacak Alt Yordam

```
Private Sub cmd_duzelt_Click()
    Dim baglanti As New ADODB.Connection     a
    Dim tablo As New ADODB.Recordset

    For i = 0 To list_kayit.ListCount
        If list_kayit.Selected(i) Then
            secilen_kayit = Split(CStr(list_kayit.List(i)), "-")  b
        End If
    Next i

     If IsEmpty(secilen_kayit) = True Then
        MsgBox "Lütfen listeden Seçim Yapınız"          c
     Else
        If UBound(secilen_kayit) = 3 Then

            baglanti.ConnectionString = baglantiyolu()    d
            baglanti.Open

            tablo.ActiveConnection = baglanti
            tablo.Open "Tablo1 Where VeriTabani_Kimlik=" & secilen_kayit(0), e
            baglanti , adOpenKeyset, adLockOptimistic, adCmdTable

            tablo("VeriTabani_Adi").Value = CStr(tb_adi.Value)
            tablo("VeriTabani_Soyadi").Value = CStr(tb_soyadi.Value)  f
            tablo("VeriTabani_Yasi").Value = CDbl(tb_yasi.Value)
            tablo.Update

            tablo.Close
            Set tablo = Nothing
            baglanti.Close                  g
            Set baglanti = Nothing

            tb_adi.Value = ""
            tb_soyadi.Value = ""
            tb_yasi = ""                    i

            For i = 0 To list_kayit.ListCount
                list_kayit.Selected(i) = False
            Next i
            Call kayit_oku      j
        End If
    End If
```

On yedinci aşama veri düzeltme işlemlerinin yapıldığı *on altıncı aşama*daki adımların benzerlerini içerir. En önemli farkı verilerin güncellenmemesidir, bunun yerine *Tablo.Delete* komutu kullanılarak tablodan seçilerek açılan veri silinir *(bkz. Şekil 7.27)*.

Şekil 7.27

VBA ile Veritabanları Uygulama Örneği İçin Sil Butonuna Tıklandığında Alt Yordam

```
Private Sub cmd_sil_Click()

    Dim baglanti As New ADODB.Connection
    Dim tablo As New ADODB.Recordset

    For i = 0 To list_kayit.ListCount
        If list_kayit.Selected(i) Then
            secilen_kayit = Split(CStr(list_kayit.List(i)), "-")
        End If
    Next i

     If IsEmpty(secilen_kayit) = True Then
        MsgBox "Lütfen listeden Seçim Yapınız"
     Else
       If UBound(secilen_kayit) = 3 Then
        baglanti.ConnectionString = baglantiyolu()
        baglanti.Open

        tablo.ActiveConnection = baglanti
        tablo.Open "Tablo1 Where VeriTabani_Kimlik=" & secilen_kayit(0),
        baglanti , adOpenKeyset, adLockOptimistic, adCmdTable
        tablo.Delete a

        tablo.Close
        Set tablo = Nothing
        baglanti.Close
        Set baglanti = Nothing

        tb_adi.Value = ""
        tb_soyadi.Value = ""
        tb_yasi.Value = ""

        For i = 0 To list_kayit.ListCount
            list_kayit.Selected(i) = False
        Next i

        Call kayit_oku

       End If
    End If
End Sub
```

Özet

Uygulama programlama arayüzü kavramını açıklamak.

Yazılım dillerinin tümünün öğrenilmesi ve uzmanlaşma mümkün değildir. Yazılım dillerinin çeşitliliği ve tekrarların önlenmesi için tüm yazılım dilleri ile çalışabilecek köprü vazifesi görecek yazılım diline ihtiyaç vardır. Tüm bu ihtiyaçları karşılamak üzere geliştirilen çözümlerden biri kütüphanelerdir. Kütüphanelerin temel kullanım amacı bir yazılım dili içerisinde tekrar kullanılabilir fonksiyonları toplamaktır. Microsoft Windows yapısında sık kullanılan kütüphanelerden biri dinamik link kütüphanesidir (Dynamic Link Library-dll). Uygulama Programlama Arayüzü (Application Programming Interface-API) bilgisayar programlamasında yazılım uygulamalarının bir birleri ile bütünleştirmek için oluşturulmuş rutinleri, protokolleri ve araçları içerir. API'ler yazılımları bir birlerine bağlayarak beraber çalışmalarına imkân sağlamaktadır. Yazılımları bütünleştirmenin yanında bilgisayar donanımları da API sayesinde kolay bir şekilde çalıştırılır.

VBA ile API işlemlerini tasarlamak.

Kullanıcılar Excel VBA fonksiyonlarının yetersiz kaldığı durumlarda Windows işletim sistemi yapısındaki kütüphanelerde bulunan fonksiyonları kullanmaktadır. VBA ile API işlemlerinde genellikle Windows işletim sistemi yapısında varsayılan olarak bulunan kütüphaneler ile bağlantı kurulur ve bu kütüphanelerde tanımlanmış fonksiyonlar kullanılır. Windows işletim sistemindeki Advapi32, Windows GDI+, Comdl32, Kernel32, Shell32, User32, Netapi32 ve Winspool sıkça kullanılan API kütüphaneleridir.

Advapi32 (Advanced Services) Windows işletim sistemi yapısı içerisinde çekirdek kayıtların (Windows registry) ve NT güvenlik sistemlerinin (NT Security) fonksiyonlarını içeren kütüphanedir. Windows GDI+ (Graphics Device Interface) Windows işletim sistemi yapısı içerisinde grafik temelli işlemler için hazırlanmış bir kütüphanedir. Comdl32 (Common Dialog Box Library) Windows işletim sistemi yapısındaki diyalog pencerelerinin yönetilmesi için kullanılan kütüphanedir. Windows işletim sistemi yapısında dosya ve sistem ile ilgili fonksiyonlar Kernel32 kütüphanesi ile yönetilir. Windows işletim sistemi yapısının ana kabuğu Shell32 kütüphanesinde yer almaktadır. Programların başlatılması, yazılım ikonlarının görüntülenmesi, dosya gezgini gibi fonksiyonların büyük bir bölümü Shell32 kütüphanesi altındadır. Excel VBA programı hazırlanırken pencerelerin yönetilmesi, klavyedeki kısa yol işlemlerinin yapılması, bir metnin kopyalama ve yapıştırma işlemlerinde User32 kütüphanesi kullanılır. Windows işletim sistemi yapısında ağ fonksiyonları için Netapi32 Kütüphanesi, yazdırma fonksiyonlarının yönetilmesi için ise Winspool kullanılmaktadır.

VBA ile API işlemlerinde fonksiyon kütüphanelerinin kullanılabilmesi için kütüphanelerin deklare edilmesi gerekmektedir. Deklarasyon deyimi ihtiyaç duyulan fonksiyonun bulunduğu kütüphanenin ve fonksiyonun yol ve içeriğinin tanımlandığı bir yordamdır. API deklarasyon deyimi sayesinde bağlantı kurulan kütüphanenin tümü değil sadece ihtiyaç duyulan fonksiyon yazılım içine çekilir.

Veri tabanı yönetim sistemlerini tanımlamak.

Veritabanları büyük miktardaki bilgileri depolamak için geliştirilen birbirleriyle ilişkili bilgilerin ihtiyaç duyulduğunda tekrar çağrılabilmesi için hazırlanmış dijital saklama alanlarıdır. Bilgisayar temelli kayıtların tutulması için hazırlanan veritabanları, veri tabanı yönetim sistemleri (Database Management System-DBMS) aracılığı ile oluşturulur ve yönetilir. Microsoft Access, Microsoft SQL Server, MySQL, Oracle, IBM DB2, Informix, PostgreSQL, Interbase ve Sysbase gibi yazılımlar veri tabanı yönetim sistemlerine örnek olarak verilebilir. Veritabanlarında genellikle tablolar bulunur ve bu tablolar bir birleri ile belirli alanlarda ilişkilidir. Veri tabanı ilişkileri birden bire, birden çoğa, çoktan bire ve çoktan çoğa olabilmektedir.

Veritabanlarındaki bilgiler sorgu cümleleri ile çağrılır. Sorgu cümleleri büyük miktardaki veriyi klasik dosya sistemlerine göre çok daha hızlı süzülebilir. Fakat veritabanlarının kurulum ve bakımı klasik dosya sisteminden daha pahalıdır. Ayrıca veri tabanı bileşenleri iyi tasarlanmazsa veri tabanı sorgu ve kayıt işlemleri başarısız sonuçlar üretebilmektedir.

AMAÇ 4

VBA ile veri tabanı işlemlerini ilişkilendirmek.

VBA yazılım platformunda veritabanlarına erişim için 3 farklı arayüz kullanılabilir. Bu ara yüzler ADO (ActiveX Data Objects), RDO (Remote Data Objects) ve DAO (Data Access Objects) şeklinde sıralanabilir.

ADO diğer bağlantı arayüzlerine göre daha basit ve esnek bir yapıda geliştirilmiştir. ADO ile veri tabanı bağlantısı yapılabilmesi için referans olarak *Microsoft ActiveX Data Objects X.X Library* Kütüphanesi eklenmelidir.

VBA ile ADO aracılığı ile veri tabanı işlemleri adımlarında sırasıyla veri tabanı ile bağlantı, tablo ile bağlantı, tablo işlemleri ve kapatma işlemleri yapılır. Veri tabanı ile bağlantı işlemlerinde; yeni bağlantı tanımı, bağlantı yolu atanması, bağlantının açılması işlemleri yapılır. Tablo ile bağlantı işlemlerinde yeni tablo tanımı, tabloya veri tabanı bağlantısının atanması, tablo kaynağının tanımlanması ve tablo açılması işlemleri yapılır. Tablo işlemleri; veri tabanından tablonun okunması, tabloya yazma ve tablo içerisindeki verileri silme işlemleridir. Son olarak kapatma işlemlerinde sırasıyla önce tablo, sonrasında ise bağlantı kapatılır.

Kendimizi Sınayalım

1. Temel kullanım amacı bir yazılım dili içerisinde tekrar kullanılabilir fonksiyonları toplayarak yazılımcının hizmetine sunmak olan, başka programlar tarafından çağırılarak çalıştırılan yapı aşağıdakilerden hangisidir?

a. Veri tabanı
b. Kütüphane
c. Kelime işlemci
d. Bellek (RAM)
e. Deklarasyon

2. Uygulama Programlama Arayüzü (API) ile ilgili aşağıdaki ifadelerden hangisi **yanlıştır?**

a. API'lerin kullanılabilmesi için kullanıcı formu oluşturulması gereklidir.
b. Rutinleri, protokolleri ve araçları içerir.
c. Yazılım uygulamalarının bir birleri ile bütünleştirmek için oluşturulmuştur.
d. Yazılımlar arasında köprü vazifesi görür.
e. Bilgisayar donanımları API kullanılarak daha kolay bir şekilde çalıştırılabilir.

3. Aşağıdakilerden hangisi Windows işletim sisteminde bulunan API kütüphanelerinden biri **değildir?**

a. Advapi32
b. Windows GDI+
c. Comdl32
d. CentOS
e. Shell32

4. Pencerelerin yönetilmesi, klavyedeki kısa yol işlemlerinin yapılması, bir metnin kopyalama ve yapıştırma işlemlerini yapan API kütüphanesi aşağıdakilerden hangisidir?

a. Comdl32
b. Windows GDI+
c. User32
d. Kernel32
e. Shell32

5. Public Declare PtrSafe Function GetUserName Lib "advapi32.dll" Alias _ "GetUserNameA" (ByVal lpBuffer As String, nSize As Long) As Long şeklinde ifade edilmiş bir API deklarasyon deyimi ile ilgili aşağıdaki ifadelerden yanlıştır?

a. Deklarasyon deyiminin geçerli olduğu yaşam alanı projenin tamamını içermektedir.
b. API köprüleme işlemi 64-bit'lik işletim sistemi yapısında çalıştırılabilir.
c. Bilgisayarın kullanıcı adını getirmeye yarayan GetUserName fonksiyonu için API deklarasyonu hazırlanmıştır.
d. API fonksiyonu Long türünden geri dönüş sağlar.
e. Advapi32 kütüphanesi yazdırma işlemleri için kullanılan kütüphanelerden biridir.

6. Aşağıdaki hangisi Windows işletim sistemi yapısı içerisinde grafik temelli işlemler için hazırlanmış bir kütüphanedir?

a. Advapi32
b. Windows GDI+
c. Shell32
d. Kernel32
e. Comdl32

7. Aşağıdakilerden hangisi veri tabanı yönetim sistemlerine (Database Management System-DBMS) örnektir?

a. Windows 10
b. Mac Os
c. Android
d. Microsoft SQL Server
e. Linux

8. Aşağıdakilerden hangisi veri tabanı tablo ilişkilerinden biri **değildir?**

a. Azdan çoğa
b. Birden bire
c. Birden çoğa
d. Çoktan bire
e. Çoktan çoğa

9. Aşağıdakilerden hangisi VBA yazılım platformunda veritabanlarına erişim için kullanılan arayüzlerden biridir?

a. EFO (End of File)
b. DEC (Digital Equipment Corporation)
c. RDO (Remote Data Objects)
d. ROM (Read Only Memory)
e. BIOS (Basic Input Output System)

10. Aşağıdakilerden hangisi hazırlanan yazılımlarda veri tabanı yönetim sistemleri kullanılmasının avantajlarından biri **değildir?**

a. Tabloda belirlenmiş bir anahtar girildiğinde anahtara bağlanmış diğer tablolardaki veriler de otomatik olarak çağrılabilmektedir.
b. Veri tabanı yönetim sistemleri ile veri tekrarları en aza inebilmektedir.
c. Büyük miktardaki veri klasik dosya sistemlerine göre çok hızlı süzülebilmektedir.
d. Veri tabanı yönetim sistemlerinde tablolardaki veriler ilişkilendirilebilmektedir.
e. Kurulum ve bakımı klasik dosya sistemlerinden daha pahalıdır.

Kendimizi Sınayalım Yanıt Anahtarı

1. b	Yanıtınız yanlış ise "Uygulama Programlama Arayüzü (API)" konusunu yeniden gözden geçiriniz.
2. a	Yanıtınız yanlış ise "Uygulama Programlama Arayüzü (API)" konusunu yeniden gözden geçiriniz.
3. d	Yanıtınız yanlış ise "Windows İşletim Sistemindeki VBA API Kütüphaneleri" konusunu yeniden gözden geçiriniz.
4. c	Yanıtınız yanlış ise "Windows İşletim Sistemindeki VBA API Kütüphaneleri" konusunu yeniden gözden geçiriniz.
5. e	Yanıtınız yanlış ise "API Deklarasyon Deyimi" konusunu yeniden gözden geçiriniz.
6. b	Yanıtınız yanlış ise "Windows İşletim Sistemindeki VBA API Kütüphaneleri" konusunu yeniden gözden geçiriniz.
7. d	Yanıtınız yanlış ise "Veri Tabanı İşlemleri" konusunu yeniden gözden geçiriniz.
8. a	Yanıtınız yanlış ise "Veri Tabanı İşlemleri" konusunu yeniden gözden geçiriniz.
9. c	Yanıtınız yanlış ise "VBA ile Veri Tabanı İşlemleri" konusunu yeniden gözden geçiriniz.
10. e	Yanıtınız yanlış ise "Veri Tabanı İşlemleri" konusunu yeniden gözden geçiriniz.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1

Mac işletim sistemleri Windows işletim sistemi kütüphanelerini (dll) çalıştırmadıkları için Windows için hazırlanmış yazılımlar Mac'lerde çalışmamaktadır.

Sıra Sizde 2

Hem 32 hem de 64 bitlik Windows işletim sistemi için API deklarasyonu hazırlama için *#If VBA7 Then* koşul cümlesi ile iki farklı deklarasyon hazırlanır. Windows işletim sistemi32-64 bitlik deklarasyon deyimini seçerek kullanır.

Yararlanılan ve Başvurulabilecek Kaynaklar

Alp, S., Özdemir, S. ve Kilitci, A.(2011). Veri tabanı yönetim sistemleri. İstanbul:Türkmen Kitabevi.

Microsoft. (2015, Aralık 01). Learn to Develop with Microsoft Developer Network. https://msdn.microsoft.com/en-us/library/windows/desktop/hh920508(v=vs.85).aspx adresinden alındı.

8

Amaçlarımız

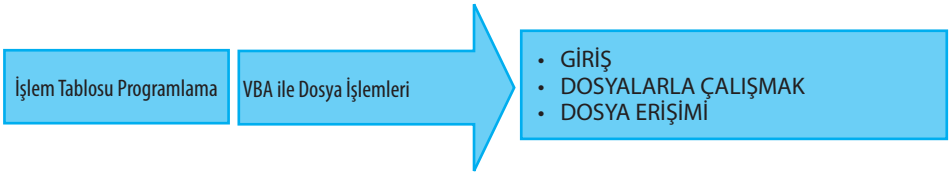
Bu üniteyi tamamladıktan sonra;

- VBA dosya sistemlerini tanımlayabilecek,
- VBA dosya işlemlerini açıklayabilecek,
- Dosya üzerine veri ekleme işlemini gerçekleştirebilecek,
- Dosya kontrolü işlemini tanımlayabilecek,
- Dosya uzunluğunu belirleyebilecek,
- Dosyadan veri okuma işlemlerini uygulayabileceksiniz.

Anahtar Kavramlar

- VBA Dosya Türleri
- Ardışık Erişimli Dosya
- Rastgele Erişimli Dosya
- İkili Dosya
- Dosya Açma
- Veri Ekleme
- Dosya Kapatma
- Dosyadan Veri Okuma
- Dosya Kontrolü
- Dosya Uzunluğu

İçindekiler



VBA ile Dosya İşlemleri

GİRİŞ

MS Excel çalışmaktayken kendi verilerinin depolanacağı bir disk alanına sahiptir. Ancak bazı durumlarda dış kaynaklı verileri MS Excel içerisine aktarmak ya da yapılan işlemlerden sonra elde edilen verileri dış bir kaynağa (bilgisayar içerisinde farklı tipte bir dosya) aktarmak gerekli olabilir. VBA özellikleri, MS Excel yazılımında, neredeyse tüm programlama dillerinde ortak bir özellik olarak bulunan, dış kaynaklara erişip, içlerinde depolanan verileri içe aktararak çalışmak ya da oluşturulan verileri dış kaynaklara depolamak fonksiyonları bulunmaktadır. Kitabın son ünitesinde, MS Excel yazılımı ile çalışırken disk sistemi üzerinde farklı bir dosya açmak, dosya içeriğindeki verileri okumak ya da dosya içeriğine veri yazmak ve dosyaları kapatmak ile ilgili işlemler ve fonksiyonlar ele alınacaktır. Daha önceki ünitelerde açıklandığı üzere MS Excel yazılımı, yardımcı program geliştirme ortamı olarak VBA (Visual Basic for Applications) platformu aracılığıyla, MS Visual Basic programlama dilini kullanarak çeşitli isteğe bağlı uygulamaların yazılmasına olanak sağlamaktadır. Bu sebeple anlatılacak olan dosya işlemleri, MS Visual Basic programlama dilinin ve VBA yazılım geliştirme ortamının özellikleri olacaktır.

MS Excel yazılımı dâhilînde dış kaynaklı dosya erişimi, genel kullanımlar göz önünde bulundurulduğunda, dış kaynaklı bir dosyadan MS Excel içine veri aktarmak için okuma işlemi ve dış kaynaklı bir dosyaya, MS Excel verisi aktarmak için yazma işlemi olmak üzere iki farklı dosya erişimi fonksiyonundan bahsedilebilir. Dosyadan okuma fonksiyonları aracılığıyla, el ile oluşturulan ya da MS Excel dışında bir yazılımın oluşturduğu veri kaynakları, otomatik olarak okunup içe aktarılarak, MS Excel dâhilînde kullanılabilir olacaktır. Ayrıca, MS Excel ile yapılan işlemler sonrasında elde edilen veriler de belirli bir formatta dışa aktarılarak, bu verilerin farklı yazılımlar tarafından kullanılması sağlanabilir. Üzerinde işlemler yapılan veri grubu dış kaynaklı bir dosyaya, çoğu zaman da kullanılacak diğer yazılım ile uyumlu bir dosyaya yazdırılarak kullanımı sağlanır.

Verileri kalıcı olarak saklamak için kullanılan dosyalar, farklı yazılımlarla kendilerine özgü biçimde oluşturulabilirler. Dosya türleri, isimlerinin noktadan sonra yazılan uzantılarında bulunan harflerle oluşturuldukları yazılımı temsil ederler (Örneğin MS Excel için .xlsx, .bas, .docx vb.).

MS Excel sayfası dışında dış kaynaklı bir dosya ile veri alışverişi işlemi yapılacaksa öncelikle MS Visual Basic programlama dili ve VBA platformu tarafından desteklenen dosya türlerini belirlemek gereklidir. MS Visual Basic programlama dili aşağıda listelenen üç farklı türdeki dosyayı desteklemektedir.

1. Ardışık Erişimli Dosyalar: Sadece metin depolamak için kullanılan ardışık erişimli dosyalar, VBA ortamında en çok karşılaşılan dosya türleri olmakla birlikte, dosya içindeki her karakterin bir metin karakterini ya da metin formatlama karakterini (Tab, yeni satır, fonksiyon tuşları vb.) temsil ettiği dosya türleridir. Bu tür dosyalarda sayılar numerik değerler olarak değil, numara karakterleri olarak

depolanırlar. Her karakterin bir bayt (byte) yer kapladığı bu dosya türünde sayısal veriler sütun hâlinde depolanırlar. Bu nedenle sayısal değerlerin depolanması istenen durumlarda diğer dosya türlerinden faydalanmak daha kolay kullanım sunacaktır. Ardışık erişimli dosyalar baştan sona doğru okunur. Bir ardışık erişimli dosya ile okuma ve yazma işlemleri aynı anda yapılmak isteniyorsa üzerinde okuma işlemlerinin yapılacağı bir dosya ve üzerine yazma işlemlerinin gerçekleştirileceği bir başka dosya olmak üzere iki ayrı dosya ile çalışmak gereklidir. Dosya içerisindeki verilerin boyutunun küçük olduğu durumlarda ise tüm içerik geçici belleğe aktarılarak işleme tabi tutulabilir. Sonrasında ise eski verilerin üzerine güncellenen yeni veriler yazılarak aynı anda iki dosya ile çalışılma zorunluluğu ortadan kalkacaktır.

2. Rastgele Erişimli Dosyalar: Ardışık erişimli dosyalarda bahsedilen verilerin belleğe aktarılarak işlenmesi ve daha sonra eski verilerin üzerine yazılması işlemine gerek duyulmayan ancak verilere sık erişim gereken durumlarda, rastgele erişimli dosyalardan faydalanılabilir. Metin verilerini karakter başına bir bayt yer kaplayacak şekilde karakterler şeklinde depolayan bu tür dosyalar, istenildiğinde numerik verileri de doğal sayısal biçimlerde saklama imkânı sağlamaktadır. Rastgele erişimli bir dosya, komut istemi yardımıyla görüntülenebilir. Fakat sadece metin kısımları okunabilir olacaktır. Rastgele erişimli dosyalar, kayıt adı verilen eşit uzunluktaki verileri saklamak için uygundur. Düzenli bir şekilde oluşturulan kayıtlar sayesinde dosya içerisinde herhangi bir kaydı bulmak ve üzerinde işlem yapmak kolaylaşmaktadır. Tüm kayıtlar aynı uzunlukta olacağı için indeksler ile erişim işlem gerçekleştirilir. Ardışık erişimli dosyaların aksine rastgele erişimli dosyalar, aynı anda okumak ve yazmak için açılabilir. Sadece bir ya da birkaç kaydın değiştirilmesi gerektiğinde, diğer kayıtlar etkilenmeden bu işlem kolaylıkla yapılabilir.
3. İkili Dosyalar: Ardışık erişimli dosya yapısına benzer şekilde çalışan ikili dosyalar, içlerinde saklanan verilerin tipleri hakkında bir düzenlemeye ihtiyaç duymazlar. Veriler dosyaya karakterler olarak değil ikili sayısal değerleriyle yazılırlar. Bu nedenle ikili dosyalar veriyi daha bütünleşik bir formatta ancak insan gözü ile bir bakışta anlaşılamayacak şekilde depolarlar. Müzik, resim ve video dosyaları gibi formatlar ikili dosyalar şeklinde saklanırlar.

SIRA SİZDE 1

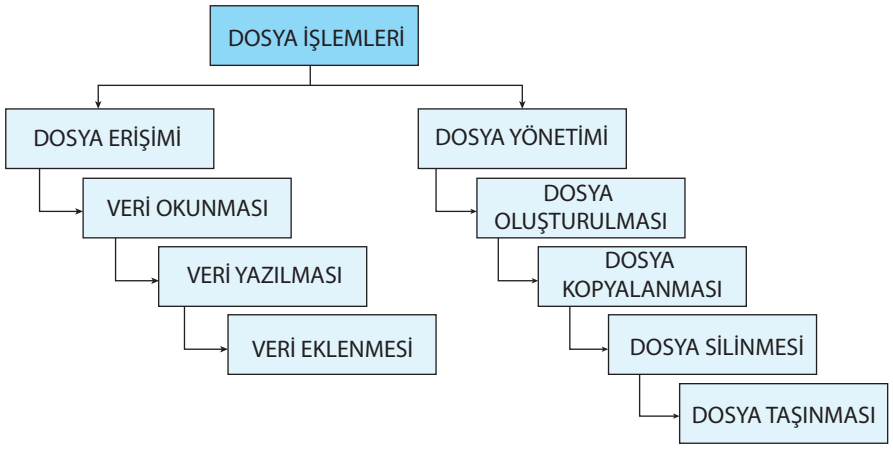
MS Visual Basic programlama dili tarafından desteklenen üç dosya çeşidini sıralayınız.

DOSYALARLA ÇALIŞMAK

Dosyalarla çalışmak için öncelikle dosyalarla yapılan işlemler konusunda bilgi sahibi olmak gereklidir. Bu işlemleri Dosya Erişimi ve Dosya Yönetimi şeklinde sınıflandırmak doğru olacaktır. Resim 8.1'de görüleceği gibi bu kategorilerin de alt başlıkları bulunmaktadır.

Resim 8.1

Dosya İşlemleri



Dosya yönetimi, dosyanın oluşturulması, kopyalanması, silinmesi ve taşınması işlemlerinden oluşmaktadır. Herhangi bir yazılım ile oluşturulan veriler, bir yere depolanmadan önce bellekte tutulur. Eğer işlemlerden elde edilen veriler MS Excel sayfası yerine başka bir dosya üzerinde depolanacaksa öncelikle depolamanın yani bir anlamda kaydetmenin üzerine yapılacağı dosya oluşturulmalıdır. Dosya oluşturma işlemi bilgisayarın depolama ünitelerinden belirlenen bir tanesi üzerinde bir dosya yaratmak olarak da tanımlanabilir. Dosya oluşturulduktan sonra yazılım ortamı ile açılarak, ünitenin devamında anlatılacağı gibi üzerine veri depolanabilir. Dosya oluşturmanın dışında, dosyanın farklı bir depolama birimine bir nüshasının daha oluşturulması ise dosyanın kopyalanması olarak adlandırılır. Dosya, üzerinde kayıtlı bulunduğu depolama ünitesi, dizin, klasörden alınarak başka bir yere taşınabilir. Her ne kadar günümüz teknolojisi büyük sığalarda veri depolanmasına izin verse de depolama ünitelerinin sınırlı bir kapasitesi vardır. Bu sığanın verimli kullanılması amacıyla ya da güvenlik gerekçesiyle işlemi sona ermiş dosyaların silinmesi de dosya yönetimi özelliklerinden bir tanesidir.

Dosya erişimi ise her ne kadar dosya üzerinden veri okunması, dosyaya veri yazılması ve/veya veri eklenmesi olarak bağımsız işler hâlinde tanımlansa da ardışık olarak gerçekleştirilen üç aşamalı işlemlerin bütününü kapsamaktadır.

- Dosyanın Açılması: Dosyanın oluşturulması işlemi tamamlandıktan sonra, üzerine veri yazma işlemi yapılabilecek biçimde dosyanın açılmasını sağlama işlemidir. Üzerinde işlem yapmak üzere dosyanın açılması için *Open* komutu kullanılır.
- Dosyanın İşlenmesi: Bir dosya, içinden veri okumak, üzerine veri yazmak ya da bu işlemlerin her ikisini de aynı anda gerçekleştirmek amacıyla açılabilir. Veri okuma/yazma ile ilgili bu işlemler, dosyanın işlenmesi olarak adlandırılır. Dosyanın işlenmesi sırasında yapılacak işlemler için dosya biçimi de önemlidir.
- Dosyanın Kapatılması: Dosyanın kapatılması, işletim sistemi tarafından bu dosya için ayrılan bellek parçasının serbest bırakılmasını ve son hali oluşturulan dosyanın sabit sürücü üzerinde kalıcı olarak depolanmasını sağlayacaktır. Üzerinde işlem yapılan dosyanın kapatılması için *Close* komutu kullanılır. Eğer üzerinde çalışılan birden fazla dosya varsa *Reset* komutu ile tamamının kapatılması mümkündür.

Kısaca tanımlanan bu işlemler, MS Visual Basic ortamında bazı komutlar ile gerçekleştirilmektedir. Bu komutlar ve kullanım şekillerinden aşağıda bahsedilmiştir.

Dosya Açmak

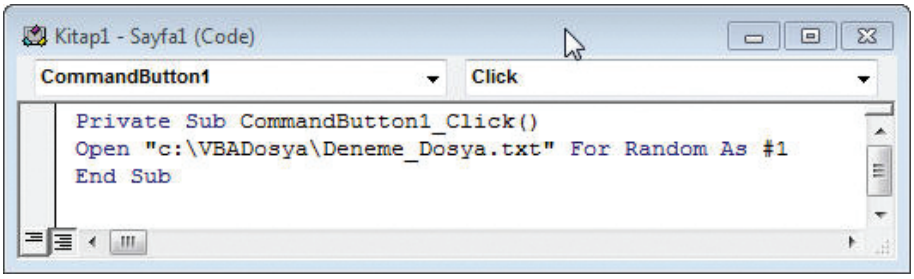
Bir dosyadan veri okuma ya da dosyaya veri yazma işlemi yapılmadan önce ilgili dosya açılmalıdır. Eğer dosya sabit disk üzerinde mevcut değilse yani henüz yaratılmamış bir dosya ile çalışmak istenirse, öncelikle dosya yaratılmalıdır. Dosyalar açılırken **Open** komutundan faydalanılır. Open komutunun en yalın kullanımı şu şekildedir:

```
Open Dosya_Adı As #1
```

Ancak bu komut kullanılırken çoğu zaman *For, Access, As, Len* gibi yardımcı terimlere de başvurulur. Yukarıdaki ifadede Dosya_Adı terimi, depolama ünitesinde kayıtlı bulunan ya da yaratılmak istenilen dosyanın ismini ifade etmektedir. Örnek olarak, daha önceki bilgilere başvurarak boş bir MS Excel dosyası açılsın ve sayfa üzerine Geliştirici sekmesinden Ekle butonu kullanılarak Active X Denetimleri altında bulunan bir Komut Butonu (Command Button) eklensin. Yaratılan Komut Butonu üzerine çift tıklandığında VBA penceresi otomatik olarak açılarak, Komut Butonuna özel alt yordam kodlama için hazır hale gelecektir. Ön hazırlık amacıyla bilgisayarın sabit sürücüsü C:\ dizini üzerine *VBADosya* isminde bir klasör oluşturulsun. Yeni oluşturulan klasörün içi boş olsun. Kod sayfasına Resim 8.2'deki kod eklendiğinde ve Komut Butonu tıklandığında klasör içerisinde belirtilen dosya adı olan *Deneme_Dosya* adı ile bir metin dosyası oluşturulacaktır. Resim 8.3 bu metin dosyasını göstermektedir.

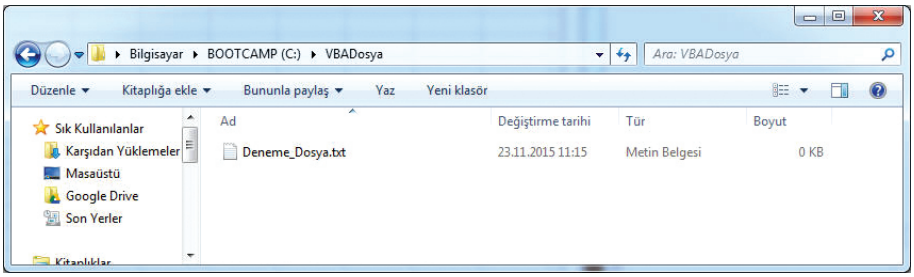
Resim 8.2

Komut Butonu Kod Ekranı



Resim 8.3

VBADosya Klasör İçeriği

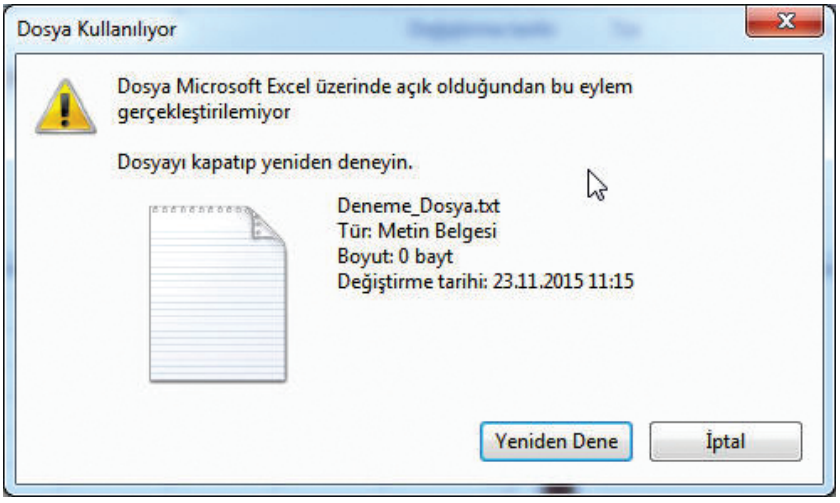


Dosya aynı zamanda kullanım için açık hâle gelecektir. Dosyanın açık olduğu bir pencere yardımıyla görülmese de dosya bulunduğu dizinden silinmeye çalışılırsa Resim 8.4'teki gibi uyarı ile karşılaşılır. Bu durumda dosyanın silinmesi için öncelikle dosyanın kapatılması gerekmektedir.

VBA ortamında açılan dosyaların tümüne tanımlayıcı tekil bir numara atanır. Açılan dosya için dosya tanımlayıcı tekil numarası #1 olarak belirtilmiştir. Bundan sonraki işlemlerde dosya tanımlayıcı tekil numarası olarak bu sayı kullanılacaktır. Bu tanımlayıcı

tekil numaralar sayesinde birden fazla dosya ile aynı anda çalışmak mümkündür. **FreeFile** fonksiyonu ile birlikte kullanıldığı durumlarda boşta bulunan en küçük numaranın dosya tanımlayıcı tekil numarası olarak atanması sağlanır.

Resim 8.4



Dosya Silinemez Uyarısı

Dosya açmak için kullanılan Open komutunun tüm özellikleri kullanılarak yazım biçimi ise aşağıdaki gibidir:

Open Dosya_Adı For Dosya_Tipi [Access Erişim_Tipi][Lock_Tipi] As [#] Dosya_No [Len Kayıt_Uzunluğu]

Dikkat edilecek olursa dosya açmak için kullanılan farklı terimler bulunmaktadır. Bu terimlere kısaca göz atılacak olursa;

Dosya_Adı: Sabit sürücü gibi bir depolama biriminde dosyayı tanımlamak için gerekli olan isimdir. Açılması istenen dosya tipine göre uzantı verilerek dosyanın oluşturulması sağlanır.

Dosya_Tipi: Bu terim ile açılacak olan dosyanın türü ve yapılacak işlem belirlenir. Argüman değerleri istenilen özelliklere göre değişmekle birlikte;

- Input: Dosya sadece Giriş yani okuma işlemleri için açılır.
- Output: Dosya sadece Çıkış yani yazma işlemleri için açılır.
- Append: Dosyanın içeriğine yeni veriler eklenmek istendiğinde kullanılır.
- Random: Dosya rastgele erişim yani açıldıktan sonra kayıt okumak ve yazmak için açılır.
- Binary: Dosyanın ikili kipte açılmasını sağlar.

Input, Output ve Append terimleri ardışık erişimli dosyalarda kullanmak için oluşturulmuş terimlerdir. Random terimi rastgele erişimli dosyalarda, Binary terimi de ikili dosyalarda kullanılır. Daha önce de bahsedildiği gibi ardışık erişimli dosyalarla çalışırken aynı anda hem okuma hem de yazma işlemi yapmak mümkün değildir. Ardışık erişimli dosyalarda, bu sebeple okuma, yazma ve üzerine veri ekleme ifadeleri ayrı ayrı tanımlanmıştır. Ardışık erişimli dosyalarla ilgili bir başka zorluk da dosya çıkış amaçlı açıldığında, dosyada bulunan eski verilerin tümünün silinmesi ve yeni verilerin yazılması durumudur. Dosya bir kere çıkış amaçlı açıldığında, daha önce yazılan tüm veriler, herhangi bir uyarı olmadan silinecektir.

Erişim_Tipi: Bu terim yalnızca Rastgele Erişimli Dosyalarda kullanılmaktadır ve dosyanın okumak (Read), yazmak (Write) ya da her iki işlemi (Read/Write) birden yapmak için açılmasını sağlar. Bir dosya okunmak için açıldığında, yanlışlıkla dahi olsa üzerindeki verilerin değiştirilmesine imkân yoktur. Bu sebeple güvenlik açısından bir dosyanın değiştirilmemesi gerekiyorsa yalnızca okuma işlemi için gerekli terim olan (Read) ile açmak doğru olacaktır.

Lock_Tipi: Bu terim ile dosyanın açık tutulduğu süre içerisinde işletim sisteminin hakları belirlenmektedir. İşletim sistemleri üzerinde aynı anda birden fazla uygulama çalışması mümkündür. VBA dışında bir uygulama da kullanılan ve açık olan dosyaya erişmek ve üzerinde işlem yapmak isteyebilir. Lock_Tipi terimi alacağı değerler ile bu hakları kilitler ya da serbest bırakabilir. Ağ ortamında kullanılan bilgisayarlarda Lock_Tipi teriminin önemi artmaktadır. Farklı bilgisayarlardaki uygulamaların dosyaya erişim yetkileri bu sayede düzenlenir. İfadenin alacağı değerler aşağıda belirtilmiştir:

- Shared: Diğer uygulamalar ile dosyanın paylaşılmasını sağlar.
- Lock Read: Dosyanın okumaya kilitlenmesini sağlar, bu sayede diğer uygulamalar dosya üzerindeki verilere hiçbir şekilde erişemezler.
- Write Lock: Dosyanın yazmaya karşı kilitlenmesini sağlar. Diğer uygulamalar dosya üzerindeki veriler üzerinde herhangi bir değişiklik yapamazlar.
- Lock Write Read: VBA dışında herhangi bir uygulamanın dosyaya erişimi yasaklanır.

As: Daha önce de bahsedildiği gibi, dosya açma işlemi sırasında her bir dosyaya birbirinden farklı olmak kaydıyla bir tanımlayıcı tekil numara verilmekte, bu sayede hem üzerinde çalışılan dosyalara erişim kolaylaşmakta hem de birden fazla dosya ile aynı anda çalışmak mümkün olmaktadır. As #1 şeklinde açılan dosya, yine sadece #1 tanımlayıcı tekil numarasıyla dosya ismi verilmeden kolayca kapatılabilmektedir.

Len Kayıt_Uzunluğu: Bu terim, yalnızca rastgele erişimli dosyalar için kullanılmaktadır. Bu tür bir dosya kullanımında MS Visual Basic yazılım ortamı, dosya içerisindeki kayıt uzunluğu veya kayıt yapısı ile ilgili bir bilgi barındırmaz. Bu sebeple rastgele erişimli bir dosya açılmadan önce kayıt yapısı hakkında bilgi sahibi olmak gereklidir. Kayıt_Uzunluğu parametresi, kayıt katarının kapladığı toplam hafıza değerini ifade eder. Bu değer programcı tarafından hesaplanabilir ya da Len (Kayıt_Uzunluğu) terimi ile yazılım tarafından otomatik olarak hesaplanması sağlanabilir.

Dosya Kapamak

Bir dosya açık iken kullanımda olacağından silinemez. Resim 8.3'te bu durumun bir örneği gösterilmiştir. VBA tarafından okunmak, yazılmak ya da her iki işlev için açılan dosya, işi bittikten sonra kapatılmalıdır. Dosyayı kapatmak için iki ifade kullanılabilir. Bunlardan ilki;

```
Close # [Dosya_Numarası]
```

ifadesidir. Hatırlanacak olursa üzerinde çalışılan dosyayı belirleyebilmek ya da birden fazla dosya ile aynı anda çalışmak amacı ile her bir dosyaya birbirinden farklı tanımlayıcı tekil bir numara atama işlemine ihtiyaç duyulur. Dosyanın açılması esnasında kullanıcı tarafından ya da yazılım ortamının desteğiyle FreeFile terimi tarafından atanan bu tanımlayıcı tekil numaraya, kapatma işleminde de gerek duyulmaktadır. Close komutundan sonra kullanılan # [Dosya_Numarası] ifadesi ile kapatılmak istenen dosya tanımlayıcı tekil numarası belirtilerek dosya kapama işlemi gerçekleştirilebilir.

Yukarıda oluşturulan Deneme_Dosya.txt dosyasını kapatmak için gerekli MS Visual Basic kodunu oluşturunuz.

SIRA SİZDE 2

Unutmamak gerekir ki kapatma işlemi, üzerinde çalışılan dosyanın yalnızca kullanıma kapatılmasını sağlar. Dosyanın silinmesi ya da dosya içerisinde bulunan kayıtların temizlenmesi gibi bir işlem yapmaz.

Dosya kapatma işlemi, o anda açık bulunan tüm dosyaların kullanıma kapatılması istendiğinde **Reset** komutu kullanılır. Bu komut, Open komutu ile açılmış ve o anda yazılım ortamı tarafından kullanılan tüm açık dosyaların herhangi bir tanımlayıcı tekil numara belirtmeden kapatılmasını sağlamaktadır. Reset komutu o anda açılmış olan tüm dosyalar için geçerli olduğundan, bu komut ile o anda açık bulunan dosyalardan herhangi birinin kapatılarak diğerlerinin açık durumda bulunmayı sürdürmesi amacı ile kullanılamaz. İfade kullanıldığı an açık olan tüm dosyalar kullanıma kapatılacaktır.

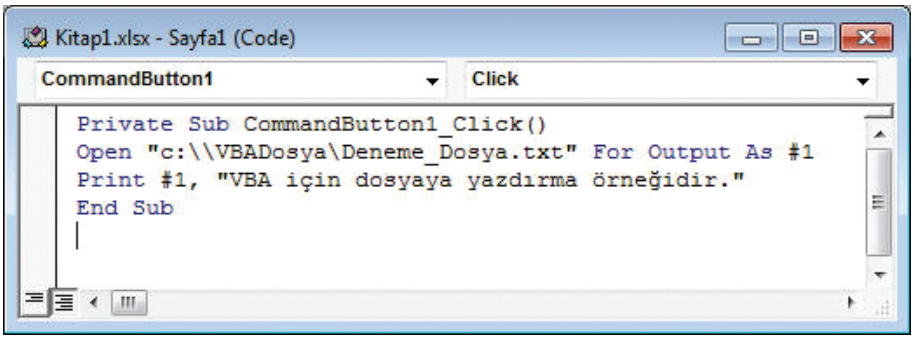
Reset komutu tüm açık olan dosyaları kullanıma kapatır. Tek bir dosya kapatmak için Close komutu ve dosya tanımlayıcı tekil numarasından faydalanılmalıdır.

DOSYA ERİŞİMİ

Dosya erişimi, açılan bir dosya üzerinde okuma, yazma ya da okuma ve yazma işlemlerini yapabilmek anlamına gelmektedir. Yaratılan dosya öncelikle erişim tipine uygun şekilde açılmalıdır. Dosya ancak açıldıktan sonra dosya içerisindeki kayıtların okunması ve işlenmesi mümkündür. Dosya içerisindeki kayıtları değiştirmek, kayıt eklemek ya da boş bir dosyaya yazma işlemleri ise dosya gerekli yetkilerle açıldıktan sonra gerçekleştirilmesi gereken işlemlerdir. Bu bölümde, dosya erişim işlemlerini gerçekleştirmek için kullanılan komutlar açıklanacaktır.

Print Komutu

Ardışık erişimli dosyalara veri yazmak için kullanılan Print komutu aslında bir MS Excel hücresi ya da bir VBA Formu üzerine veri yazdırılmasına benzemektedir. Bilindiği gibi bu komut ile tırnak içerisine alınmış bir karakter dizisi, belirlenen hedefe yazdırılır. Aynı mantıkla daha önce yazma yetkisi ile açılmış bir dosyanın tanımlayıcı tekil numarası kullanılarak ilgili dosya işaret edilirse tırnak içerisinde yazılan karakter dizesi dosya içerisine yazdırılacaktır. Örneklendirmek için Resim 8.5'te görülen yazdırma kodu çalıştırıldığında, yaratılan dosyanın çıktısı Resim 8.6'da incelenmelidir.

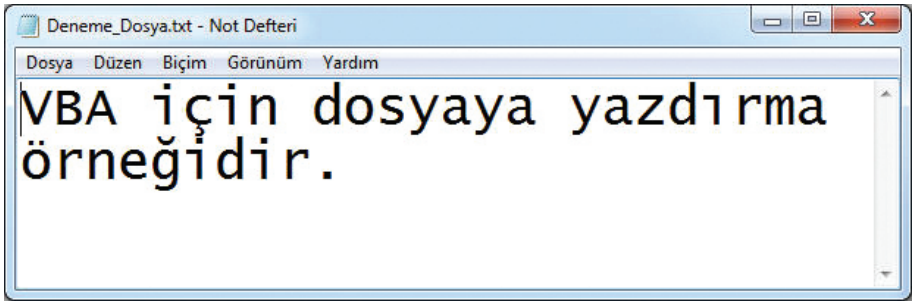


Resim 8.5

Print Komutu Kod Örneği

Resim 8.6

Deneme_Dosya.txt İsimli Dosyanın Kod Çalıştırıldıktan Sonraki Görünümü.

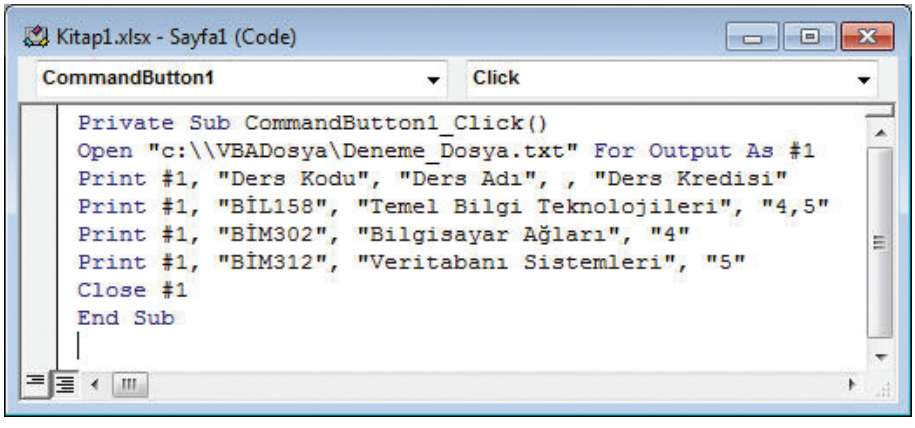


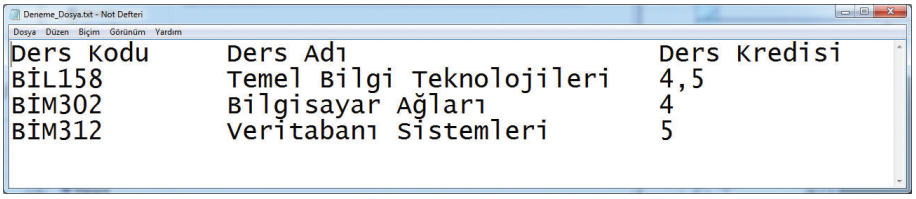
Print komutunda satır aralarını ayarlamak için noktalı virgül ve virgülden faydalanılır. Noktalı virgül ile ayrılan değerler önceki değerin bitiminden hemen sonra, virgül ile ayrılan değerler ise baskı sınırı olarak adlandırılan, 14 sütundan sonra yeni değerin yazdırılacağını göstermektedir. Her virgül karakteri, değerleri, bir sonraki 14 baskı sınırına ilerletmektedir. Bu sebeple gerekli durumlarda birden fazla virgül yan yana kullanılabilir. Sütunların düzgün hizalanması için Print komutu kullanılırken yazı tipinin de eşit aralıklara sahip karakterli bir yazı tipi olması (Courier New, Lucida Consolas vb.), düzenin sağlanabilmesi için önemlidir. Satırlara yazılacak değerlerin sütunlar hâlinde düzgün görüntülenmesi istendiğinde, Tab ya da Tab(n) tuşundan faydalanılır. Resim 8.7'de virgül kullanımına örnek bir kod ve Resim 8.8'de bu kodun dosya üzerine yazdırdıkları görüntülenmektedir.

Dikkat edilmesi gereken noktalardan biri de Print komutu ile yazdırılan metinlerin kod her çalıştığında eski verilerin üzerine tekrar yazılacağıdır. Resim 8.7'de bulunan kod bir butonun tıklama işlemine atanmıştır. Buton tıklandığında işlem yapılacak ve gerekli dosyaya komutlar aracılığıyla gönderilen değerler yazdırılacaktır. Bu kayda başka bir nesne eylem ikilisi ile farklı bir değer yazdırılması durumunda daha önce dosya üzerine yazdırılmış bulunan değerler silinecek ve yeni değerler onların yerini alacaktır.

Resim 8.7

Print Komutunda Virgül Kullanımı Örneği



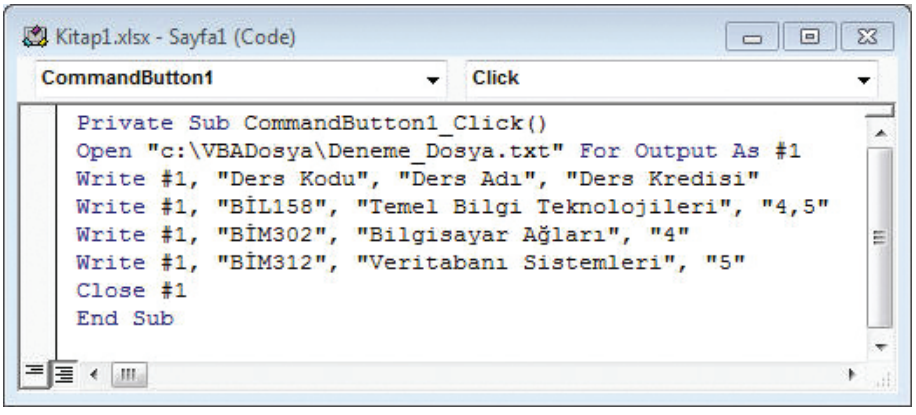


Resim 8.8

Virgül Kullanılarak Yazdırılmış Print Komutu Çıktısı

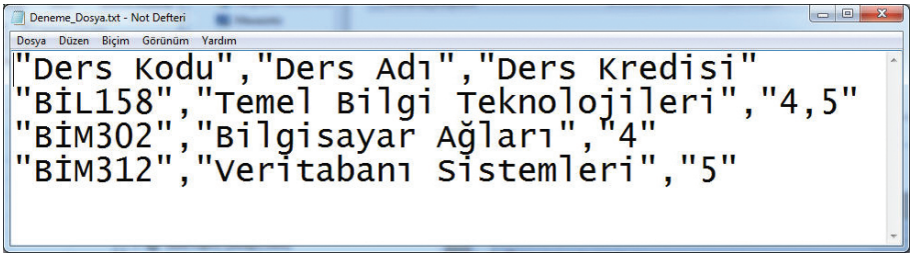
Write Komutu

Write komutu tıpkı Print komutu gibi ardışık erişimli dosyaya veri yazdırmak için kullanılan bir komuttur. Dosyaya yazdırılacak olan veriler virgül ile ayrılır ancak burada Print komutundan farklı olarak virgül, aralık tanımlamak için değil, değerleri birbirinden ayırmak için kullanılmaktadır. Resim 8.9'da Print için tanımlanan değerlerin Write komutu ile yazıldığı kod parçacığı ve Resim 8.10'da bu kodun dosya üzerindeki çıktısı görüntülenmektedir.



Resim 8.9

Write Komutu Kod Örneği



Resim 8.10

Write Komutu Çıktısı

Print ve Write, ardışık erişimli dosyalara veri yazdırmak için kullanılan komutlardır. Rastgele erişimli dosyalar ve ikili dosyalar için kullanılan komut ise Put komutudur.

DİKKAT

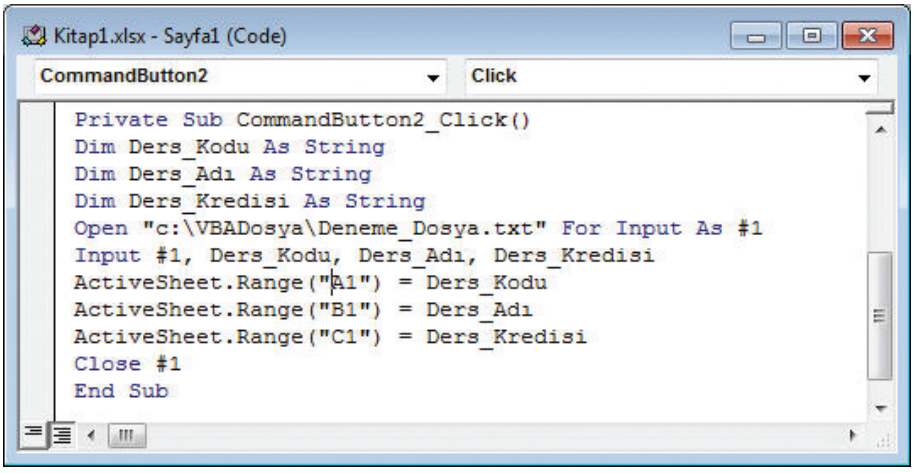
Input İfadesi

Input komutu, ardışık erişimli bir dosyadan okunan verileri değişkenlere atamak için kullanılır. Komuttan sonra içerisinden veri okunacak dosya tanımlayıcı tekil numarası yazılmalı, daha sonra ise okunan verilerin atanacağı değişkenler belirtilmelidir. Bu değişkenler daha önceden tanımlanmış yerel ya da global değişkenler olabilir. Input komutu ile değerler tek tek atanabilir. Bir satır hâlinde tüm veriler okunmak isteniyorsa ilerleyen konuda anlatılacak olan Line Input ifadesinden faydalanılmalıdır.

Örnek olarak, Write ile üzerine veri yazılmış olan Deneme_Dosya.txt dosyasının ilk satırındaki değerler okunarak MS Excel sayfasındaki A1, B1 ve C1 adresli hücrelere yazdırılsın. Bunun için ikinci bir buton kullanarak ilgili işlemler bu buton aracılığıyla yapılsın. Resim 8.11'de görüntülenen kod parçacığı incelenecek olursa değerler dosyadan okunduktan sonra ilgili hücrelere yazdırılmadan önce, yaratılan üç adet dinamik değişkene atanmakta ve sonrasında bu değişkenlerin değerleri hücrelere yazdırılmaktadır. Bu kod parçacığının çıktısı Resim 8.12'de MS Excel sayfası üzerinde görülmektedir. Input komutu, tıpkı Print ve Write komutlarında olduğu gibi tekrar ettiğinde diğer kayıtları da okuyarak belirtilen yerlere yazdıracaktır.

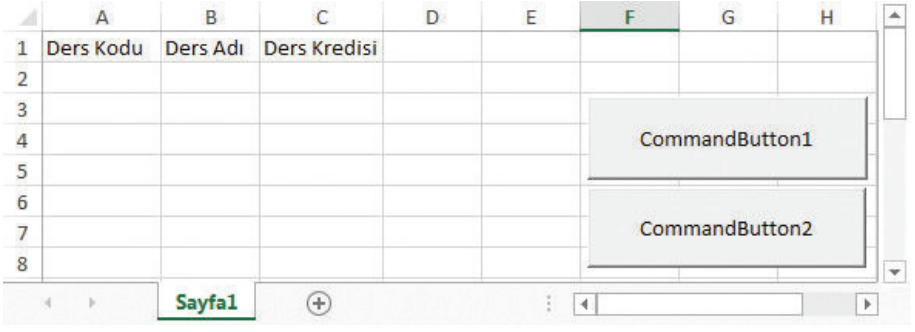
Resim 8.11

Input Komutu Kod Örneği.



Resim 8.12

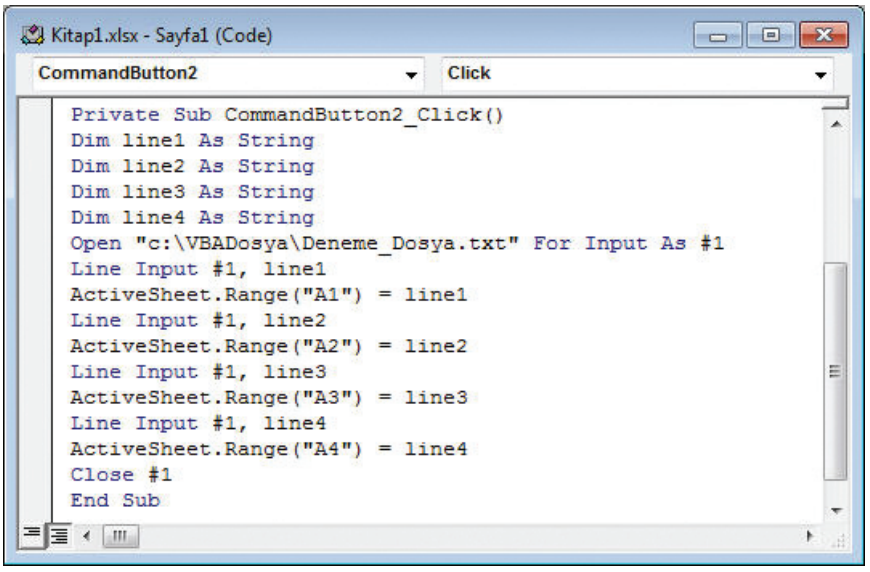
MS Excel Sayfası



Line Input Komutu

Ardışık erişimli dosyalardaki verileri satır satır okumayı sağlayan komuttur. Input komutu ile tek tek okunarak ayrı değişkenlere atanan veriler yerine dosyaya Print ya da Write komutları ile kaydedilmiş tüm bir satırı okumak amaçlanmaktadır. Resim 8.13'te, dört ayrı değişkene atanan dört satırlık Deneme_Dosya.txt verisi, A1 hücresinden başlayarak aşağıya doğru yazdırılmıştır. Resim 8.14'te bu kod çalıştırıldığında yapılan işlem görülmektedir. Dikkat edilecek olursa dosya yalnızca okumak için açılmıştır.

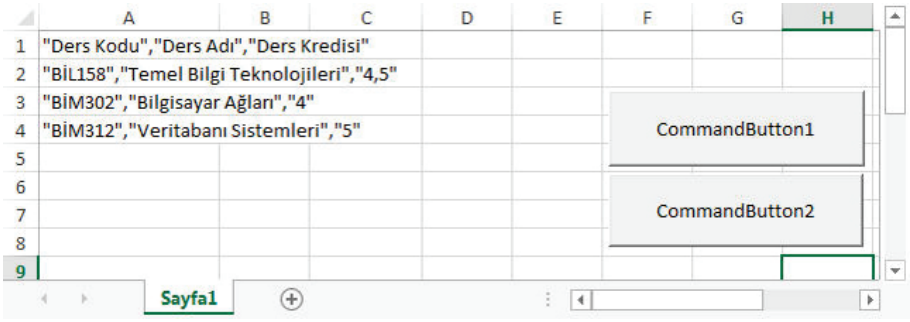
Resim 8.13



Line Input Komutu Kodu

Bu kod için dört ayrı String tip değişken tanımlanmış ve bu değişkenlere okunan her bir satır A1, A2, A3 ve A4 hücrelerine yazdırılmıştır. Write komutu ile dosyaya yazdırılan veriler, tıpkı Resim 8.10'da, yani kod çıktısında görüldüğü gibi satır hâlinde, hücrelere değer olarak atanmıştır.

Resim 8.14



MS Excel Sayfası Line Input Komutu Kod Çıktısı

Line Input komutu ile veri girişi yapmak için dosya açarken kullanılan Input komutu yerine, Output komutu kullanılsaydı sonuç ne olurdu?

SIRA SİZDE 3

EOF ve LOF İfadeleri

EOF (End of File – Dosya Sonu) ve LOF (Length of File – Dosya Uzunluğu) ifadeleri, dosyalarla çalışma esnasında sıkça başvurulan iki ifadedir. EOF ifadesi kullanıldığında, açık olan dosyanın sonuna gelinip gelinmediği test edilir. Bu sayede dosya içerisindeki kayıtlar okunurken, var olan tüm kayıtların okunup okunmadığı, dosyanın sonu test edilerek anlaşılabilir. EOF ifadesini kullanırken hangi dosya ile ilgili işlem yapılmak isteniyorsa o dosyanın tanımlayıcı tekil numarasından faydalanılır. LOF ifadesi ise yine dosya tanımlayıcı tekil numarasından faydalanılarak, o dosya içerisindeki kayıtların uzunluğu hakkında bilgi verir. LOF ifadesinin rastgele erişimli dosyalarda, kayıt sayısının hesaplanmasına yönelik kullanımı da mevcuttur.

Put ve Get İfadeleri

Hatırlanacak olursa ardışık erişimli dosyalarda dosyaya veri girişi yapmak için kullanılan komutlar, Print ve Write, veri okumak için kullanılan komutlar ise Input ve Line Input olarak açıklanmıştır. Rastgele erişimli dosyalar ve ikili dosyalarda ise bu işlemler Put ve Get komutları ile gerçekleştirilir.

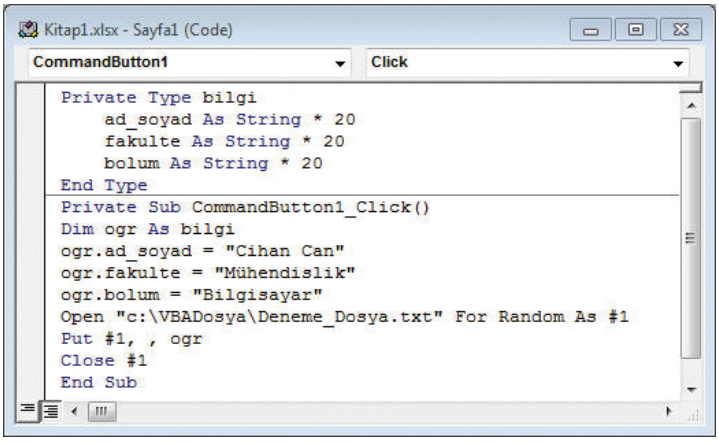
Put komutunu, gerekli durumlarda üzerinden veri girişi yapılacak olan dosyanın tanımlayıcı tekil numarasından sonra, Kayıt_Sayısı değeri ile birlikte kullanmak da mümkündür. Dosyaya kaydedilecek olan değerler ise daha sonra yazılarak komut tamamlanır.

Bu kod için kayıt uzunluğu (Len) ifadesi kullanılmak istendiğinde, bir kayıt için hesaplanacak uzunluk, bilgi değişken yapısının tümü için hesaplanacaktır.

Resim 8.15'te görüleceği üzere öğrencinin adı, soyadı, okuduğu fakültenin adı ve bölümün adı bir öğrenci dosyasında toplanmak istensin. Bunun için öncelikle bir değişken yapısı oluşturulmuş ve her bir değer için 20 karakterlik yer ayrılmıştır. Daha sonra değişken yapısı içerisinde tanımlanan üç değişken için değerler atanmış ve açılan rastgele erişimli Deneme_Dosya.txt dosyası üzerine bu bilgiler Put komutu kullanılarak yazdırılmıştır. Değerlerin dosya üzerine yazdırıldıktan sonra dosyanın durumu Resim 8.16'da görülmektedir.

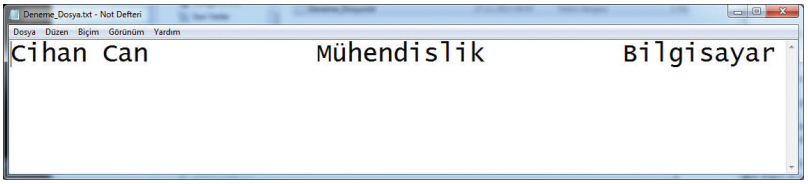
Resim 8.15

Put Komutu ile Değerleri Dosyaya Yazdıran Kod



Resim 8.16

Rastgele Erişimli Dosya Görüntüsü



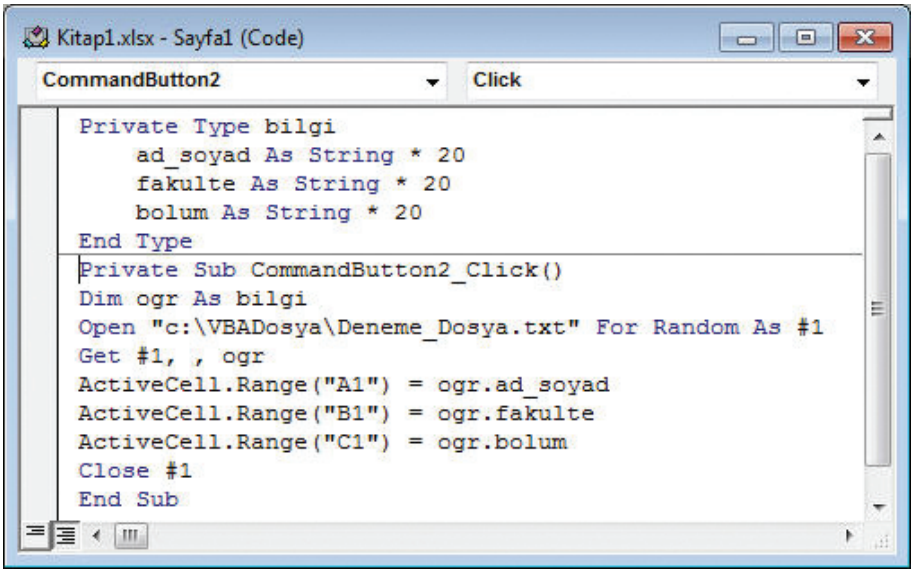
SIRA SİZDE 4

Resim 8.15'teki kodu inceleyiniz. Dosya açmak için kullanılan Open komutunda Output yerine Random kullanılmasının sebebi sizce nedir?

Görüldüğü üzere Put komutu, oluşturulan ya da daha önceden oluşturulmuş bir dosyanın üzerine veri yazmak için kullanılan bir komuttur. Put komutu ile üzerine veri yazılmış ya da daha önceden verilerle dolu bulunan bir dosyadan veri okuma işlemi ise Get komutu ile yapılabilir. Get komutu ile okunan veriler yine Input komutunda olduğu gibi önce değişkene ya da değişken yapısına, sonra da gerekli görülürse MS Excel hücrelerine yazdırılabilir. Get komutu da Put komutu gibi öncelikle hangi dosyadan veri okunacağını belirlemek amacıyla dosya tanımlayıcı tekil numarasına ihtiyaç duyar. Daha sonra verilerin hangi değişkene ya da değişken yapısına atanacağı belirlenir. Put komutu için kullanılan örnekten veri okuyarak MS Excel hücrelerine yazan kod parçacığı Resim 8.17'de görülmektedir.

Resim 8.17

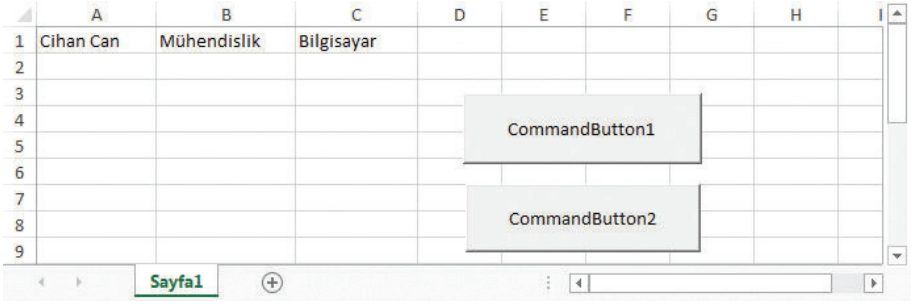
Get Komutu İçin Kod Örneği



Get komutu için ikinci bir buton eklenmiş ve kod bu butona tanımlanmıştır. Değişken yapısı olarak Put komutu için oluşturulan bilgi yapısından faydalanılmış ve yapı içerisindeki değişkenlere atanan değerler, MS Excel sayfasında A1, B1 ve C1 hücrelerine ayrı ayrı yazdırılmıştır. Kod çalıştırıldıktan sonra MS Excel sayfası çıktısı Resim 8.18'de görülmektedir.

Resim 8.18

Dosyadan Okunan Verilerin MS Excel Sayfasına Gönderilmiş Hâli



Put ve Get komutları aynı zamanda ikili dosyalar için de kullanılmaktadır. Put komutu dosya tanımlayıcı tekil numarasından sonra değişken değerlerini ikili dosya üzerine gönderir. Dikkat edilmesi gereken hususların başında, Get ile ikili dosya üzerinden veri okurken değişken tiplerinin, dosya yazımında kullanılan veri tipleriyle aynı olmasıdır.

VBA ortamında dosyalarla çalışırken sık karşılaşılan sorunlardan biri de üzerinden veri okunmak istenilen dosyanın veri yapısı hakkında bilgi sahibi olunmaması durumudur. Böyle durumlarda dosyayı ikili dosya olarak açarak verileri her defasında bir bayt şeklinde okumak bir çözüm olabilir. Daha önce bahsedilen LOF ifadesi de dosya içerisindeki kayıtların uzunluğunu bulmada programcıya kolaylık sağlayacaktır.

Seek ve Loc İfadeleri

Rastgele erişimli dosyalarla çalışırken etkin kayıt kavramı da önemlidir. MS Visual Basic yazılım ortamı, rastgele erişimli dosyaların etkin kayıtları için bir işaretçi tutmakta ve bu sayede kayıtlar belirlenebilmektedir. Tüm kayıtların aynı uzunlukta olduğu düşünüldüğünde, bir kayıt okunduktan sonra, sıradaki kaydın etkin olması kolaylaşmaktadır.

Seek ifadesi sayesinde üzerinde çalışılan dosyada istenilen kayda gidilebilmektedir. Loc ifadesi ise o anda etkin olan kaydın numarasını döndürerek hangi kayıtta olduğunun kullanıcı tarafından belirlenmesine yardımcı olmaktadır. Seek ifadesi kaydın yerini belirleyen bir işaretçi gibi düşünülürse Loc ifadesinin sadece bu işaretçinin değerini döndüren bir fonksiyon olduğu anlaşılmaktadır.

Lock İfadesi

Bazı kayıtlara, genellikle güvenlik nedeniyle kullanıcı tarafından erişilmemesi gerekli olabilmektedir. Lock ifadesi ile gerekli bir ya da daha fazla kaydı kilitlemek mümkün olmaktadır. Lock ifadesinden sonra dosyanın tanımlayıcı tekil numarası ve kilitlenmek istenilen kayıt uzunluğu girilmelidir. Kayıt uzunluğu girilmediyse tüm dosya kilitlenir. Lock ifadesi ile kilitlenen bir dosya ya da dosya içerisindeki kayıtların bir bölümü, Unlock ifadesi ile açılabilmekte ve tekrar kullanıcı kullanımına sunulmaktadır.

Özet

VBA dosya sistemlerini tanımlamak.

VBA ortamı, MS Visual Basic programlama dilinin dosya sistemi ve işlemlerini kullanan bir yapıya sahiptir. Bu sebeple MS Visual Basic programlama dilinde tanımlanan Ardışık Erişimli Dosya, Rastgele Erişimli Dosya ve İkili Dosya olarak tanımlanan üç farklı dosya tipini desteklemektedir. Bu dosyaların kullanım amaçları, MS Excel verilerinin bilgisayarların depolama birimlerinde yaratılan dosyalar üzerine depolanması ya da depolama birimlerinde kayıtlı bulunan bir dış kaynaklı dosya üzerindeki verilerin MS Excel sayfasına aktarılması olarak sıralanabilir.

VBA dosya işlemlerini açıklamak.

Dosyalarla işlem yapmak için öncelikle dosyanın yaratılması ve açılması ya da daha önce yaratılan dosyanın açılması gereklidir. Dosya yaratmak ya da açmak için Open komutundan faydalanılır. Open komutu dosyanın Ardışık Erişimli, Rastgele Erişimli ve İkili dosya olmasına göre farklı şartlarda açılabilir. Depolama birimleri üzerinde kayıtlı olmayan bir dosya açılmak istendiğinde, öncelikle tanımlanan yola göre dosya yaratılır ve sonra açılır. Açık olan dosyayı dosya işlemleri ile depolama biriminden silmek imkânsızdır. Bu işlem için dosya önce kapatılmalı ve sonra silme işlemi gerçekleştirilmelidir.

Dosya üzerine veri ekleme işlemini gerçekleştirmek.

Dosya üzerine veri ekleme işlemi ardışık erişimli dosyalar, rastgele erişimli ve ikili dosyalar için farklı ifadelerle sağlanmaktadır. Ardışık erişimli dosyalar için dosya uygun komut ile (Input) açıldıktan sonra Print ve Write komutları kullanılır. Rastgele erişimli dosyalar ve ikili dosyalar için ise yine dosya açıldıktan sonra Put komutu kullanılmaktadır. Put komutu kullanılmadan önce değerlerin, oluşturulan bir değişken yapısına atanması ve bu yapının dosyaya yazdırılması yolu izlenir.

Dosya kontrolü işlemini tanımlamak.

Dosya okunmak için açıldıysa, dosya üzerindeki kayıtların sonuna gelinip gelinmediği EOF (End of File) yani dosya sonu ifadesi ile kontrol edilir. Bu ifade dosya üzerindeki kayıtların sonuna gelindiğinde True (Doğru) değer döndüren bir ifadedir.

Dosya uzunluğunu belirlemek.

Dosya üzerindeki kayıtların uzunluğu, özellikle okuma işleminde gerekli olan bir özelliktir. LOF (Length of File) yani Dosya Uzunluğu ifadesi, dosya içerisinde bulunan kayıtların uzunluğunu bildiren bir ifadedir.

Dosyadan veri okuma işlemlerini uygulamak.

Veri okumak için dosyanın, Open komutundan sonra dosya tipine uygun ifade ile açılması gereklidir. Daha sonra ardışık erişimli dosyalar için Input ve Line Input komutları yardımıyla veriler önce değişkenlere, sonra da MS Excel sayfası üzerinde belirlenen hücrelere yazdırılabilir. Rastgele erişimli ve ikili dosyalarda ise bu işlemler Get komutu yardımıyla gerçekleştirilmektedir. Get komutu ile alınan değerler, yine gerekli durumlarda tek bir değişkene değil bir değişken yapısına atanabilir.

Kendimizi Sınayalım

1. Aşağıdakilerden hangisi dosya yönetimi işlemlerinden biri **değildir?**

a. Dosya Okunması
b. Dosya Oluşturulması
c. Dosya Kopyalanması
d. Dosya Silinmesi
e. Dosya Taşınması

2. Aşağıdakilerden hangisi dosya erişimi işlemlerinden biridir?

a. Dosya Taşınması
b. Dosya Silinmesi
c. Dosya Kopyalanması
d. Veri Yazılması
e. Dosya Oluşturulması

3. Resim dosyaları aşağıdaki dosya tiplerinden hangisinde depolanır?

a. Rastgele Erişimli Dosyalar
b. Sistem Dosyaları
c. İkili Dosyalar
d. Ardışık Erişimli Dosyalar
e. Direkt Erişimli Dosyalar

4. Dosya açmak için kullanılan ana komut aşağıdakilerden hangisidir?

a. For
b. Open
c. Len
d. As
e. #

5. Kodlama anında birden fazla dosyayı birbiri ile karıştırmadan kullanmak için gerekli olan öğe aşağıdakilerden hangisidir?

a. Dosya Yolu
b. Dosya Uzunluğu
c. Dosya Sonu
d. Dosya Oluşturma Şekli
e. Dosya Tanımlayıcı Tekil Numarası

6. Birden fazla dosya ile çalışıldığı durumlarda, tüm dosyalar tek bir komut ile kapatılmak istenirse aşağıdaki komutlardan hangisi kullanılmalıdır?

a. Open
b. Lock
c. Reset
d. Print
e. Seek

7. Dosyanın İkili Kipte açılmasını sağlayan ifade aşağıdakilerden hangisidir?

a. Input
b. Output
c. Append
d. Binary
e. Random

8. Ardışık erişimli dosyaya veri yazdırmak için aşağıdaki komutlardan hangisi kullanılır?

a. Write
b. Close
c. Open
d. Get
e. Binary

9. "Get" komutu ile yapılan işlem aşağıdakilerin hangisinde doğru verilmiştir?

a. Ardışık erişimli dosya açar.
b. Rastgele erişimli dosya kapatır.
c. İkili dosyaya veri yazar.
d. Dosya kayıt sayısını verir.
e. Rastgele erişimli dosyadan veri okur.

10. Dosya içerisindeki kayıt uzunluğunu hesaplamaya yarayan ifade aşağıdakilerden hangisidir?

a. EOF
b. LOF
c. LOCK
d. SEEK
e. PUT

Kendimizi Sınayalım Yanıt Anahtarı

1. a	Yanıtınız yanlış ise "Dosyalarla Çalışmak" başlıklı konusunu yeniden gözden geçiriniz.
2. d	Yanıtınız yanlış ise "Dosyalarla Çalışmak" başlıklı konusunu yeniden gözden geçiriniz.
3. c	Yanıtınız yanlış ise "Giriş" başlıklı konusunu yeniden gözden geçiriniz.
4. b	Yanıtınız yanlış ise "Dosyalarla Çalışmak" başlıklı konusunu yeniden gözden geçiriniz.
5. e	Yanıtınız yanlış ise "Dosyalarla Çalışmak" başlıklı konusunu yeniden gözden geçiriniz.
6. c	Yanıtınız yanlış ise "Dosyalarla Çalışmak" başlıklı konusunu yeniden gözden geçiriniz.
7. d	Yanıtınız yanlış ise "Dosyalarla Çalışmak" başlıklı konusunu yeniden gözden geçiriniz.
8. a	Yanıtınız yanlış ise "Dosya Erişimi" başlıklı konusunu yeniden gözden geçiriniz.
9. e	Yanıtınız Yanlış ise "Dosya Erişimi" başlıklı konusunu yeniden gözden geçiriniz.
10. b	Yanıtınız Yanlış ise "Dosya Erişimi" başlıklı konusunu yeniden gözden geçiriniz.

Sıra Sizde Yanıt Anahtarı

Sıra Sizde 1

MS Visual Basic yazılım ortamı tarafından desteklenen üç dosya çeşidi sırası ile Ardışık Erişimli Dosyalar, Rastgele Erişimli Dosyalar ve İkili Dosyalar olarak sıralanabilir.

Sıra Sizde 2

Dikkat edilecek olursa Deneme_Dosya.txt dosyası için dosya tanımlayıcı tekil numarası #1 olarak belirlenmiştir. Kapatma işleminde dosya tanımlayıcı tekil numarası da kullanılması gerektiği için bu bilgi önemlidir. Bu dosyayı kapatmak için gerekli kod:

Close #1

şeklinde olmalıdır.

Sıra Sizde 3

Dosya açarken Input yerine Output ifadesi kullanılsaydı kod hata verecek ve çalışmayacaktı. Bunun sebebi Line Input kodunun okuma yapmak için yani dosya üzerindeki değerleri belirlenen değişkenlere atamak için kullanılan bir kod olmasındandır. Input ile dosyayı açmak bu işleme izin verir. Output ise dosya üzerine değer yazmak için açılmasını sağlar.

Sıra Sizde 4

Hatırlanacak olursa Input ve Output ifadeleri ardışık erişimli dosyalar için kullanılan ifadelerdi. Random ifadesi ise rastgele erişimli dosyada kullanılmaktadır. Put ifadesi ile üzerine bilgi yazılan dosya rastgele erişimli dosya olduğu için ifade Random olarak kullanılmıştır.

Yararlanılan ve Başvurulabilecek Kaynaklar

MS Visual Basic 6.0 Temel Kullanım Kılavuzu, Alfa Yayınları.