T.C.
MİLLİ EĞİTİM BAKANLIĞI

MEGEP

(MESLEKİ EĞİTİM VE ÖĞRETİM SİSTEMİNİN
GÜÇLENDİRİLMESİ PROJESİ)

BİLİŞİM TEKNOLOJİLERİ

GÖRSEL PROGRAMLAMADA HATA GİDERME

ANKARA 2007

Milli Eğitim Bakanlığı tarafından geliştirilen modüller;

- Talim ve Terbiye Kurulu Başkanlığının 02.06.2006 tarih ve 269 sayılı Kararı ile onaylanan, Mesleki ve Teknik Eğitim Okul ve Kurumlarında kademeli olarak yaygınlaştırılan 42 alan ve 192 dala ait çerçeve öğretim programlarında amaçlanan mesleki yeterlikleri kazandırmaya yönelik geliştirilmiş öğretim materyalleridir (Ders Notlarıdır).
- Modüller, bireylere mesleki yeterlik kazandırmak ve bireysel öğrenmeye rehberlik etmek amacıyla öğrenme materyali olarak hazırlanmış, denenmek ve geliştirilmek üzere Mesleki ve Teknik Eğitim Okul ve Kurumlarında uygulanmaya başlanmıştır.
- Modüller teknolojik gelişmelere paralel olarak, amaçlanan yeterliği kazandırmak koşulu ile eğitim öğretim sırasında geliştirilebilir ve yapılması önerilen değişiklikler Bakanlıkta ilgili birime bildirilir.
- Örgün ve yaygın eğitim kurumları, işletmeler ve kendi kendine mesleki yeterlik kazanmak isteyen bireyler modüllere internet üzerinden ulaşılabilirler.
- Basılmış modüller, eğitim kurumlarında öğrencilere ücretsiz olarak dağıtılır.
- Modüller hiçbir şekilde ticari amaçla kullanılamaz ve ücret karşılığında satılamaz.

İÇİNDEKİLER

AÇIKLAMALAR

KOD	**482BK0071**
ALAN	**Bilişim Teknolojileri**
DAL/MESLEK	**Veritabanı Programcılığı**
MODÜLÜN ADI	**Görsel Programlamada Hata Giderme**
MODÜLÜN TANIMI	Görsel programlamasa hata giderme ile ilgili öğrenme materyalidir.
SÜRE	40/32
ÖN KOŞUL	Görsel Programlama Yardımcı Kodları modülünü bitirmiş olmak
YETERLİK	Görsel programlamada kod hata gidermek.
MODÜLÜN AMACI	**Genel Amaç** Gerekli ortam sağlandığında, program hatalarını giderebilecek ve değişken değerlerini takip edebileceksiniz. **Amaçlar** 1. Basit hata ayıklama işlemlerini yapabileceksiniz. 2. Program çalışırken değişkenlerin değerlerini takip edebileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	Bilgisayar laboratuvarı ve bu ortamda bulunan; bilgisayar, yazıcı, internet bağlantısı, bilgisayar masaları, kâğıt, kalem
ÖLÇME VE DEĞERLENDİRME	Her faaliyet sonrasında o faaliyetle ilgili değerlendirme soruları ile kendi kendinizi değerlendireceksiniz. Modül içinde ve sonunda verilen öğretici sorularla edindiğiniz bilgileri pekiştirecek, uygulama örneklerini ve testleri gerekli süre içinde tamamlayarak etkili öğrenmeyi gerçekleştireceksiniz. Sırasıyla araştırma yaparak, grup çalışmalarına katılarak ve en son aşamada alan öğretmenlerine danışarak ölçme ve değerlendirme uygulamalarını gerçekleştireceksiniz.

GİRİŞ

Sevgili Öğrenci,

Programcılıkta en önemli hususlardan biri hata vermeyen ve doğru çalışan bir program oluşturmaktır. Programınızın hata vermemesi onun doğru sonuç ürettiğini göstermez.

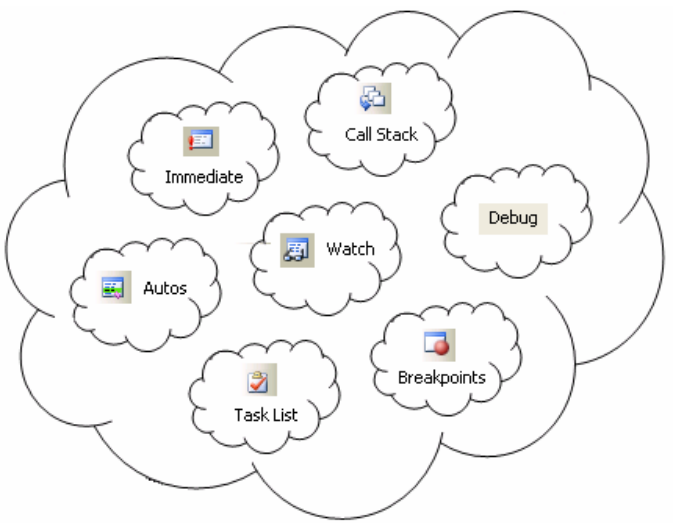
Yapılan programda oluşabilecek mantık hatalarını incelemek için genellikle programın adım adım çalıştırılması veya belirli bir bölgesinin çalıştırılıp incelenmesi gerekir. Bu modül yardımı ile bu işlemleri gerçekleştirebileceksiniz.

Genel olarak bu modülde öğreneceğiniz konular programınızdaki böcekleri ayıklama ve değişkenlerin değerlerini takip etme işlemleridir. Modülü bitirdiğinizde anlamadığınız yerleri tekrar okuyup uygulayınız. Konuları tam olarak kavramadan diğer modüle geçmeyiniz.

Konular kapsamlı olarak, derinlemesine anlatılmamıştır. Ne kadar çok uygulama, araştırma yapar iseniz kendinizi o kadar geliştirirsiniz ve ilerletirsiniz.

Görsel programlama dillerinden birini öğrenmek hepsini öğrenmek demek değildir. Fakat işlemlerin mantıkları aynıdır. Bu sebeple yapılan işlemlerin mantığını, yöntemini çalışma prensibini anlamaya çalışmalısınız. Bu sayede hangi programlama dilini kullanırsanız kullanın sadece komutları değiştirerek programlama yapabilirsiniz. Asıl istenen, konunun uygulanmasıdır, somut hale gelmesidir.

Bu modülde verilen programlarda bilerek veya bilmeyerek yapılmış hatalar olabilir. Bulduğunuz hataları ve yeni önerilerinizi arkadaşlarınız ve öğretmenleriniz ile paylaşınız. Belli yerlerde geçen araştırma konuları için "Önerilen Kaynaklar" kısmından yardım almayı unutmayınız.



ÖĞRENME FAALİYETİ–1

AMAÇ

Gerekli ortam sağlandığında, hata ayıklama işlemlerini yapabileceksiniz.

ARAŞTIRMA

Bu faaliyet öncesinde yapmanız gereken araştırmalar şunlardır:

1. Programlamada böcek kelimesinin anlamını araştırınız.
2. Program yazımında oluşabilecek hataları araştırınız.

Araştırma işlemleri için internet ortamı ve görsel programlama dilini anlatan kaynak kitaplardan faydalanınız.

1. HATA AYIKLAMA

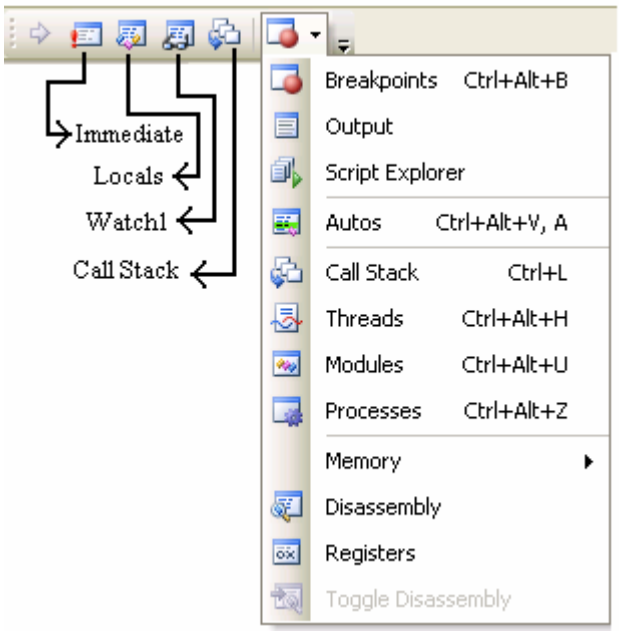
Böcek (bug) kelimesi, programcıların yaptıkları programlarda önceden tahmin edemedikleri "çalışma zamanında", yani program çalışırken ortaya çıkan hataları belirtmek için kullanılır. Başka bir deyişle "böcek" kelimesi, yazılımların çalışırken, çoğunlukla beklemedikleri veya kontrol etmedikleri bir girdi karşısında ne yapacaklarını bilememeleri veya kendilerine işletim sistemi tarafından yasaklanmış olan bazı şeyler yapmaya kalkmaları yüzünden "göçmesi" ile sonuçlanan "yazılım hataları" için kullanılır.

Bilgisayarların henüz bir oda büyüklüğünde olduğu zamanlarda, bir böceğin bilgisayarın içine girerek kısa devreye sebep olması, böylelikle o anda çalışmakta olan "yazılım"ın görevini yerine getirememesine yol açması sebebiyle çalışma zamanındaki yazılım hatalarına böcek denir. Çalışma zamanında oluşan bu böcekleri ayıklama işlemine de debug (böcek ayıklama) denir. Bu etkinlikte debug menüsü ile yapılan işlemler anlatılacaktır.

Debug menüsündeki komutlara kısayoldan ulaşmak için debug araç çubuğu kullanılabilir. Bu araç çubuğunu açmak için view menüsünden toolbars seçeneğinden debug araç çubuğu aktif edilmelidir.

Menü veya araç çubuklarının bulunduğu bölüme Mouse'un sağ tuşu ile tıklanıp gelen araç çubuklarından debug seçilerek de araç çubuğu Visual Basic.Net programına eklenmiş olur.

Araç çubuğu üzerindeki seçeneklerin birçoğu modül içerisinde bulunan öğrenme faaliyeti -2 de anlatılacaktır.



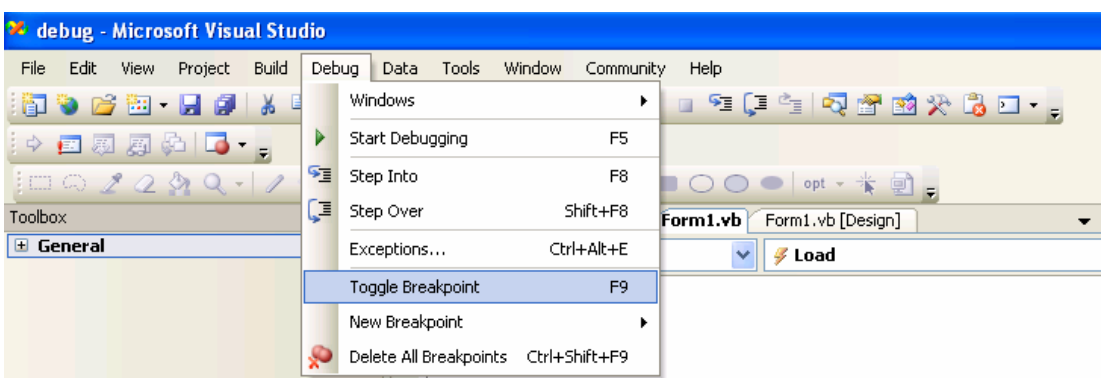
Resim 1: Debug araç çubuğu

1.1. "Breakpoint" Yapma

Yazılan bir programın doğru çalışıp çalışmadığını test etmek amacıyla parça parça incelenmesi gerekebilir. Bu işlem için breakpoint (kırılma noktası) oluşturulur. Böylece uzun bir programı parçalara bölerek hataya sebep olan komut satırları rahat bir şekilde bulunabilir.

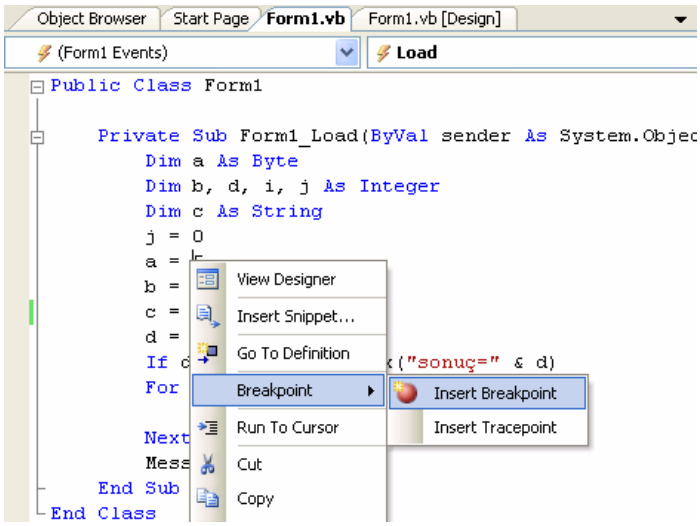
Visual Basic.Net programlama dilinde Breakpoint oluşturmak için iki temel yöntem kullanılabilir. Bunlardan birincisi programın herhangi bir satıra gelindiğinde kırılmasıdır. Bu yöntem 3 değişik şekilde yapılabilir. Bunlar;

- Breakpoint oluşturacağımız satıra Mouse yardımı ile gelip resim 1.1'deki gibi Debug menüsünden Toggle Breakpoint komutu seçerek veya F9 tuşuna basarak oluşturabiliriz.



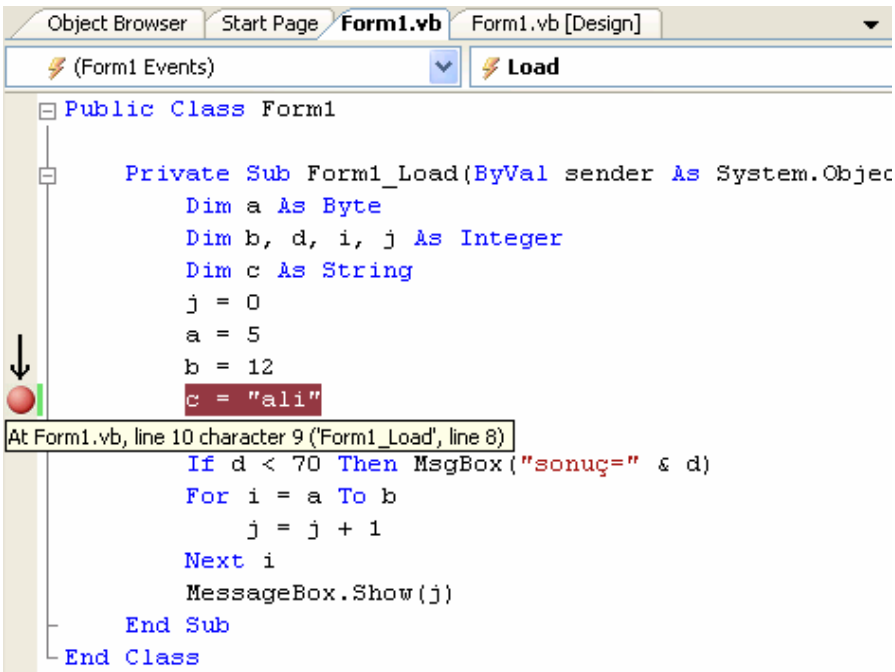
Resim 1.1: Breakpoint oluşturma 1

- Breakpoint oluşturacağımız satıra gelip resim 1.2'de görüldüğü gibi Mouse'un sağ tuşuna basarak karşımıza gelen pencereden Insert Breakpoint komutu seçerek oluşturabiliriz.



Resim 1.2: Breakpoint oluşturma 2

- Resim 1.3'te görüldüğü gibi Margin Indicator çubuğunda Breakpoint oluşturacağımız satıra gelip Mouse'un sol tuşuna tıklayarak oluşturabiliriz.

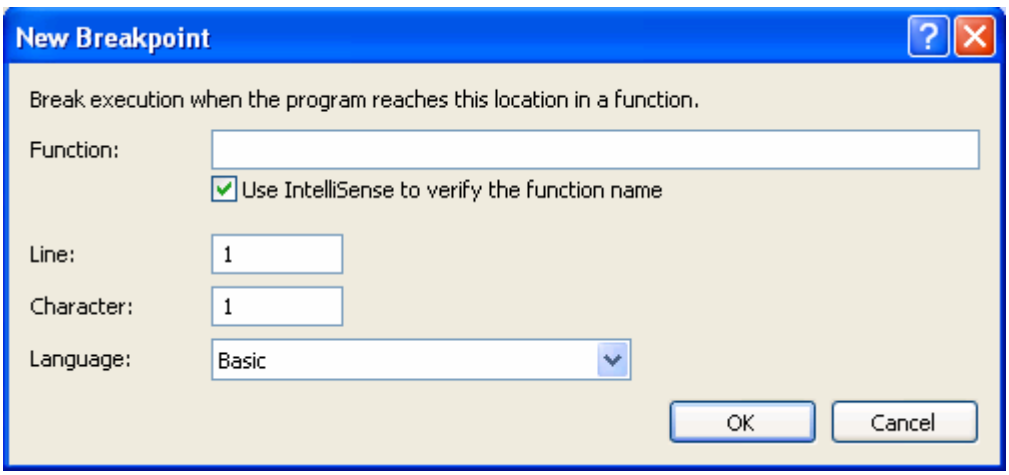


Resim 1.3: Breakpoint oluşturma 3

Uzun programlarda birden fazla breakpoint noktası oluşturarak programımızı daha kolay bir şekilde inceleyebiliriz. Birden fazla breakpoint oluşturulduğunda programımız 1. breakpoint e kadar çalıştıktan sonra durur. Eğer bu bölümde hata yoksa Start debugging ( ) butonuna veya F5 tuşuna basarak 2. breakpoint noktasına kadar program tekrar çalıştırılır. Bu şekilde tüm hatalar bulunup temizleninceye kadar program parça parça incelenebilir.

Breakpoint oluşturmak için kullanılan ikinci yöntem ise herhangi bir işlemin gerçekleşmesine bağlı olarak programın kırılmasını sağlayan New Breakpoint yöntemidir. Mesela bir fonksiyon içerisinde a değişkeninin değeri 15'ten farklı ise programın kırılmasını sağlamak buna örnek olarak gösterilebilir. Bu işlem için Debug menüsünden New Breakpoint komutu veya Ctrl+B kısayol tuşu kullanılır.

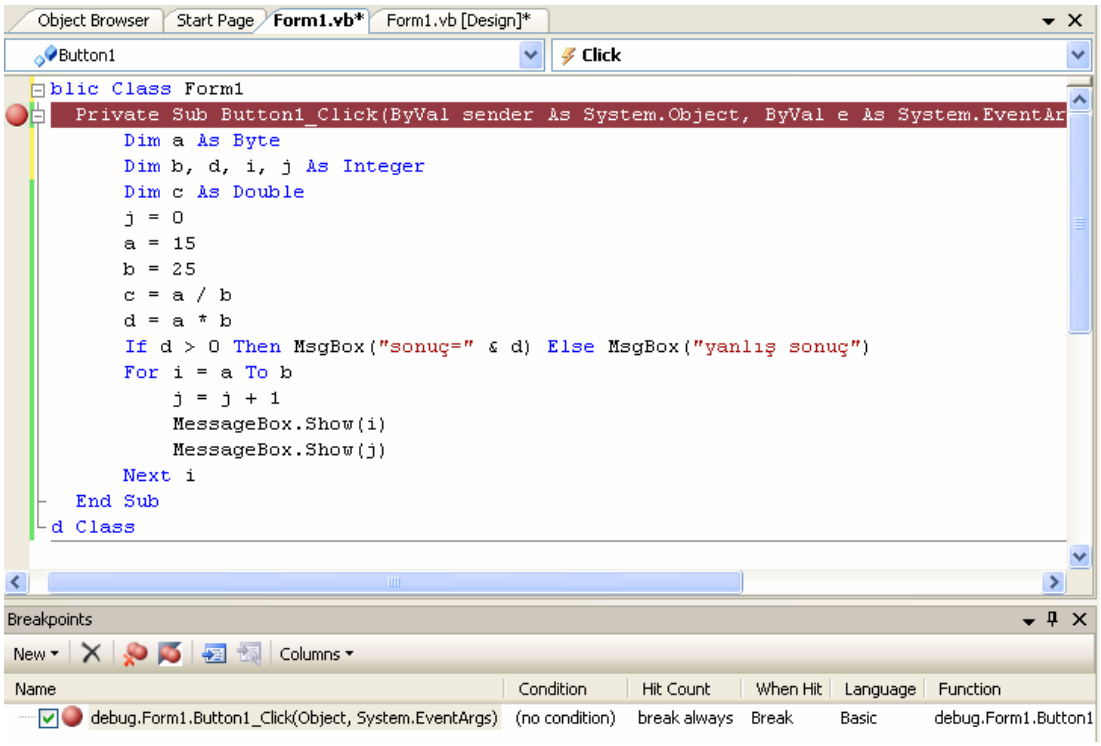
New Breakpoint komutu çalıştırıldığında karşımıza Resim 1.4'teki pencere gelir. Bu pencerede kırılmanın gerçekleşeceği fonksiyonun ismi, satır ve karakter numarası ile hangi programlama dilinde yazıldığı bilgileri bulunmaktadır.



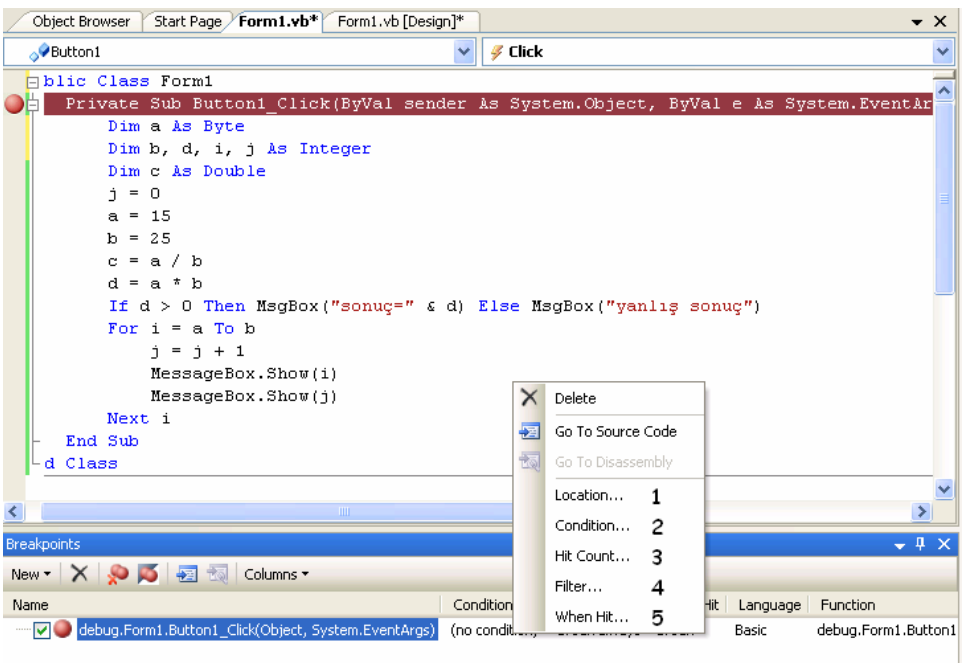
Resim 1.4: New Breakpoint penceresi

Burada Function ismi olarak Button1_click yazılır ve ok tuşuna basılırsa kırılma noktası programımıza eklenmiş olur. Bu kırılma noktası resim 1.5'te görüldüğü gibi breakpoint penceresine eklenir.

Tüm oluşturulan kırılma noktaları Breakpoints penceresinde görülür. Eğer kırılma noktası üzerinde herhangi bir işlem yapmazsak formdaki Buton1 tıklandığında programımız durdurulacaktır. Eğer fonksiyon içerisindeki herhangi bir değişkenin değerine göre programın kırılmasını istiyorsak condition özelliğini ayarlamamız gerekecektir. Bunun için breakpoint penceresindeki kırılma noktası üzerine gelinip Mouse'un sağ tuşu tıklandıktan sonra açılan pencereden condition seçeneği seçilir. Condition penceresine istenilen şart ifadesi yazılarak kırılma noktası ayarlanmış olur.



Resim 1.5: Kırılma noktası ve Kırılma noktası penceresi



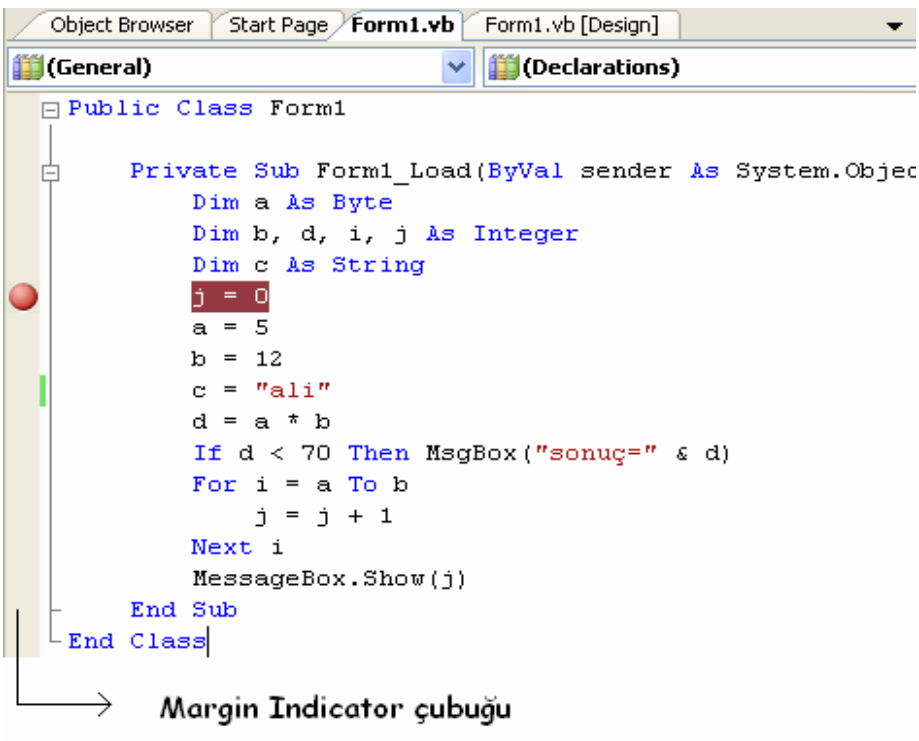
Resim 1.6: Kırılma noktası özellikleri

Kırılma noktasının özellikleri resim 1.6'da numaralandırılmıştır. Bunların görevleri sırasıyla;

1. Location: Kırılma noktasının bulunduğu yeri gösterir.
2. Condition: Kırılma işleminin herhangi bir koşula bağlı olarak gerçekleşmesini sağlar.
3. Hit Count: Kırılma işleminin ne zaman yapılacağını belirler. Örnek olarak istediğimiz şart sağlandığı zaman veya şart 3. defa gerçekleştiğinde.
4. Fitler: Kırılma işleminin hangi işlem veya iş parçalarında olabileceğini belirler.
5. When Hit: Kırılma işlemi gerçekleştiğinde ekrana mesaj aktarmayı veya herhangi bir makroyu çalıştırmayı sağlar.

Not: Resim 1.6'da condition işlemi seçilip şart olarak b<>25 yazıldığında, program çalıştırılıp button1 tıklanırsa b değişkeninin değeri 25 olduğunda şart sağlanmadığından kırılma oluşacaktır.

Oluşturulan breakpointler hatalar giderildikten sonra kaldırılmalıdır. Çünkü programın kaydedilmesiyle bu breakpointlerde kaydedilmiş olur. Breakpointler Visual Basic.Net programı kapatılsa dahi silinmez. Kullanıcının bu noktaları kaldırdıktan sonra programı kaydetmesi gerekir. Breakpoint kaldırma işlemi oldukça basittir. Resim 1.7'de görüldüğü gibi Magrin Indicator çubuğundan kaldırmak istediğimiz breakpoint noktasına gelip Mouse'un sol tuşuna basarak breakpointi kaldırmış oluruz.



Resim 1.7: Breakpoint kaldırma işlemi

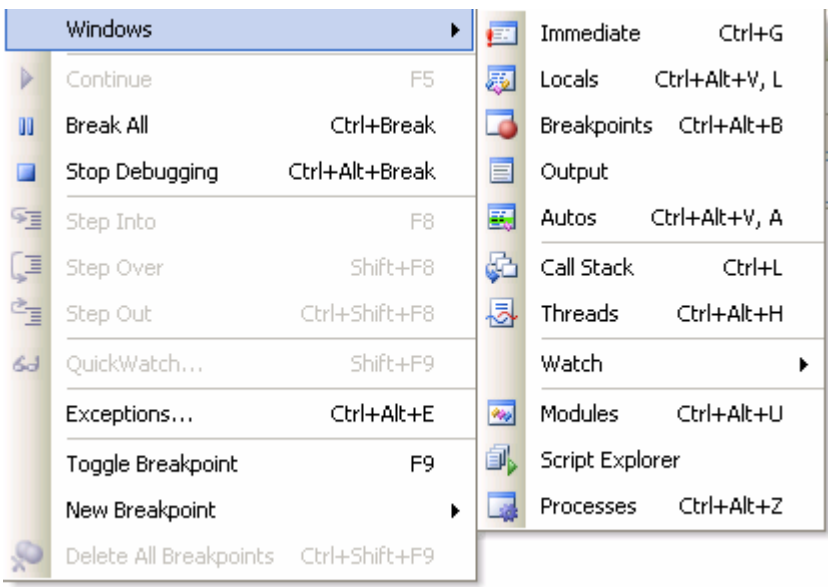
Not: Oluşturduğumuz breakpointleri otomatik olarak açılan breakpoint penceresinde görüp inceleyebiliriz. Ayrıca bu pencere vasıtasıyla yeni breakpoint ekleyebilir, breakpointleri silebilir veya pasif hale getirebiliriz.

1.2. "Break" Durumu Yapma

Bilindiği gibi bulunması en zor hata türü mantık hatasıdır. Diğer hata türlerinde derleyici programın yazımı sırasında veya hatalı satıra gelindiğinde kullanıcıyı uyarmaktadır. Bir mantık hatasını belirlemenin yollarından biri program kodunuzu satır satır çalıştırıp değişken veya özellikler değişirken bunların içeriğini incelemektir. Bu işlemi yapmak için programımız çalışırken break (kırılma, kesme) moduna girilebilir. Kesme modu, Visual Basic.NET derleyicisi programınızı çalıştırırken durdurup onu incelemenizi sağlar.

Visual Basic.NET programlama dilinde break yapmak için start debugging butonunun ( ) yanında bulunan break all butonuna ( ) Mouse'un sol tuşu yardımı ile tıklamak veya Ctrl+Break kısayol tuşunu kullanmak yeterlidir. Bu işlem yapılınca programımızın code penceresi ekrana gelir ve programımızı inceleme imkânı buluruz.

Break modunda ayrıca resim 1.8'de görüldüğü gibi debug menüsünde bulunan windows sekmesi kullanılarak programı incelemek için gerekli olan pencereler açılabilir, değişkenler incelenebilir.



Resim 1.8: Break modunda açılabilen pencereler

Bu pencereler kullanılarak oluşturulan programın ayrıntılı olarak incelemesi yapılabilir.

1.3. Klavye Kısayolları

F5: Oluşturulan programın çalıştırılması veya duraklatılmış (break modda) bir programın çalışmasına devam ettirilmesi için kullanılan kısayol tuşudur.

F9: Breakpoint eklemek için kullanılan kısayol tuşudur. Bu işlem için öncelikle hangi satıra breakpoint eklenecek ise Mouse o satıra konumlanmalıdır.

F10: Programımızı adım adım (Step Over) çalıştırmak için kullanılır. Program akışında procedureleri tek satır olarak kabul eder ve içine girmez.

F11: Programımızı adım adım (Step Into) çalıştırmak için kullanılır. Bu kısayol yardımı ile procedurelerin içine girilip incelenebilir. Bu işlem F8 tuşu yardımı ile de yapılabilir.

Ctrl+Break: Çalışmakta olan (running) programdaki bütün işlemleri geçici olarak durdurur ve hata ayıklama moduna (debugging) geçilmesini sağlar. Bu sayede programdaki hatalar düzeltilip programın kaldığı yerden çalıştırılması sağlanır.


Not: Eğer oluşturduğumuz programda procedure yoksa F10 ile F11 kısayol tuşlarının görevleri aynıdır. Aralarındaki fark programımızda procedure varsa ortaya çıkar. F11 tuşu yardımı ile procedure içine girip adım adım inceleme imkânı buluruz. F10 tuşu ise procedurelerin içine girmeden programı incelememizi sağlar.

1.4. Hatalı Satırın Altında Oluşan Mavi Dalgalı Çizgi Kullanımı

Visual Basic.Net programlama dili diğer programlama dillerinde olduğu gibi programın yazımı sırasında oluşabilecek hataları denetler. Programın yazımı sırasında oluşacak hatalara söz dizimi hatası ya da derleyici hatası denir. Bu hata bir komutun, bir özelliğin yanlış yazılmasıyla veya bir değişkenin tanımlanmaması sonucu oluşabilir. Kullanıcı oluşan söz dizimi hatalarını düzeltmediği zaman derleyici programın çalışmasına izin vermez. Visual Basic.NET programlama dilinde, oluşacak söz dizimi hatasının altı mavi dalgalı çizgi ile çizilerek kullanıcıya oluşan söz dizimi hatası gösterilir. Kullanıcı mouse yardımı ile söz dizimi hatasının olduğu yere geldiğinde hatanın oluş sebebi hakkında ayrıntılı bilgi edinebilir. Bu sayede hatalı bilgiler düzeltilip program çalıştırılmaya hazır hale getirilebilir.

Örnek 1: Aşağıdaki programa dikkat edilirse a karakterlerinin altının çizili olduğu görülür. Bu bir söz dizimi hatasıdır. Programcı programı çalıştırmadan bu hatayı düzeltmek zorundadır. Hatayı düzeltmek için öncelikle hatanın oluşum nedenini bulmamız gerekir. Mouse yardımı ile mavi dalgalı çizgi üzerine gelindiğinde resim 1.9'da görüldüğü gibi hatanın sebebi konusunda ekrana bir uyarı mesajı gelecektir.

```
Dim b, d, i, j As Integer
Dim c As String
j = 0
a = 5
b = 12
c = "ali"
d = a * b
If d Name 'a' is not declared. ox("sonuç=" & d)
For i = a To b
    j = j + 1
Next i
MessageBox.Show(j)
```

Resim 1.9: Söz dizimi hatasının oluşumu

Bu uyarı mesajında görüldüğü gibi hatanın sebebi a değişkeninin tanımlanmamasıdır. Bu hatayı gidermek için en üst satıra Dim a As Byte komut satırını yazmak yeterlidir. Daha sonra programdaki mavi dalgalı çizgilerin kalktığı görülecektir (Resim 1.10)

```
Dim a As Byte
Dim b, d, i, j As Integer
Dim c As String
j = 0
a = 5
b = 12
c = "ali"
d = a * b
If d < 70 Then MsgBox("sonuç=" & d)
For i = a To b
    j = j + 1
Next i
MessageBox.Show(j)
```

Resim 1.10: Söz dizimi hatasının giderilmesi

Örnek 2: Aşağıdaki programda (Resim 1.11) çeşitli söz dizimi hataları görülmektedir.

```
Dim a As Byte
Dim b, d, i, j As Integer
Dim c As String
j = 0
a = 5
b = 12
c = TextBox1.Txet          (1)
d = a * b
If d < 70 Then MsgBox("sonuç=" & d)
For i = a To b
    j = j + 1              (2)
    i()
    MessageBox.Show(j)
```

Resim 1.11: Çeşitli söz dizimi hataları

Örnekte görülen söz dizimi hataları komut ve özelliklerin yanlış yazılması sonucu oluşmuştur.

- 1 nu.lı sözdizimi hatası Textbox1 nesnesinin text özelliğinin yanlış yazılması sonucu oluşmuştur. Text yerine Txet yazılmıştır.
- 2 nu.lı söz dizimi hatası for komutunun bir parçası olan next ifadesinin yazılmaması sonucu oluşmuştur.

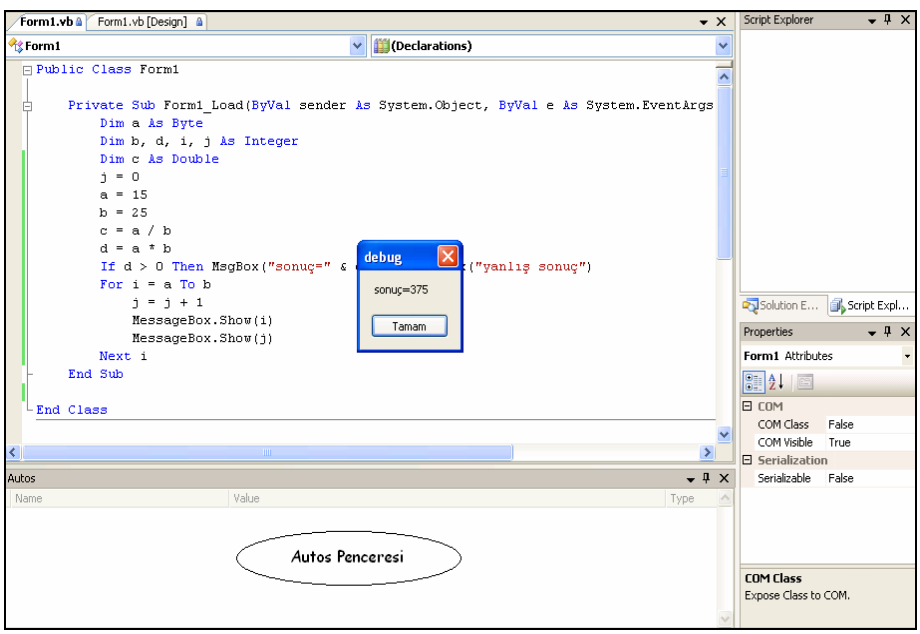
Mouse yardımı ile bu söz dizimi hatalarının üzerine gelindiğinde hataların oluşma sebebi öğrenilebilir.

1.5. "Task List" Penceresi

Task List (Görev listesi) penceresi programcının oluşturduğu görev ve açıklamaların listelenmesini ve kontrolünü sağlar. Programcı kendine göre bir görev listesi belirler ve bu sıraya göre programın yapımını kontrol eder. Ayrıca programın belli noktalarına açıklama satırları ekleyebilir. Bu pencereye View menüsünden Other Windows seçeneği işaretlenerek veya Ctrl+Alt+K kısayol tuşu vasıtasıyla ulaşılabilir.

1.6. "Autos" Penceresi

Autos penceresi program içerisindeki değişkenlerin değerlerini takip etmek için kullanılan bir penceredir. Bu pencere yardımı ile değişkenleri sadece takip etmeyip onların değerlerini program çalışırken değiştirip, programın bu değişime verdiği tepkiyi de görebiliriz. Autos penceresi otomatik olarak ekrana gelen bir penceredir. Bu pencereyi incelemek için programımızı debugging modda çalıştırmamız gerekir.



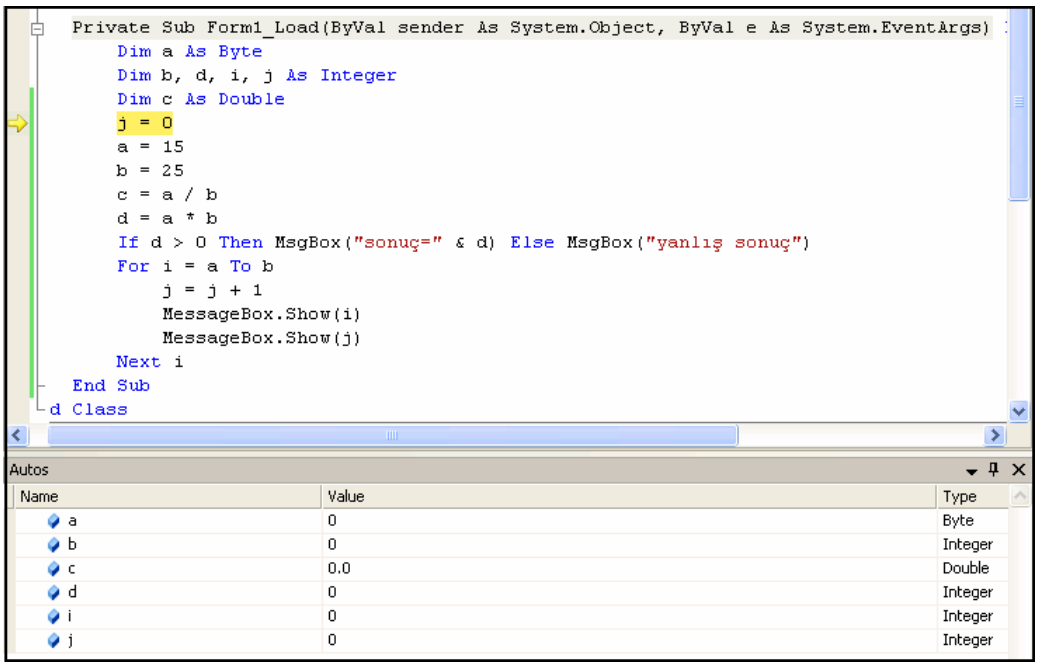
Resim 1.12: Autos penceresi

Programımızda tanımlı tüm değişken değerlerinin değişimini Autos penceresi yardımı ile takip edilebiliriz.

Örnek: Aşağıdaki programımızı Autos penceresini kullanarak inceleyelim.

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
    Dim a As Byte
    Dim b, d, i, j As Integer
    Dim c As Double
    j = 0
    a = 15
    b = 25
    c = a / b
    d = a * b
    If d > 0 Then MsgBox("sonuç=" & d) Else MsgBox("yanlış sonuç")
    For i = a To b
      j = j + 1
      MessageBox.Show(i)
      MessageBox.Show(j)
    Next i
     End Sub
```

- Programımızı F11 tuşu veya Step Into ( ) butonu yardımı ile çalıştırdıktan sonra ilk olarak Autos penceresi resim 1.13'teki hali alır.

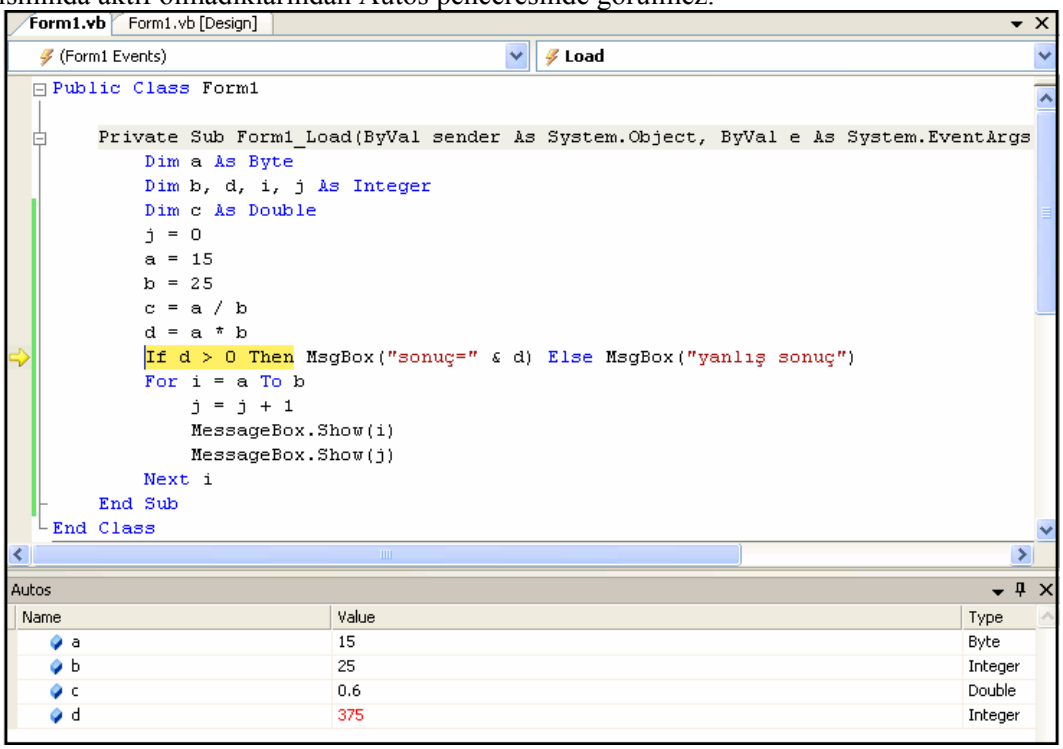


Resim 1.13: Autos penceresi ile değişken izleme

Autos penceresinde programımızda kullanılan tüm değişkenler ve bunların başlangıç değerleri ile değişken türleri görülebilmektedir.

- Programımızı adım adım çalıştırmaya devam edersek Autos penceresindeki değişimleri rahatlıkla takip edebiliriz.

Resim 1.14'te görüldüğü gibi Autos penceresi aktif olan değişkenleri görüntüler, o an aktif olmayan değişkenler pencerede görülmez. Burada i ve j değişkenleri programın bu kısmında aktif olmadıklarından Autos penceresinde görülmez.



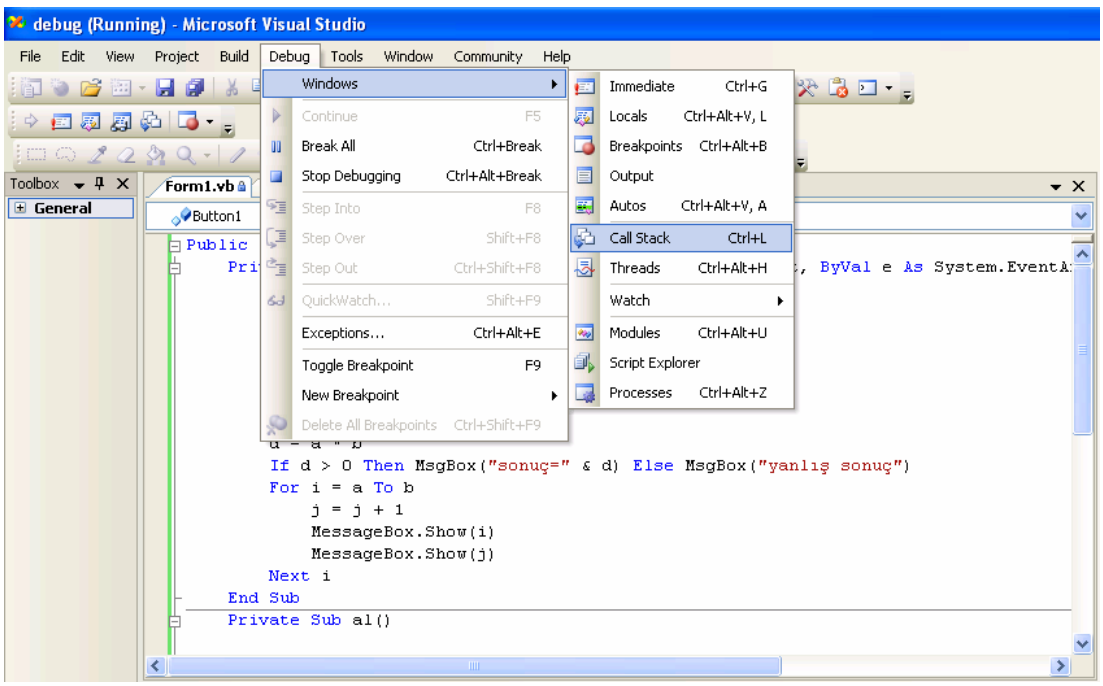
Resim 1.14: Autos penceresi ile değişkenlerin incelenmesi

- Programımızı adım adım çalıştırmaya devam edersek program değişken değerlerine göre işlemleri yapıp sona erecektir. Eğer programa herhangi bir yerde müdahale etmek istersek (değişken değerlerini değiştirmek gibi) istediğimiz değişkenin ismi üzerine gelip Mouse yardımı ile çift tıklayarak değerini değiştirebiliriz. Örnek olarak resim 1.14'te görüldüğü dibi d değişkeninin değeri 375'tir. Bu değere göre if komutu şarta bağlı olarak then ifadesinden sonra gelen komutu çalıştırıp bir sonraki komuta (for komutu) geçilmesini sağlar. Burada program akışını değiştirmek istersek, Autos penceresinde bulunan d değişkeninin değeri üzerine Mouse ile çift tıklayıp -5 yazarsak programa direkt olarak müdahale etmiş oluruz. Böylece if komutundaki şart sağlanmadığından else ifadesindeki komut satırı çalıştırılacaktır. Bu şekilde programımızın istenilen işi yapıp yapmadığını rahat bir şekilde gözlemlemiş oluruz.

Not: Program kesme modunda çalışırken değişken değerleri Mouse yardımı ile de görülebilir. Mouse değişken isminin üzerine getirildiğinde değişkenin değeri ekranda görülür.

1.7. “Call Stack” Penceresi

Call stack penceresi bellekte bulunan güncel fonksiyon ve pocedureleri görüntülemek ve bir fonksiyonun hangi fonksiyonlar tarafından çağrıldığını göstermek için kullanılır. Bu pencereyi açmak için programımızın debugging veya running modda çalışması gerekmektedir. Programımız bu modlardan herhangi birinde iken Debug menüsünden Windows sekmesi tıklanarak, Ctrl+L kısayol tuşu ile veya debug araç çubuğundan call stack penceresi açılabilir.



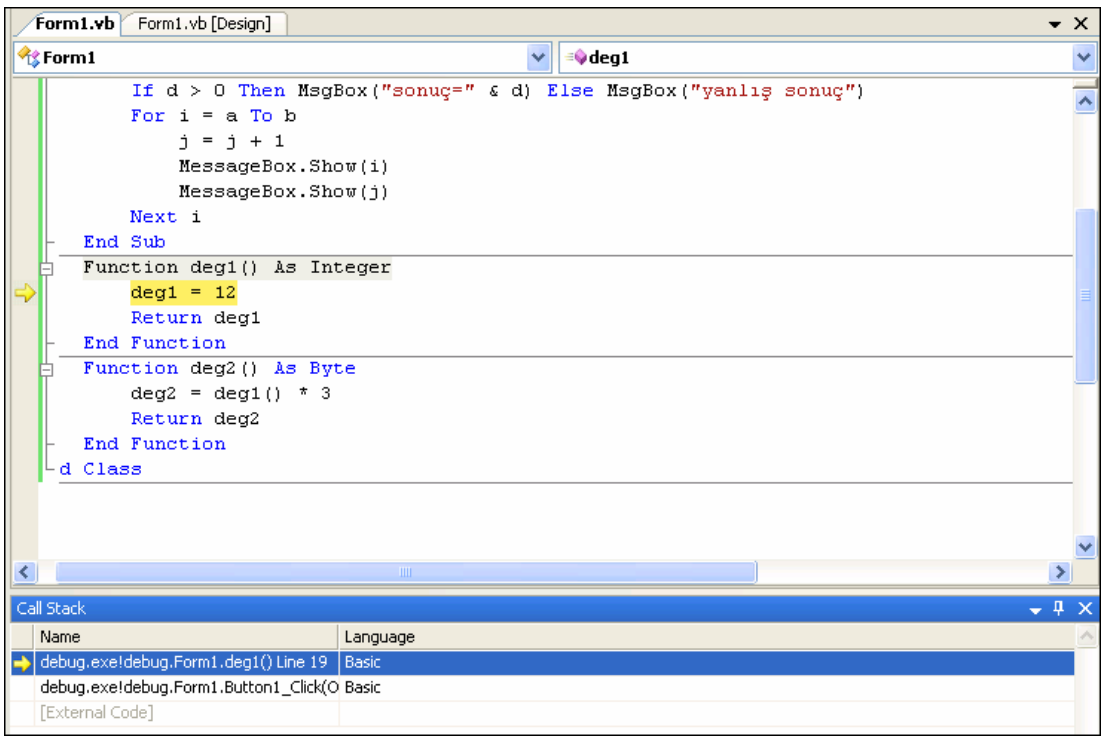
Resim 1.15: Call Stack penceresinin açılması

Örnek: Aşağıdaki program incelenirse 3 adet function bulunmaktadır. Bunlar deg1 ve deg2 ve Button1_Click functionlarıdır. Deg1 ve Deg2 functionlarından değerler alınıp Button1_Click functionunda kullanılmaktadır.

```
Dim a As Byte
   Dim b, d, i, j As Integer
   Dim c As Double
   Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
      j = 0
      a = 5
      b = deg1()
      c = deg2()
      d = b * c
      If d > 0 Then MsgBox("sonuç=" & d) Else MsgBox("yanlış sonuç")
      For i = a To b
         j = j + 1
```

```
            MessageBox.Show(i)
            MessageBox.Show(j)
        Next i
    End Sub
    Function deg1() As Integer
        deg1 = a * 2
        Return deg1
    End Function
    Function deg2() As Byte
        deg2 = deg1() * 3
        Return deg2
            End Function
```

Programı debugging modunda adım adım çalıştırırsak o an kullanılan functionları Call Stack penceresinde görebiliriz.



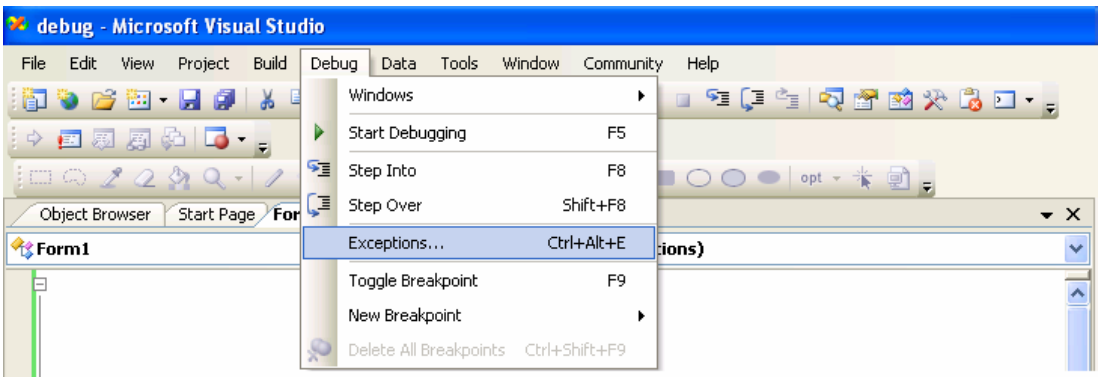
Resim 1.16: Call Stack penceresi

Call Stack penceresinde seçili satır, programın debug modunda çalıştığını, deg1 functionunda ve programın 19. satırında bulunduğumuzu gösterir. Deg2 fonksiyonu aktif olmadığından gösterilmez.

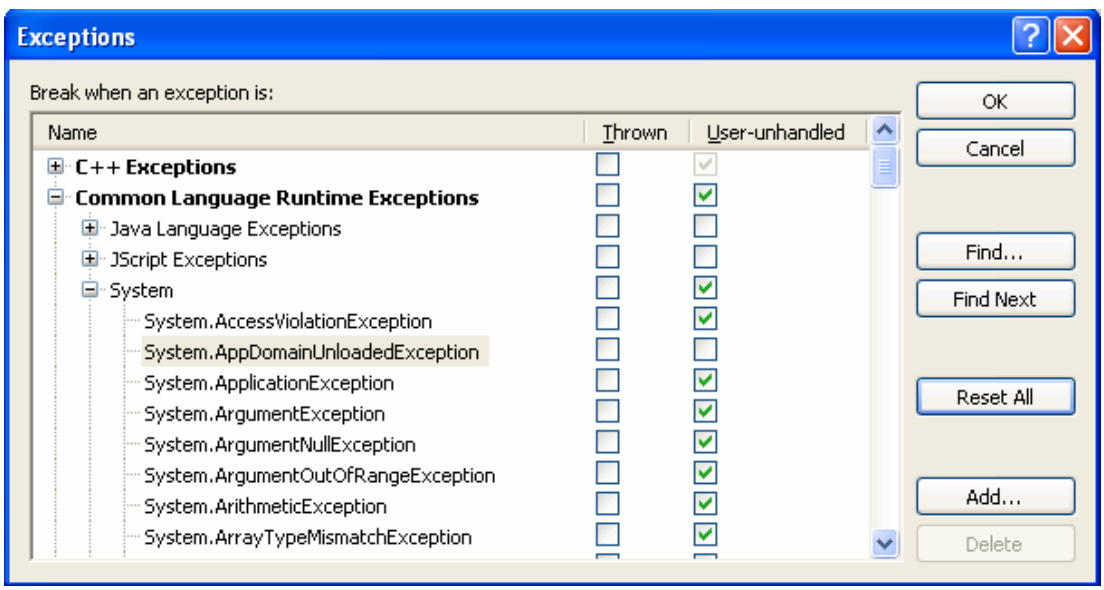
Araştırma: Resim 1.16 daki Call stack penceresinde Button1_Click functionunun gösterilme nedenini araştırınız.

1.8. “Exceptions” Diyalog Kutusu

Exceptions iletişim kutusu.NET programlama dilinde bulunan tüm istisnai durumları görmemizi sağlamak için kullanılır.



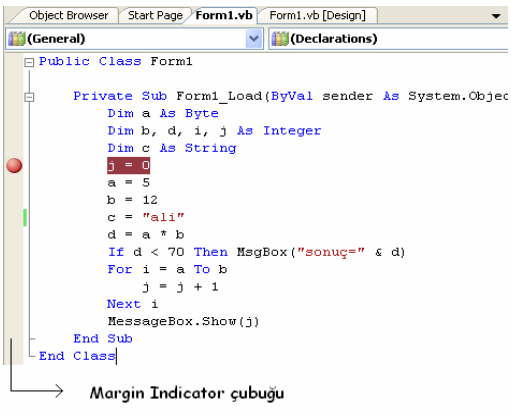
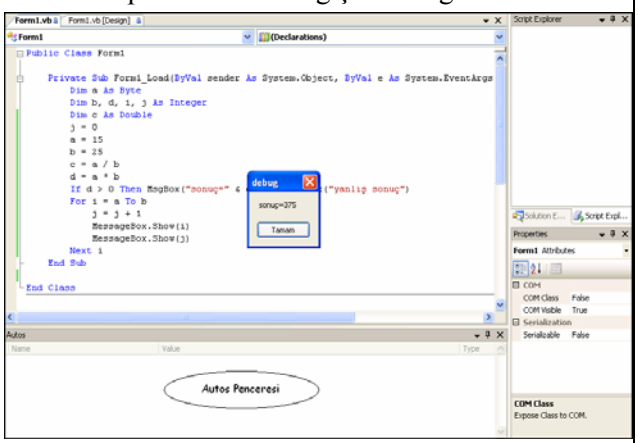
Resim 1.17: Exceptions iletişim kutusunun açılması



Resim 1.18: Exceptions penceresi

Bu pencere yardımı ile istisnai durumlar aratılabilir (find), yeni istisnai durum oluşturulabilir (Add) ve yapılan tüm değişiklikler silinip başlangıç durumuna dönülebilir (Reset All). Bir veya birden fazla istisnai durum için Thrown seçeneği seçilirse o istisnai durumların Try…Catch bloğu ile yönetilmesi engellenmiş olur. Ayrıca bu pencere yardımı ile istisnai durumların hangi programlara veya alan adlarına ait oldukları da bulunabilir. Mesela yukarıdaki pencerede ArgumentException istisnai durumunun sistem adalanının bir üyesi olduğu görülmektedir.

UYGULAMA FAALİYETİ

İşlem Basamakları	Öneriler
1. Marjin çubuğuna tıklayıp durak noktası oluşturunuz. 	Oluşturulan programın çeşitli noktalarına durak noktası ekleyerek programınızı çalıştırınız.
2. Programı çalıştırarak Mouse ile değişkenler üzerinden bilgi alınız.	Programı kesme modunda çalıştırıp değişkenlerin değer değiştirmesini bekleyiniz. Daha sonra değişken değerlerini Mouse ile inceleyiniz.
3. "Autos" penceresinde değişkenleri görünüz. 	Oluşturduğunuz programı kesme modunda çalıştırıp Autos penceresini açınız.
4. Programı adım adım çalıştırınız.	Debugging modunu kullanınız.
5. "Debug –Exceptions" ile istisnai durumları ayarlayınız.	Exceptions ayarlarını değiştirerek oluşturulan istisnai durumu, Try…Catch bloğu ile yakalamaya çalışınız.

ÖLÇME VE DEĞERLENDİRME

A- OBJEKTİF TESTLER (ÖLÇME SORULARI)

Aşağıdaki sorulardan; ilk 6 soruda verilen ifadeye göre parantez içine doğru ise "D", yanlış ise "Y" yazınız. Diğer sorular için uygun şıkkı işaretleyiniz.

1. Debug kelimesini Türkçe karşılığı böcektir. ( )

2. Bilgisayar yazılım hatalarına Debug denir. ( )

3. Programı durdurmak için break komutu kullanılır. ( )

4. Satırların altında bulunan mavi dalgalı çizgi o satırın function olduğunu gösterir. ( )

5. Task List penceresi değişkenleri görmek için kullanılır.( )

6. Autos penceresi ile programdaki bütün değişkenler aynı anda görülebilir. ( )

7. Aşağıdaki pencerelerden hangisi programda bulunan procedureleri gösterir?
A) Autos penceresi
B) Call Stack penceresi
C) Exceptions penceresi
D) Task List penceresi

8. Aşağıdaki pencerelerden hangisi tüm istisnai durumları görmemizi sağlar?
A) Task List penceresi
B) Exceptions penceresi
C) Call Stack penceresi
D) Autos penceresi

DEĞERLENDİRME

Sorulara verdiğiniz cevaplar ile cevap anahtarını karşılaştırınız, cevaplarınız doğru ise bir sonraki öğrenme faaliyetine geçiniz. Yanlış cevap verdiyseniz öğrenme faaliyetinin ilgili bölümüne dönerek konuyu tekrar ediniz.

ÖĞRENME FAALİYETİ–2

AMAÇ

Program çalışırken değişkenlerin değerlerini takip edebileceksiniz.

ARAŞTIRMA

Bu faaliyet öncesinde yapmanız gereken araştırmalar şunlardır:

1. Değişken değerlerinin izlenmesi bize ne tür faydalar sağlar? Araştırınız.
2. Immediate penceresini ve bu pencere ile yapılabilecek işlemleri araştırınız.

Araştırma işlemleri için internet ortamı ve görsel programlama dilini anlatan kaynak kitaplardan faydalanınız.

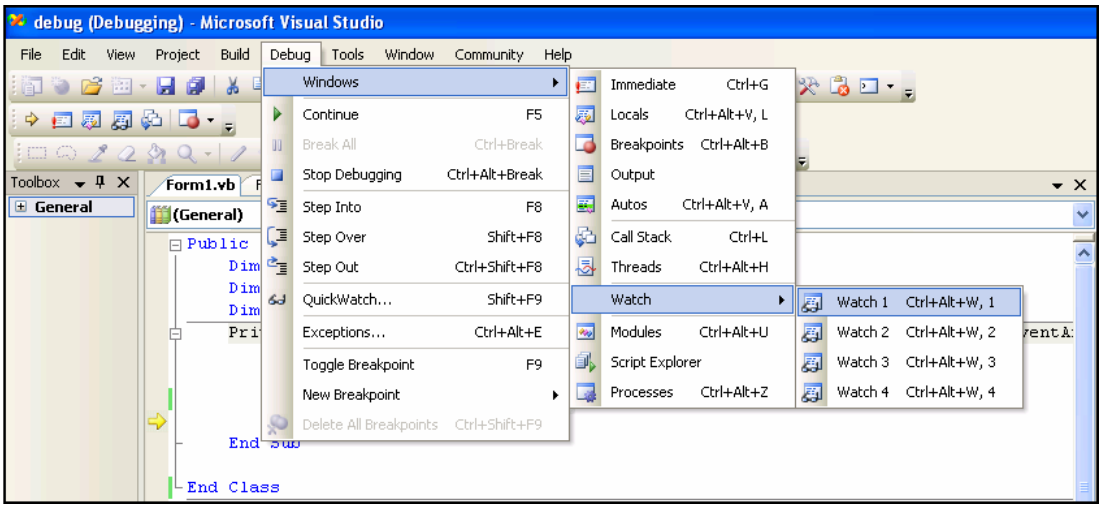
2. DEĞİŞKENLERİN DEĞERLERİNİ TAKİP ETME

Programcıların yaptıkları programların çalışmasını incelerken dikkat edecekleri en önemli hususlardan biri değişken değerleridir. Değişken değerleri programcının istediği şekilde değişiyorsa programın çalışması açısından sorun yok demektir. Program yazıldıktan sonra değişken değerlerinin takip edilmesi o programın doğru çalışıp çalışmadığını kontrol açısından önemlidir. Eğer programın çalışmasında sorun varsa değişken takibi ile oluşacak hataları yerinde görüp müdahale edebilir. Bu işlem için Visual Basic.Net programlama dilinde Watch penceresi kullanılmaktadır. Bu pencerenin kullanımı ve özellikleri aşağıda anlatılacaktır.

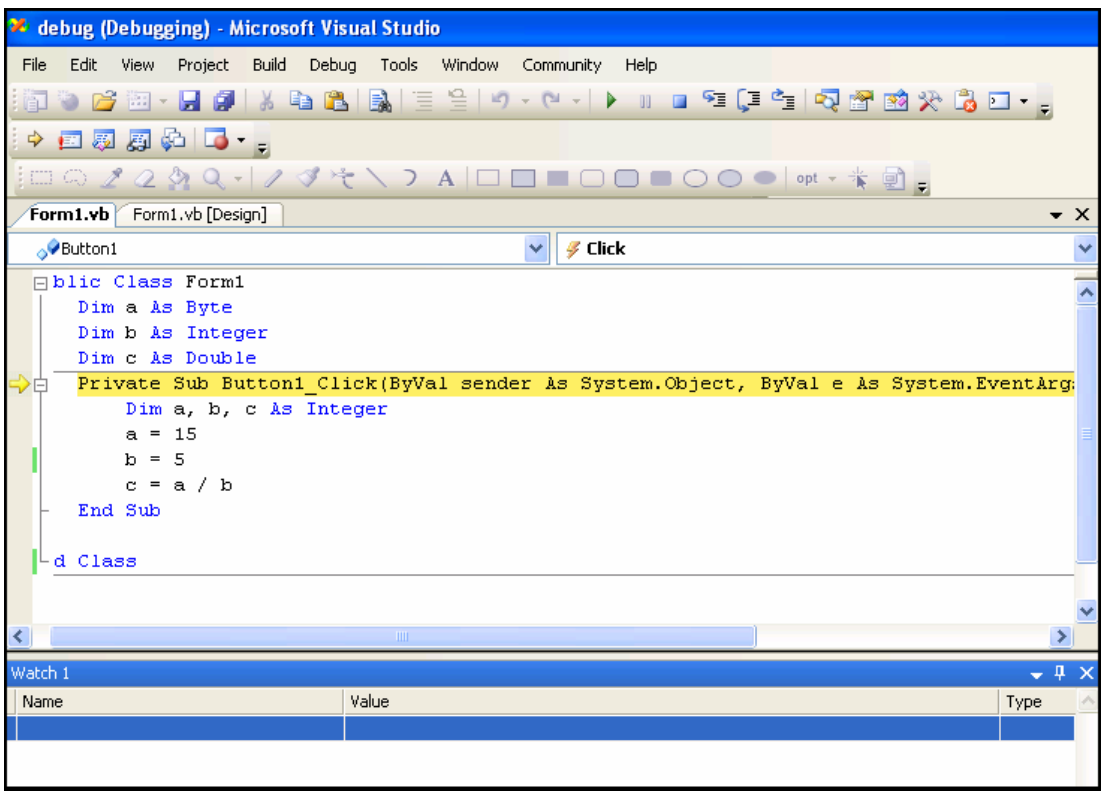
2.1. “Watch” Penceresi

Watch penceresi değişken değerlerinin takibi için kullanılan bir penceredir. Öğrenme faaliyeti 1’de anlatılanlar Autos penceresinde değişken değerlerinin takibi için kullanılır. Fakat Autos penceresi o an aktif olan değişkenlerin takibinde kullanılır. Değişken kullanılmıyorsa otomatik olarak Autos penceresinden silinir.

Değişkenlerin ve özelliklerin değerlerini program çalıştığı sürece görmek istediğimizde Watch penceresini kullanmanız gerekir. Bu pencereyi görmek için programı kesme modunda çalıştırmamız gerekmektedir. Watch pencerelerini Debug menüsünden Windows komutunu tıkladıktan sonra Watch sekmesi yardımıyla Watch 1, Watch 2, Watch 3 ve Watch 4 seçeneklerinden herhangi birini seçerek açabiliriz. Bu pencereler Ctrl+Alt+W kısayol tuşu kullanıldıktan sonra 1, 2, 3, 4 tuşlarından herhangi birine basılarak da açılabilir. 4 ayrı pencerenin olması bize aynı anda birçok değişken ve özelliği inceleme imkânı verir. Herhangi bir değişkeni seçip Add Watch komutunu kullandığımızda değişkeni otomatik olarak watch1 penceresine aktarır. Diğer watch pencerelerine değişken aktarmak istiyorsak değişkeni seçip kopyala komutu ile hafızaya alarak istediğimiz watch penceresine yapıştırabiliriz.

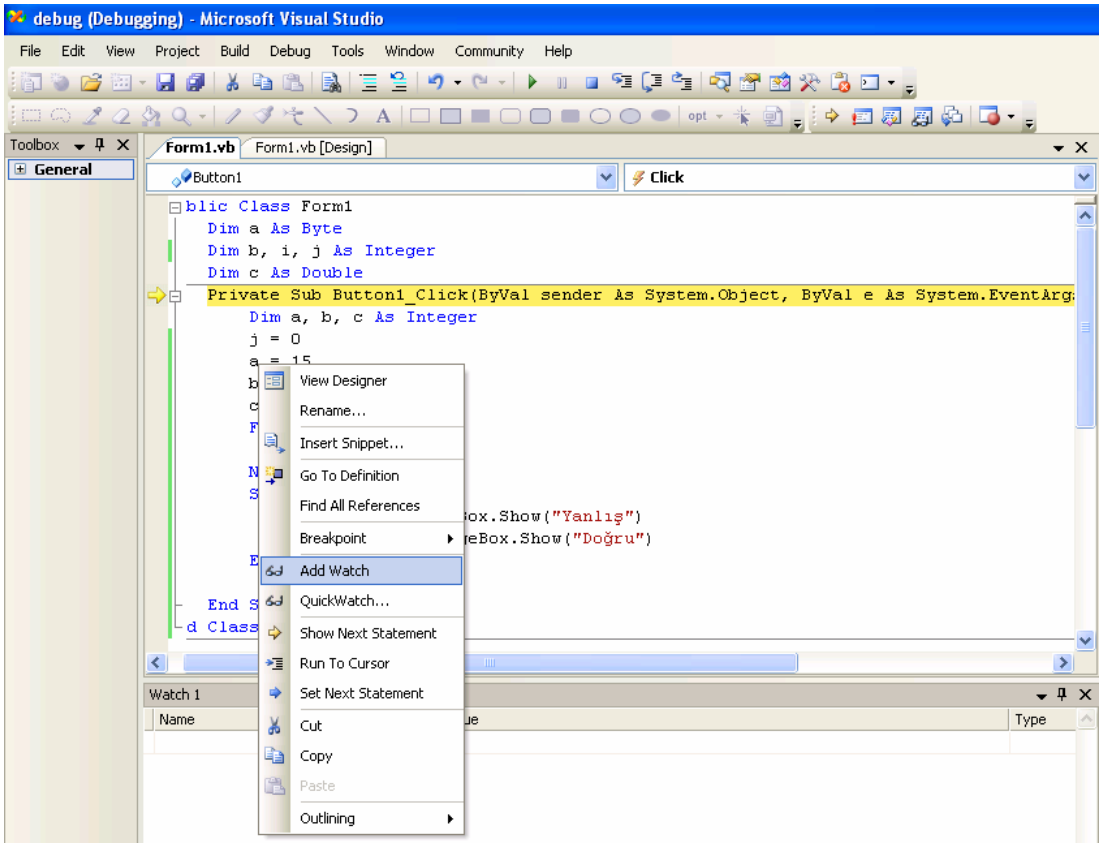


Resim 2.1: Watch pencerelerinin açılması



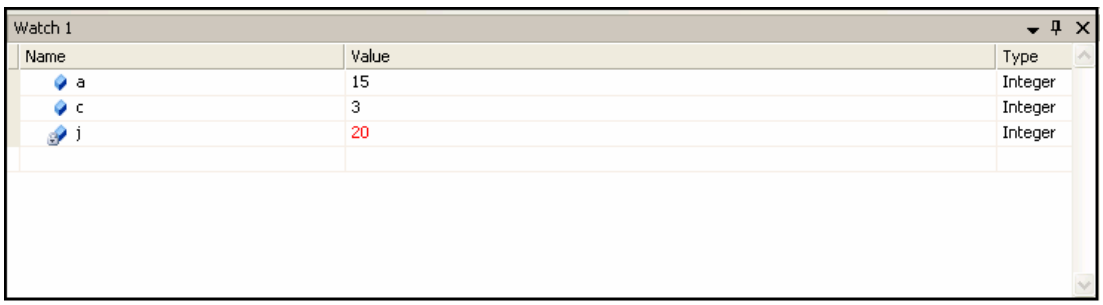
Resim 2.2: Watch penceresi

Watch penceresinde değişken takibi için değişkenlerin bu pencereye eklenmesi gerekir. Bu işlem debugging modunda iken (program adım adım çalıştırılırken) Mouse yardımı ile değişken üzerine gelinip Add Watch seçeneği seçilerek yapılır.



Resim 2.3: Değişkenlerin Watch penceresine eklenmesi

Bir veya birden fazla değişken Watch penceresine eklendikten sonra (Resim 2.4) değişken türleri ve değerleri program çalıştığı sürece takip edilebilir.



Resim 2.4: Değişkenlerin watch penceresinde görünümü

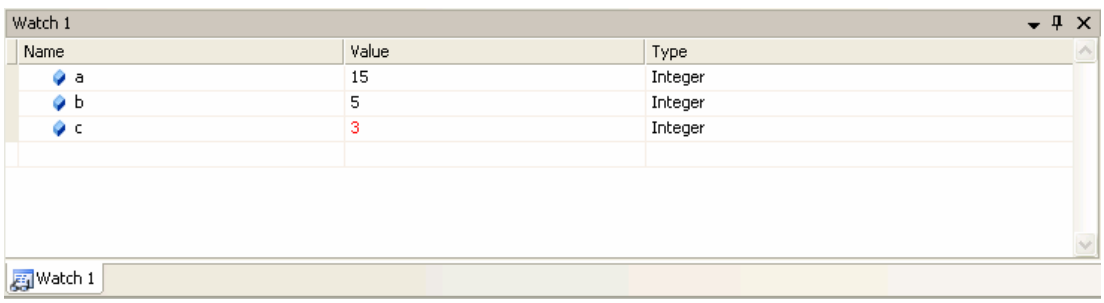
Not: Uzun programlarda programın belli satırlarına kesme işareti konularak program parça parça incelenip değişkenler takip edilebilir. Oluşturulan durak noktalarına gelindiğinde değişkenlerin değerleri watch penceresinde görülebilir.

Watch penceresine eklenen değişkenler pencere içerisinde seçilip delete tuşuna basılarak veya pencere içinde Mouse'un sağ tuşuna tıklanıp "clear all" komutu yardımı ile silinebilir.

Örnek: Aşağıdaki programdaki değişkenleri Watch penceresini kullanarak inceleyelim.

```
Dim a As Byte
   Dim b, i, j As Integer
   Dim c As Double
   Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
      Dim a, b, c As Integer
      j = 0
      a = 15
      b = 5
      c = a / b
      For i = 1 To 15
         j = j + 10
      Next
      Select Case j
         Case 0 : MessageBox.Show("Yanlış")
         Case 150 : MessageBox.Show("Doğru")
      End Select
   End Sub
```

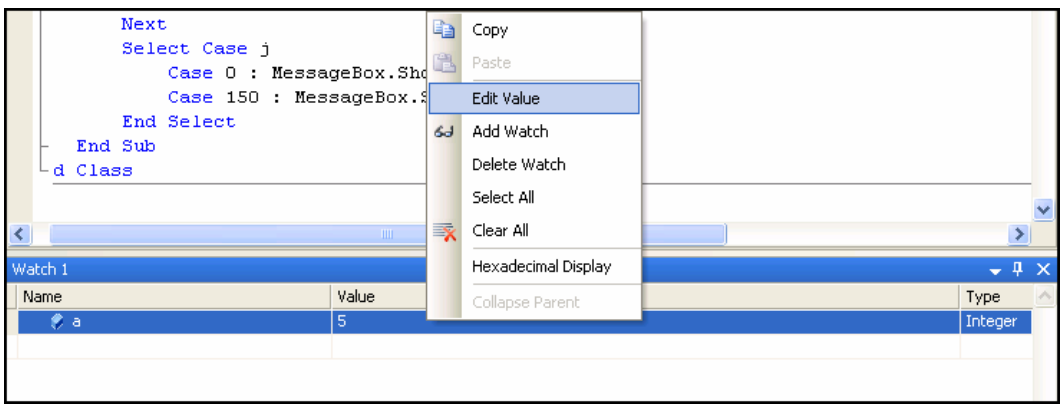
Programda a, b ve c değişkenlerini watch1 penceresinde inceleyelim. Debugging modunda Watch 1 penceresi açılıp a,b ve c değişkenleri bu pencereye aktarıldıktan sonra program adım adım çalıştırılır.



Resim 2.5: a,b,c değişkenlerinin watch penceresi ile incelenmesi

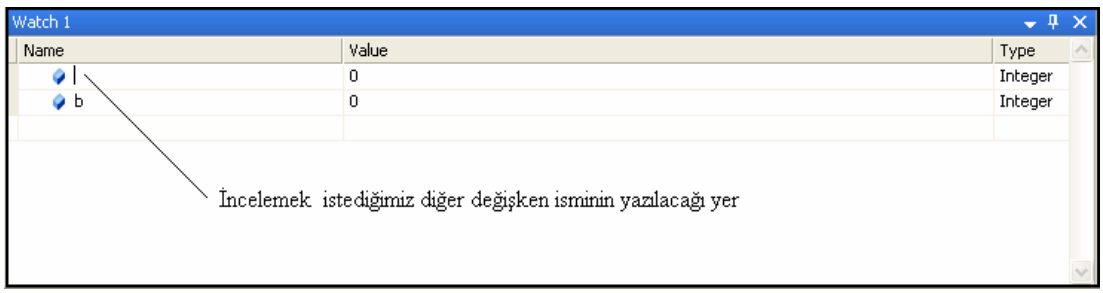
Resim 2.5'te görüldüğü gibi a, b, c değişkenlerinin değerleri watch1 penceresinde izlenebilir. For i=1 To 15 satırına gelindiğinde a,b,c değişkenleri son değerlerini almış olacaktır. Bu durumda a=15, b=5 ve c=3 değerini alacaktır. Eğer bu değerlerde bir sorun varsa programda gerekli düzeltmeler yapılıp ortadan kaldırılabilir.

Not1: Watch penceresi ile değişken değerleri incelenirken program akışını kontrol amacıyla değişken değerleri değiştirilebilir. Bu işlem için Watch penceresinde değişkenin üzerine imleç konumlandırılır, Mouse'un sağ tuşuna basılıp Edit Value komutu seçilir (Resim 2.6). Bu işlemden sonra istediğimiz değeri yazıp programın çalışmasını kontrol edebiliriz.



Resim 2.6: Watch penceresinde değişken değerinin değiştirilmesi

Not2: Watch penceresi ile Name bölümünde yer alan isim Mouse yardımı ile çift tıklanıp programda kullanılan başka bir değişkenin ismi yazılarak farklı değişkenler incelenebilir.

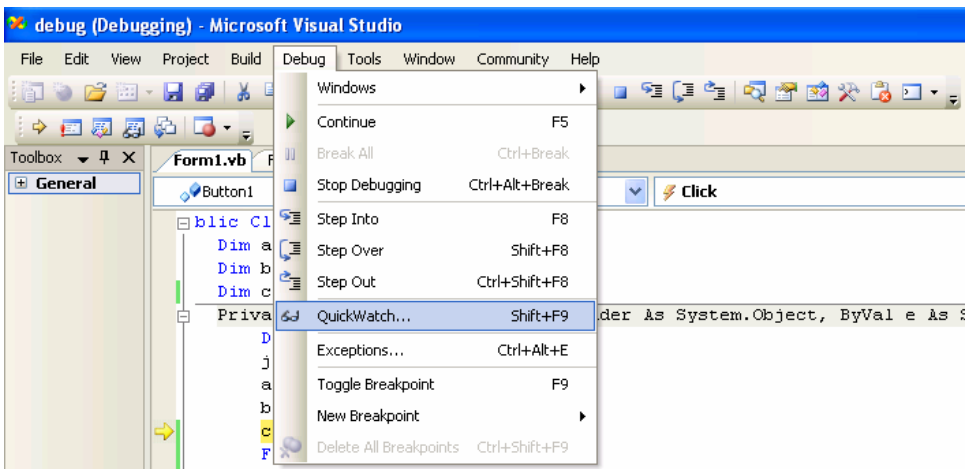


Resim 2.7: Watch penceresinde değişken isminin değiştirilmesi

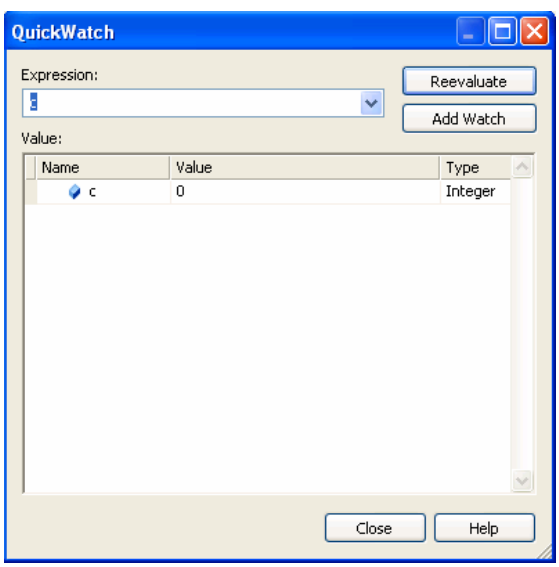
2.2. “QuickWatch” Penceresi

Değişkenlerin değerlerinin takibi için kullanılır. Watch penceresinden farkı değişken veya değişkenlerin anlık değerini gösterir. Programı çalıştırılmaya devam ettiğimizde bu pencere kaybolur. Bu pencerenin açılması için programı debugging modunda çalıştırmak gerekir.

QuickWatch penceresi debug menüsünden QuickWatch sekmesi tıklanarak açılabilir (Resim 2.8). Bu pencereyi hangi satırda açmışsak o satırdaki değişkenin değerini otomatik olarak gösterir (Resim 2.9).

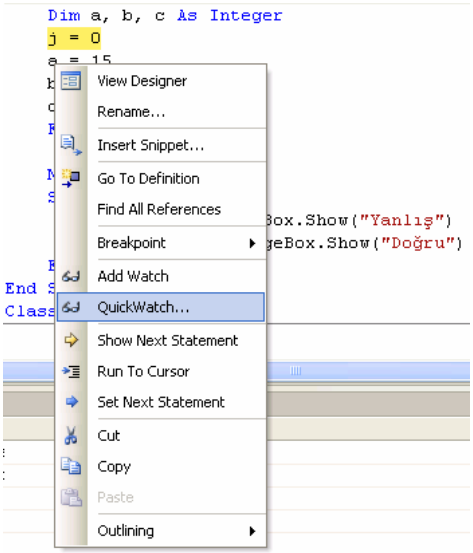


Resim 2.8: QuickWatch penceresinin açılması



Resim 2.9: QuickWatch penceresi

OuickWatch penceresinde değişkenlerin anlık bilgilerini görmek için kullanılan bir başka yöntem ise değişkeni seçip Mouse'un sağ tuşuna basarak açılan menüden "QuickWatch" penceresini açmaktır (Resim 2.10).

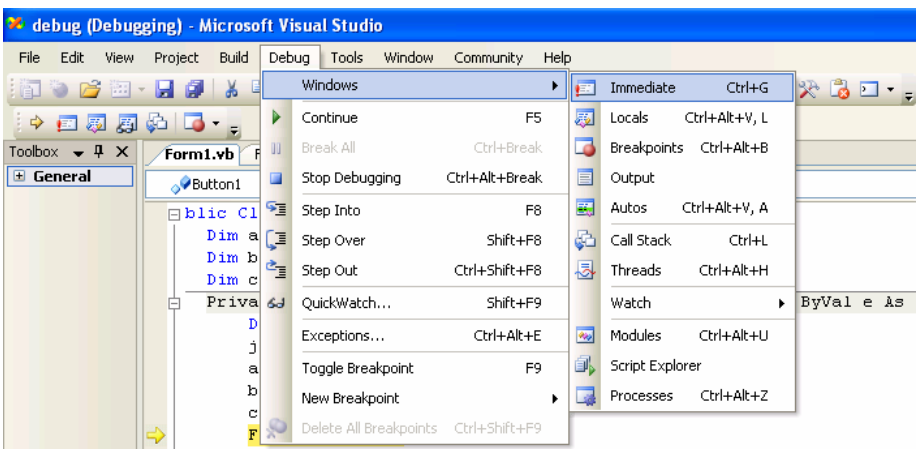


Resim 2.10: QuickWatch penceresinin açılması

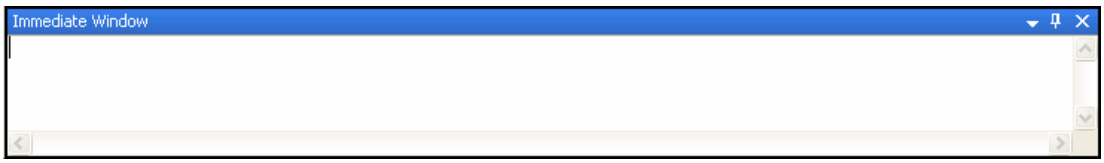
Resim 2.9'da görülen QuickWatch penceresi ile c değişkeninin o anki değerini ve değişken türünü öğrenebiliriz. Ayrıca belirlenen değişkenleri Add watch komutu ile Watch penceresine ekleyebilir, Reevaluate butonu ile değişken değerini güncelleyebiliriz.

2.3. "Command Window – Immediate" Penceresi

Command window kesme (Debugging) modunda değişkenlerin değerlerini değiştirmek veya istenilen komutları çalıştırmak için kullanılır. Bu pencere 2 modda çalışır. Bunlar Immediate modu ve Command modudur. Command window- immediate penceresini açmak için kesme modunda Debug menüsünden Windows sekmesi tıklanarak immediate komutu seçilir veya Ctrl+G kısayol tuşu kullanılır.

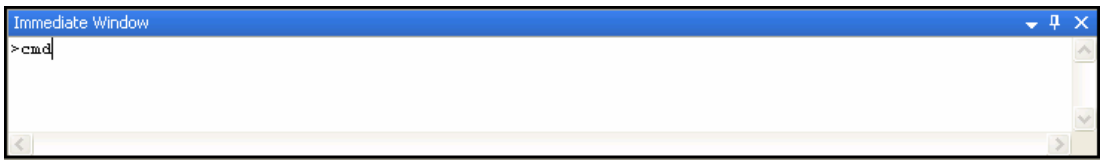


Resim 2.11: Command penceresinin açılması

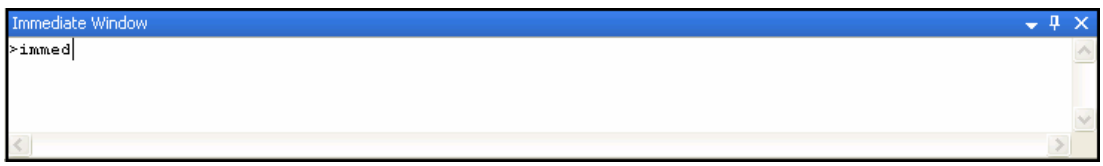


Resim 2.12: Command penceresi

Açılan pencere yardımı ile command ve immediate modları arasında geçiş yapılabilir. Immediate modundan command moduna geçmek için ">cmd" komutunu (Resim 2.13), command modundan immediate moduna geçmek için ise ">immed" komut satırını pencere içine yazmak gerekmektedir (Resim 2.14).

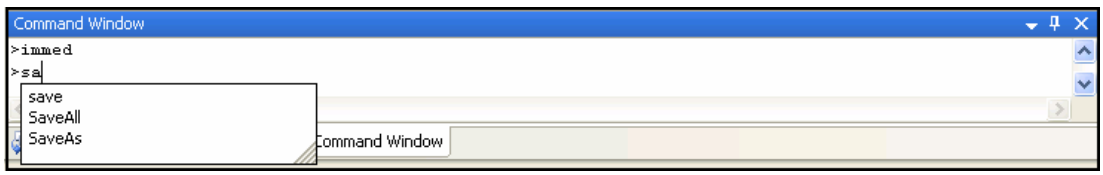


Resim 2.13: Immediate modundan Command moduna geçiş



Resim 2.14: Command modundan Immediate moduna geçiş

Command modu yardımı ile menülerde bulunan birçok komut çalıştırılabilir. Bunlara örnek verecek olursak save, save all, close, code, designer, help, callstack, autos gibi. Bu işlemleri yapmak için pencere içerisinde > işaretinden sonra istediğimiz komutu yazıp enter tuşuna basmamız gerekir (Resim 2.15).

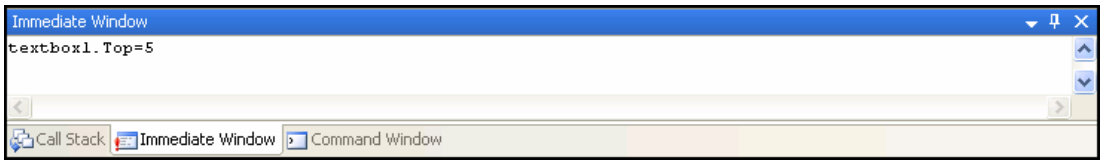


Resim 2.15: Command modunda komut yazılımı

Immediate modunda ise program içerisindeki değişken değerlerini değiştirebilir ve bu değerleri kullanarak işlem yaptırabiliriz. İmmediate penceresinde değişkenin değeri değiştirildiğinde otomatik olarak watch penceresinde de değeri değişir. Bu iki pencere birbirini tamamlar. Bu modu kullanarak değişkenlerin dışında nesne özelliklerini de değiştirebiliriz. Textbox nesnesinin text, top, width vb. özelliklerini değiştirmek gibi.

Not: *Immediate modunda ">" işaretini kullanarak komutlar da çalıştırılabilir.*

Örnek:



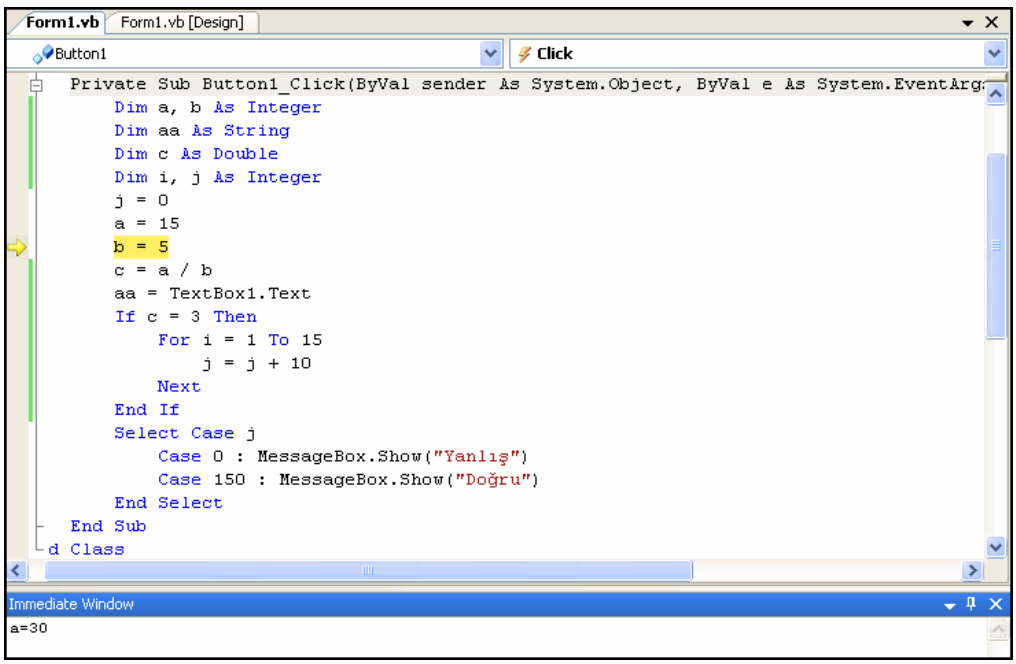
Resim 2.16: Textbox1 nesnesinin top özelliğinin değiştirilmesi

Örnekte textbox1 nesnesinin top özelliğine 5 bilgisi aktarılmıştır. Bu pencereyi kullanarak program kesme modunda çalışırken birçok nesne özelliğini değiştirip daha mükemmel bir program oluşturulabilir.

Örnek: Aşağıdaki örnek programı adım adım (kesme modunda) çalıştırıp b=5 satırına gelindiğinde a değişkeninin değerini immediate penceresi yardımı ile 30 olarak değiştirip program akışındaki değişikliği inceleyelim.

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
    Dim a, b As Integer
    Dim aa As String
    Dim c As Double
    Dim i, j As Integer
    j = 0
    a = 15
    b = 5
    c = a / b
    aa = TextBox1.Text
    If c = 3 Then
      For i = 1 To 15
        j = j + 10
      Next
    End If
    Select Case j
      Case 0 : MessageBox.Show("Yanlış")
      Case 150 : MessageBox.Show("Doğru")
    End Select
  End Sub
```

Normal şartlarda programa müdahale etmediğimizde c değeri 3 olacak, for döngüsü çalıştırılacak bunun sonucunda j değeri 150 olacak ve ekrana "Doğru" ifadesi yazdırılacaktır. Immediate penceresi kullanılarak a değeri 30 yapıldığında ise for döngüsü çalıştırılmayacak, j değeri 0 olacak ve ekrana "Yanlış" ifadesi yazdırılacaktır.

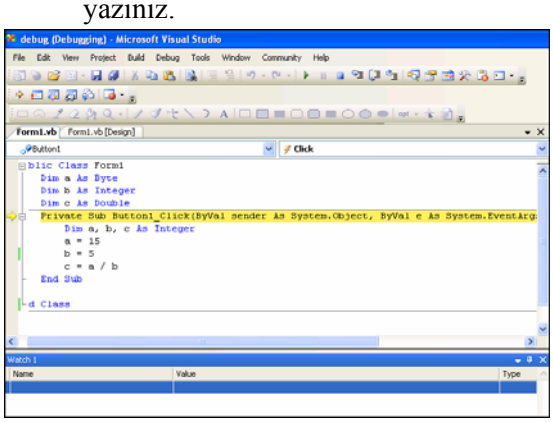


Resim 2.17: a değerinin değiştirilmesi

Değişken değerlerini değiştirerek programın hangi değişken değerine hangi tepkiyi verdiğini veya hangi değişken değerinde hangi işlemin yapılacağı incelenebilir. Bu programcıların yaptığı programı kullanıcıdan önce incelemesini ve oluşabilecek hataları bulup düzeltmesini sağlayacaktır.

Araştırma: Command Window penceresini kullanarak çalıştırılamayan komutları araştırınız.

UYGULAMA FAALİYETİ

İşlem Basamakları	Öneriler
1. Programda istenilen yere F9 kısayolu ile kesme noktası belirtiniz.	Oluşturduğunuz programa iki veya üç kesme noktası ekleyiniz. Bu noktaları breakpoint penceresinde inceleyiniz.
2. Programı çalıştırarak kesme olayını aktifleştiriniz.	Oluşturduğunuz programa kesme noktaları ekleyip programı çalıştırınız.
3. Debug-windows-Watch penceresinde "Name" satırına değişkenin adını yazınız. 	Name özelliğini kullanarak inceleyeceğiniz değişkenleri değiştirebilirsiniz. Watch penceresini açmak için programı kesme modunda çalıştırmalısınız.
4. Adım adım programı çalıştırarak değişken değerini takip ediniz.	İstediğiniz herhangi bir değişkeni Watch penceresine ekleyip değişken değerini takip ediniz.
5. İşlem bitince durak noktasını siliniz.	Parse komunu kullanınız.

ÖLÇME VE DEĞERLENDİRME

A- OBJEKTİF TESTLER (ÖLÇME SORULARI)

Aşağıdaki sorularda verilen ifadeye göre parantez içine doğru ise "D", yanlış ise "Y" yazınız.

1. Visual Basic.Net programlama dilinde 5 adet watch penceresi bulunur. ( )

2. Watch penceresini açmak için Ctrl+Alt+W kısayol tuşu kullanılır. ( )

3. QuickWatch penceresi değişkenleri takip etmek için kullanılır. ( )

4. Command Window penceresi üç modda açılabilir. ( )

5. Command Window penceresinde komutları çalıştırmak için command modu kullanılır. ( )

6. Command modundan Immediate moduna >cmd komutu yardımı ile geçilir. ( )

7. Watch penceresinde nesne özellikleri değiştirilebilir. ( )

8. QuickWatch penceresi ile sadece 1 değişken izlenebilir. ( )

DEĞERLENDİRME

Sorulara verdiğiniz cevaplar ile cevap anahtarını karşılaştırınız, cevaplarınız doğru ise bir sonraki öğrenme faaliyetine geçiniz. Yanlış cevap verdiyseniz öğrenme faaliyetinin ilgili bölümüne dönerek konuyu tekrar ediniz.

MODÜL DEĞERLENDİRME

PERFORMANS TESTİ (YETERLİK ÖLÇME)

Modül ile kazandığınız yeterliği aşağıdaki kriterlere göre değerlendiriniz.

DEĞERLENDİRME ÖLÇÜTLERİ	Evet	Hayır
Marjin çubuğuna tıklayıp durak noktası oluşturdunuz mu?		
Programı çalıştırarak mouse ile değişkenler üzerinden bilgi aldınız mı?		
"Autos" penceresinde değişkenleri gördünüz mü?		
Programı adım adım çalıştırdınız mı?		
"Debug –Exceptions" ile istisnai durumları ayarladınız mı?		
Programda istenilen yere F9 kısayolu ile kesme noktası belirttiniz mi?		
Programı çalıştırarak kesme olayını aktifleştirdiniz mi?		
Debug-windows-Watch penceresinde "Name" satırına değişkenin adını yazdınız mı?		
Adım adım programı çalıştırarak değişken değerini takip ettiniz mi?		
İşlem bitince durak noktasını sildiniz mi?		

DEĞERLENDİRME

Yaptığınız değerlendirme sonucunda eksikleriniz varsa öğrenme faaliyetlerini tekrarlayınız.

Modülü tamamladınız, tebrik ederiz. Bu modül, program yaparken sürekli kullanacağınız bilgileri içerdiğinden belli zamanlarda bu modülü tekrar gözden geçiriniz.

Öğretmeniniz size çeşitli ölçme araçları uygulayacaktır. Öğretmeninizle iletişime geçiniz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ–1 CEVAP ANAHTARI

1	D
2	D
3	D
4	Y
5	Y
6	D
7	B
8	B

ÖĞRENME FAALİYETİ-2 CEVAP ANAHTARI

1	Y
2	D
3	D
4	Y
5	D
6	Y
7	D
8	D

Cevaplarınızı cevap anahtarları ile karşılaştırarak kendinizi değerlendiriniz.

ÖNERİLEN KAYNAKLAR

- www.yazgelistir.com
- www.vbturk.net
- www.programlama.com
- www.yazilimgrubu.com
- www.findikkurdu.com
- www.yazilimuzmanı.com
- www.godoro.com
- www.ceturk.com
- www.dotnetturk.com
- www.startvbdotnet.com
- www.vbasicmaster.com
- http://bilisim-kulubu.com
- http://msdn.microsoft.com/netframework/
- www.wikipedia.org
- www.freevbcode.com

KAYNAKÇA

- BOWMA Richard, **Visual Basic.NET**, Hungry Minds inc., New York 2002.
- DAVIS Harold, **Visual Basic.NET Programlama Kılavuzu**, ALFA Yayınevi, İstanbul, 2002.
- EVJEN Bill, Jason Beres, **Visual Basic.NET Bible,** Hungry Minds inc., New York 2002.
- GRUNDGEIGER Dave, **Programming Visual Basic.NET**, O'Reillly 2002.
- HALVORSON Michael, **Adım Adım Microsoft Visual Basic.Net**, Arkadaş Yayınevi, Ankara 2002.
- HALVORSON Michael, **Adım Adım Microsoft Visual Basic 6.0 Professional,** Arkadaş Yayınevi, Ankara 2002.
- PALA Zeydin, Microsoft Visual Basic.NET, Türkmen Kitapevi, 2003.