T.C.
MİLLÎ EĞİTİM BAKANLIĞI

BİLİŞİM TEKNOLOJİLERİ

BASİT KODLAR

Ankara, 2017

- Bu bireysel öğrenme materyali, mesleki ve teknik eğitim okul/kurumlarında uygulanan çerçeve öğretim programlarında yer alan kazanımların gerçekleştirilmesine yönelik öğrencilere rehberlik etmek amacıyla hazırlanmıştır.
- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.
- PARA İLE SATILMAZ.

İÇİNDEKİLER

AÇIKLAMALAR

ALAN	**Bilişim Teknolojileri**
DAL	**Alan Ortak**
MODÜLÜN ADI	**Basit Kodlar**
MODÜLÜN SÜRESİ	**40/35**
MODÜLÜN AMACI	Bireye/öğrenciye; iş sağlığı ve güvenliği tedbirlerini alarak basit kodlar yazma ile ilgili bilgi ve becerileri kazandırmaktır.
MODÜLÜN ÖĞRENME KAZANIMLARI	**1.** Program için gerekli değişken ve sabitleri kullanabileceksiniz **2.** Verilen problemdeki işlemlere uygun operatörleri kullanabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Bilgisayar laboratuvarı. **Donanım:**Kâğıt, kalem, akış diyagramları ile ilgili panolar, bilgisayar, lisanslı programlama yazılımı.
ÖLÇME VE DEĞERLENDİRME	Bireysel öğrenme materyali içinde yer alan ve her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendireceksiniz. Öğretmeniniz, bireysel öğrenme materyalinin sonunda, ölçme araçları (uygulamalı faaliyetler, iş ve performans testleri, çoktan seçmeli / doğru-yanlış ve boşluk doldurmalı sorular, vb.) kullanarak kazandığınız bilgi ve becerileri ölçüp değerlendirecektir.

GİRİŞ

Sevgili Öğrencimiz,

Günümüzdegörsel programlama dilleri hızla gelişmekte ve kullanımı yaygınlaşmaktadır. Bu materyalde bir programlama dilinde kullanılabilecek basit kodlar ve uygulamalar anlatılmaktadır. Bir programlama dilinde kullanılan değişken ve sabit kavramlarını tanımlayabilecek, değişken ve sabitlerinprogram içinde kullanım şekillerini kavrayabilecek; değişken ve sabit üzerinde işlem yapabilmek için operatörlerin kullanımlarını öğreneceksiniz.

Bu bireysel öğrenme materyalinde anlatılanlarla sizler programlama mantığını ve becerisini çok daha kolay kavrayabileceksiniz.

Sevgili öğrencimiz, okulda öğreneceğiniz her konu, yaptığınız her uygulama ve tamamladığınız her materyal bilgi dağarcığınızı geliştirecek ve ileride atılacağınız iş yaşantınızda karşılaşacağınız zorlukları yenmede size yardımcı olacaktır. Eğitim sürecinde daha özverili çalışır ve çalışma disiplini kazanırsanız önünüze çıkacak engellerin üstesinden kolaylıkla gelebilecek ve iş hayatınızda başarılı olabileceksiniz.

ÖĞRENME FAALİYETİ–1

ÖĞRENME KAZANIMI

Program için gerekli değişken ve sabitleri kullanabileceksiniz.

ARAŞTIRMA

- Çevrenizde değişken olarak nitelendirebileceğiniz durumları listeleyiniz.
- Bir gün içinde kullandığınız nesneleri türlerine göre gruplandırınız.

1. DEĞİŞKENLER VE SABİTLER

1.1. Değişkenler

Değişkenler bir programlama dilinde verilerin depolanma alanlarını temsil eder.Tanımlanan her değişkene bellek bölgesinden bir alan ayrılır. Bu bellek bölgesine okuma ve yazma işlemleri ise değişken ismi üzerinden sağlanır. Genel olarak değişkenler aşağıdaki şekilde tanımlanır:

<veri tipi><değişken adı>;

Örneğin;

int i;

Yukarıdaki örnekte bir int(sayı) veri tipinde bir değişken tanımlanmıştır. Böylelikle bellek bölgesinde bir veri saklamak ve ileride kullanmak üzere 4 byte'lık bir alan açılmıştır. Program içinde bu bellek bölgesine erişmek için değişken ismi olan i ifadesini kullanılır.

Örneğin;

```
int i;   //i adında bellekte 4 bytelık bir bölge aç.;
i = 5;  //i adının temsil ettiği bellek bölgesine 5 değerini yaz.;
```

Örnek 1.1-1:

```
booldogrumu =false;

doubleyuzde = 98.32, ortalama = 35.32;

charkarakter ='A';
```

Bazı programlama dillerinde yerel bir değişken tanımlanıyorsa bu değişken kullanılmadan önce bir değer ataması yapılmak zorundadır.

```
staticvoidMain(string[] args)
{
    intsayi;
    //sayi = 5;
Console.WriteLine(sayi);
Console.ReadKey();
}
```

Yukarıdaki kod blokunda sayi değişkenine değer atanmadığı için hata verecektir**(Use of unassigned local variable 'sayi').** Çünkü sayi değişkeni yerel bir değişkendir. Eğer `//sayi = 5;` açıklama satırındaki `//`işaretlerini siler ve bu satır koda dâhil edilerse hatanın ortadan kalktığı görülür.

Tanımlanan yerel değişkenler sadece tanımlandıkları bloktan erişebilir. Yaşam döngüleri sadece o blok olduğu için başka bir bloktan erişilemez.

Örnek 1.1-2:

```
namespace megepBilisim
{
classProgram
    {
staticvoid Main(string[] args)
        {
            { //Birinci blok
int a = 10;
            }
            {//İkinci blok
int a = 20;
            }
Console.WriteLine(a);// a değişkeni bu blokta tanımlanmadığı için
program hata verecektir.
```

```
        }
    }
}
```

Yukarıdaki örnekte birinci ve ikinci blokta tanımlanan "a" isimli değişkenler sadece kendi bloklarında geçerlidir. Main bloğunda kullanılmaya çalışıldığı zaman program hata verecektir.

+= operatörü: Eşitliğin sağındaki değerle eşitliğin solundaki değişken değerini toplayıp tekrar eşitliğin solundaki değişkene atar.

1.2. Değişkenleri İsimlendirme Kuralları

Değişkenlerin isimleri alfabede bulunan karakterlerle veya _(alt çizgi) ile başlamalıdır. Ama ilk harf hariç diğer karakterler sayı olabilir.

Bazı programlama dilleri büyük ve küçük harf duyarlıdır. Yani**Sayi, sayi**ve**SAYI**hepsi ayrı değişken olarak algılanır.

Değişken isimleri birden fazla kelime olduğu zaman; kelimelerin arasına boşluk konmaz. Bu tür değişkenleriya kelimeleri birleştirerek veya kelimeler arasına_(alt çizgi) karakteri konararak isimlendirilir.

Değişkenlerin isimleri !, ?, {,] gibi karakterler içeremez.

Programlama dili için tanımlanmış anahtar kelimeler de değişken isimleri olarak kullanılamaz.

Değişken tanımlarken boşluk kullanılmaz.

Visual Studio desteklese de Türkçe karakter (ç, ı, ğ, ö, ş) kullanılmamaya çalışılır.

- **Standart yazım şekilleri**

Camel notasyonunda isim küçük harfle başlar, değişken isminde birden fazla kelime geçiyor ise isimdeki diğer kelimeler büyük harfle başlar.

Camel Notasyonu:

```
maas;
maasMiktari;
massMiktariAciklama;
```

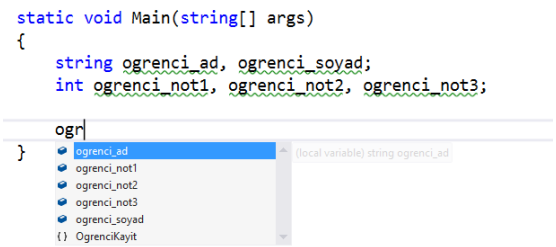
Pascal notasyonunda kelime büyük harfle başlar. Camel notasyonunda da olduğu gibi diğer kelimelerde büyük harfle başlar.

Pascal Notasyonu:

```
Maas();
MaasHesapla();
```

Bu notasyonların kullanımı mecburi değildir. Fakat sürekli olarak bu tür bir notasyona uyarak kodlar yazılırsa kodlar daha anlaşılır bir hâle gelir.

Değişkenlere isim verirken kullanılacağı amaç doğrultusunda isim vermek çok önemlidir. Bu yazılan kod uzadığı zaman o değişkene daha rahat ulaşmayı sağlayacaktır. Örneğin bir öğrencinin üç notunun ortalamasını alan bir program yazılacaksa değişken isimlerinin başına *ogr* yada *öğrenci* yazılarsa ilerleyen zamanlarda bu bilgiye ulaşmak çok daha kolay olacaktır. Visual studio yazılımı kullanıcı*ogr* yazdığındakullanıcıya *ogr* ile başlayan bütün değişkenleri listeleyecektir. Bu da istenen değişkene çok daha kolay ulaşmayı sağlacaktır. Bunu alışkanlık hâline getirmek ileride kod yazmayı çok kolaylaştıracaktır.



1.3. VeriTipleri

.Net de iki çeşit veri tipi vardır:

- Değer tipleri (value type)
- Referanstipleri(reference type)

Değişkenler bellekte bulunan verilerdir. Bir değişken kullanıldığı zaman o değişkenin bellekte bulunduğu yerdeki bilgi kullanılır. Değer tipleri belleğin 'stack' bölgesinde saklanır ve veriyi doğrudan bellek bölgesinden alırken referans tipleri bellekte 'heap' alanında saklanır. Yani referans tipleri içinde veri değil bellekteki 'heap' alanının adres bilgisini tutar.

int, double, float gibi veri tipleri değer tiplerine örnek gösterilebilir. Herhangi bir sınıf türü ise referans tipine örnek gösterilebilir. Değer tipleri birbirine eşitlenirken değişkenin barındırdığı değer bir diğer değişkene kopyalanır. Böylece iki farklı bağımsız değişken oluşur. Referans tipleri ise eşitleme sırasında değişkenlerin taşıdıkları veri değil 'heap' bölgesinde işaret ettikleri adres kopyalanır. Böylece iki referans değişkeni birbirine eşitlenir.Daha sonra bunlardan birinde bulunan veri değiştirildi ise otomatik olarak diğer referans değişkeninin değeri de değişir. Çünkü adreste bulunan veri değişince bu adresi işaret eden iki değişkende yeni veri bilgisine ulaşır.

Toplam 15 veri tipi vardır bunlardan 13'ü değer tipindedir, 2'si ise referans tipindedir.

Değer tipleri

Adı	CTS Karşılığı	Açıklama	Max ve Min aralık yada değeri
sbyte	System.Byte	8 bit işaretli tam sayı	-128 : 127
short	System.Int16	16 bit işaretli tam sayı	-32.768 : 32.767
int	System.Int32	32 bit işaretli tam sayı	-2.147.483.648 : 2.147.483.647
long	System.Int64	64 bit işaretli tam sayı	-9.223.372.036.854.775.808 : -9.223.372.036.854.775.807
byte	System.Byte	8 bit işaretsiz tam sayı	0,177083333
float	System.Single	32 bit tek kayan sayı	+yada - $1,5*10^{-45}$: + ya da - $3,4*10^{38}$
double	Sytem.Double	64 bit çift kayan sayı	+yada - $5*10^{-324}$: + ya da - $1,7*10^{308}$
bool	System.Boolean		true ya da false
char	System.Char	Karakterleri temsil eder	16 Unicode karakterleri

Tablo 1.1: Değer tip listesi

Referans tipleri

Adı	CTS Karşılığı	Açıklama
object	System.Object	Bütün veri türlerinin türediği kök eleman
string	System.String	Unicode karakterlerinden oluşan string

Tablo 1.2: Referans tip listesi

1.4. Sabitler

Program boyunca sabit kalacak veriler için kullanılan tanımlamalardır. Bir sabiti tanımlamak için **const**anahtar kelimesi kullanılır.

- Sabitler tanımlanırken ilk değer ataması yapılmak zorundadır.
- Program boyunca sabit değeri değiştirelemez.

Kullanımı: **const***<veri tipi><değişken adı>=değer;*

Örnek 1.4-1:

```
constdoublePI = 3.14;

constdoublePI;
```

Yukarıdaki tanımlanan sabitlerden birincisi doğru ikincisi ise yanlış bir tanımlamadır. Çünkü sabitler tanımlanırken ilk değer ataması yapılmak zorundadır.

Sabit kullanım hata mesajları :

- **The left-hand side of an assignment must be a variable, property or indexer:** Tanımlanan sabit değişkenin değeri değiştirilmeye çalışıldığı zaman oluşacak hata mesajıdır. Bu yüzden sabit değişken tanımlanırken atanan değer program boyunca sabit kalmalıdır.

Örnek 1.4-2:

```
constdouble pi = 3.14159265;

pi = 2 * pi; //Burada pi değerini değiştirdiğiniz için hata mesajı
alırsınız.
```

- **A const field requires a value to be provided:** Tanımlanan sabit değişkene ilk değer ataması yapılmadıysa oluşan hata mesajıdır. Dolayısıyla sabit değişken tanımlanırken ilk değer ataması yapılmalıdır.

Örnek 1.4-3:

```
constint pi; // Burada pi sabitine değer atanmadığı için hata mesajı alırsınız.
```

1.5. Atama İşlemi

= operatörü: Genel atama işlemlerinde kullanılır. Eşitliğin sağındaki **değer** eşitliğin solundaki **değişkene** atanır.

Örnek 1.5-1:

int x, y=5; // 5 değerini y değişkenine atamak için = operatörü kullanılmıştır.

x = y + 2; // y değişken değeri ile 2 sayısı toplanarak x değişkenine atamak için = operatörü kullanılmıştır.

- Bir değişkene değer atama işlemi tanımlarken yapılabilir.
- Bir değişkene değer atama işlemi yukardaki örnekte olduğu gibi program içinde herhangi bir satırda yapılabilir.
- Bir veri tipi altında birden fazla isimle farklı değişkenler tanımlanabilir.

Örnek 1.5-2:

int x=0, y=0,z=0;

x += 5;//x'e 5 ekle ve x'e eşitle 2.yol x = x + 5 şeklinde de yazılabilir.

y += 7;//y'ye 7 ekle ve y'ye eşitle 2.yol y = y + 7 şeklinde de yazılabilir.

z += x;//z'ye x'i ekle ve z'ye eşitle 2.yol z = z + x şeklinde de yazılabilir.

İşlem sonucu: x=5, y=7, z=5 olur.

Not:Bir bir artırma işlemi için x+=1 (veya x=x+1) yerine x++ işlemi kullanılabilir.

Örnek 1.5-3:

```
int x=0, y=0,toplam;

x++;//x'i bir artır

y++;//y'yi bir artır

toplam = x + y;//x ve y'yi toplayarak toplam değişkenine ata.
```

İşlem sonucu: x=1, y=1, toplam=2 olur.

- **++ değişkenden sonra kullanılırsa önce atama işlemi yapılır sonra artırma yapılır.**

Örnek 1.5-4:

```
int x=0, y=0,toplam;

x=y++;

toplam = x + y;
```

önce x y'ye eşitlenir,daha sonra y artırılır. İşlem sonucu: x=0, y=1, toplam=1 olur.

- **++ değişkenden önce kullanılırsa önce artırım yapılır daha sonra atama işlemi yapılır.**

Örnek 1.5-5:

```
int x=0, y=0,toplam;

x=++y;

toplam = x + y;
```

önce y artırılır daha sonra x y'ye eşitlenir.İşlem sonucu: x=1, y=1, toplam=2 olur.

-= operatörü:Eşitliğin sağındaki değeri eşitliğin solundaki değişken değerinden eksilterek tekrar eşitliğin solundaki değişkene atar.

Örnek 1.5-6:

int x=50, y=50, z=100;

x -= 5;//x'den 5'i çıkar ve x'e eşitle 2.yol x = x - 5 şeklinde de yazılabilir.

y -= 7;//y'den 7'yi çıkar ve y'ye eşitle 2.yol y = y - 7 şeklinde de yazılabilir.

z -= x;//z'den x'i çıkar ve z'ye eşitle 2.yol z = z - x şeklinde de yazılabilir.

İşlem sonucu: x=45, y=43, z=55 olur.

Not: Bir bir azaltma işlemi için x-=1 (veya x=x-1) yerine x-- işlemi kullanılabilir.

Örnek 1.5-7:

int x=20, y=10,fark;

x--;//x'i bir azalt

y--;//y'yi bir azalt

fark = x - y;//x ve y'yi çıkararak fark değişkenine ata.

İşlem sonucu: x=19, y=9, fark=10 olur.

- **-- değişkenden sonra kullanılırsa önce atama işlemi yapılır, sonra azaltma yapılır.**

Örnek 1.5-8:

int x=10, y=10,fark;

x=y--;

fark = x - y;

önce x y'ye eşitlenir,daha sonra y azalttırılır. İşlem sonucu: x=10, y=9, fark=1 olur.

- **-- değişkenden önce kullanılırsa önce azaltma yapılır daha sonra atama işlemi yapılır.**

Örnek 1.5-9:

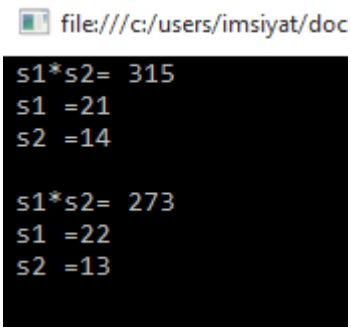
int x = 10, y = 10, fark;

x = --y;

fark = x - y;

önce y artırılır daha sonra x y'ye eşitlenir.İşlem sonucu: x=9, y=9, fark=0 olur.

```
static void Main(string[] args)
{
    int s1, s2, sonuc;
    s1 = 20;
    s2 = 15;
    sonuc= ++s1 * s2--; //Bu işlemde önce s1 değeri 1 artarak 21 oluyor ve
    //s2 değeri 15 olarak kullanıldıktan sonra 14'e iniyor.
    Console.WriteLine("s1*s2= "+sonuc); //375
    Console.WriteLine("s1 =" + s1); //s1=21
    Console.WriteLine("s2 =" + s2); //s2=14
    Console.WriteLine();
    sonuc = s1++ * --s2; //Bu işlemde s1 21 iken kullanılıyor ve 1 artıyor
    //s2 1 düşerek 13 olduktan sonra kullanılıyor.
    Console.WriteLine("s1*s2= " + sonuc);//273
    Console.WriteLine("s1 =" + s1); //22
    Console.WriteLine("s2 =" + s2); //13
    Console.ReadLine();
}
```



***= operatörü:**Eşitliğin sağındaki değerle eşitliğin solundaki değişken değeri çarpılıp tekrar eşitliğin solundaki değişkene atar.

Örnek 1.5-10:

int x = 2, y = 3, z = 2;

x *= 2;//x ile 2'i çarp ve x'e eşitle 2.yol x = x * 2 şeklinde de yazılabilir.

y *= 2;//y ile 2'yi çarp ve y'ye eşitle 2.yol y = y * 2 şeklinde de yazılabilir.

z *= x;//z ile x'i çarp ve z'ye eşitle 2.yol z = z * x şeklinde de yazılabilir.

İşlem sonucu: x=4, y=6, z=8 olur.

/= operatörü:Eşitliğin solundaki değişken değerini eşitliğin sağındaki değere bölerek tekrar eşitliğin solundaki değişkene atar.

Örnek 1.5-11:

int x = 4, y = 10, z = 64;

x /= 2;//x'i 2'ye böl ve x'e eşitle 2.yol x = x / 2 şeklinde de yazılabilir.

y /= 2;//y'yi 2'ye böl ve y'ye eşitle 2.yol y = y / 2 şeklinde de yazılabilir.

z /= x;//z'yi x'e böl ve z'ye eşitle 2.yol z = z / x şeklinde de yazılabilir.

İşlem sonucu: x=2, y=5, z=32 olur.

1.6. ÇıkışİşlemleriÇıkışİşlemleri

1.6.1. Bir Metin İfadesini Ekrana Yazdırma

Bir metin ifadesini ekrana yazdırmak için iki metot kullanılır:

- **1. metot Console.Write():** Yazdırma işleminden sonra imleç yazdırılan ifadenin yanında bekler.

Örnek 1.6-1:

```
staticvoid Main(string[] args)
        {
Console.Write("Merhaba Dünya");
Console.ReadKey();// Klavyeden bir tuşa basılana kadar bekle.
        }
```

Çıktısı:

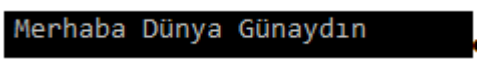
Fotoğraf1.1: Örnek 1.6-1 ekran çıktısı

Dikkat edilerse imleç yazının sonunda beklemektedir. Bir başka metin yazılmaya çalışıldığı zaman imlecin bulunduğu yerden devam edilir.

Örnek 1.6-2:

```
staticvoid Main(string[] args)
        {
Console.Write("Merhaba Dünya");
Console.Write("Günaydın");
Console.ReadKey();
        }
```

Çıktısı:



Fotoğraf 1.2: Örnek 1.6-2 ekran çıktısı

- **2. metot Console.WriteLine():**Yazdırma işleminden sonra imleç yazdırılan ifadenin alt satırında bekler.

Örnek 1.6-3:

```
staticvoid Main(string[] args)
        {
Console.WriteLine("Merhaba Dünya");


Console.ReadKey();
        }
```

Çıktısı:

Fotoğraf 1.3:Örnek 1.6-3 ekran çıktısı

Örnek 1.6-4:

```
staticvoid Main(string[] args)
        {
Console.WriteLine("Merhaba Dünya");
Console.WriteLine("Günaydın");


Console.ReadKey();
        }
```

Çıktısı:

Fotoğraf 1.4:Örnek 1.6-4 ekran çıktısı

Program çalıştırıldığında ikinci metin bir alt satıra yazılmış olur.

Çıkış parametreleri:

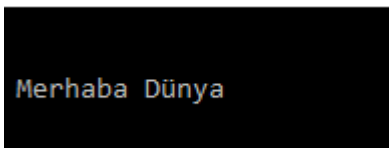
\n: Bir alt satıra geçmek için kullanılır.

\r: Paragraf başı yapmak için kullanılır.

Örnek 1.6-5:

```
staticvoid Main(string[] args)
        {
Console.WriteLine("\n\nMerhaba Dünya");
Console.ReadKey();
        }
```

Çıktısı:



Fotoğraf 1.5:Örnek 1.6-5 ekran çıktısı

Ekran çıktısına bakıldığında iki tane \n kullanıldığı için yazıiki satır aşağıdan başlamıştır.

1.6.2. İlkDeğer Atanan Değişken Değerini Ekrana Yazdırma

Değişken tanımlamadan daha önce bahsedildi. Burada değer atanan değişkeni ekrana yazdırma işlemleri işlenecektir.

Değişkene ilk değer tanımlanırken veya değer tanımladıktan sonra atanabilir.

```
int x = 5;  // ilk değer tanımlanırken atandı.

int y;
y = 3;      // ilk değeri tanımlandıktan sonra atandı.
```

Bu değişkenlerin ekrana nasıl yazdırılabileceği aşağıdaki örneklerde yer almaktadır.

Örnek 1.6-6:

```
staticvoid Main(string[] args)
        {
int x = 5;  // ilk değer tanımlanırken atandı.

int y;
            y = 3;      // ilk değer tanımlandıktan sonra atandı.

Console.WriteLine(x);
Console.WriteLine(y);

Console.ReadKey();
        }
```

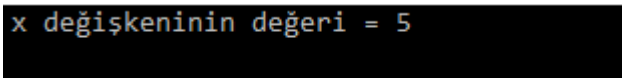
Çıktısı:

Fotoğraf 1.6:Örnek 1.6-6 ekran çıktısı

Örnek 1.6-7:

```
staticvoid Main(string[] args)
       {
int x = 5;  // ilk değer tanımlanırken atandı.

int y;
           y = 3;      // ilk değer tanımlandıktan sonra atandı.

Console.WriteLine("x değişkeninin değeri = "+ x);  // x'in değerini süslü parantez
içinde belirtilen sıfır (o) yazan yere yaz.

Console.ReadKey();
       }
```

Çıktısı:



Fotoğraf1.7: Örnek 1.6-7 ekran çıktısı

{0} anlamı: Virgülden sonra tanımlanan ilk değişken değer sıfırın yerine yazılır.

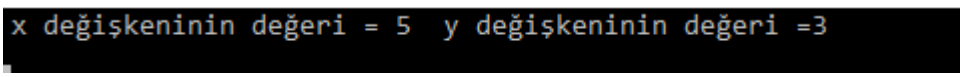
Aynıanda birden fazla değişken değeri ekrana yazdırılmak istenirse {0},{1},{2}… şeklinde devam eder.Değişkenler virgülle ayrılarak tanımlanır.

Örnek 1.6-8:

```
staticvoid Main(string[] args)
        {
int x = 5;  // ilk değer tanımlanırken atandı.

int y;
            y = 3;      // ilk değer tanımlandıktan sonra atandı.

Console.WriteLine("x değişkeninin değeri = {0}  y değişkeninin değeri
={1}", x, y);

Console.ReadKey();
        }
```

Bu örnekteki değişkenler farklı bir şekilde ekrana yazdırılmıştır. {0} ve {1} kodun sonundaki değişkenleri temsil etmektedir. Burada sıralama önemlidir.

Çıktısı:



Fotoğraf1.8:Örnek 1.6-8 ekran çıktısı

Değişken değerlerini ekrana birlikte yazdırmanın bir diğer yoluda + işaretini kullanmaktır. + işareti ifadeleri birleştirme işlemini gerçekleştirir.

Örnek 1.6-9:

```
staticvoid Main(string[] args)
        {
int x = 5;  // ilk değer tanımlanırken atandı.
int y;
            y = 3;      // ilk değer tanımlandıktan sonra atandı.

Console.WriteLine("x değişkeninin değeri = " + x);

Console.ReadKey();
        }
```

Çıktısı:

```
x değişkeninin değeri = 5
```

Fotoğraf 1.9:Örnek 1.6-9 ekran çıktısı

Görüldüğü üzere süslü parantez ile yapılan örnekle artı işareti kullanılarak yapılan örneğin çıktıları aynıdır.

Örnek 1.6-10:

```
staticvoid Main(string[] args)
        {
int x = 5;  // ilk değer tanımlanırken atandı.
int y;
            y = 3;      // ilk değer tanımlandıktan sonra atandı.

Console.WriteLine("x değişkeninin değeri = " + x + "
       y değişkeninin değeri =" + y);

Console.ReadKey();
        }
```

Çıktısı:

```
x değişkeninin değeri = 5  y değişkeninin değeri =3
```

Fotoğraf1.10: Örnek 1.6-10 ekran çıktısı

Örnek 1.6-11:

```
staticvoid Main(string[] args)
        {
string ad = "Murat";//İlk değer tanımlanırken atandı.
string soyad;
            soyad = "SEVCAN";   //İlk değer tanımladıktan sonra atandı.

Console.WriteLine(ad + " " + soyad);

Console.ReadKey();
        }
```

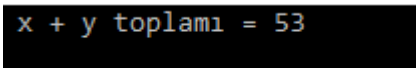
```
Murat SEVCAN
```

Fotoğraf 1.11: Örnek 1.6-11 ekran çıktısı

Örnek 1.6-12:

```
staticvoid Main(string[] args)
        {
int x = 5;  // ilk değer tanımlanırken atandı.
int y;
            y = 3;      // ilk değer tanımlandıktan sonra atandı.

Console.WriteLine("x + y toplamı = " + x + y);

Console.ReadKey();
        }
```

Çıktısı:



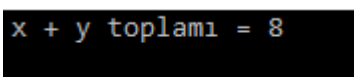
Fotoğraf 1.12:Örnek 1.6-12 ekran çıktısı

Yapılmak istenen işlem aslında x ve y değişken değerlerini toplayıp ekrana yazdırmaktır. Fakat + işaretinin buradaki görevi ifadeleri birleştirmek olduğundan x ve y değişken değerleri yan yana yazılarak yanlış sonuç üretilmiştir.

Örnek 1.6-13:

```
staticvoid Main(string[] args)
        {
int x = 5;  // ilk değer tanımlanırken atandı.
int y;
            y = 3;      // ilk değer tanımlandıktan sonra atandı.

Console.WriteLine("x + y toplamı = " + (x + y)); // x + y işlemi parantez
içine alınırsa + işareti toplama işlemi gerçekleştirilecektir.

Console.ReadKey();
        }
```

Çıktısı:



Fotoğraf 1.13:Örnek 1.6-13 ekran çıktısı

`Console.WriteLine("x + y toplamı = " + (x + y));` satırında x + y işlemini parantez içine alınca doğru sonuç elde edilmiştir.

Kod satırı`Console.WriteLine("x + y toplamı = {0}", x + y);` şeklinde de yazılsaydı aynı sonuç elde edilirdi.

1.6.3. Formatlı Çıkış İşlemleri

Tam sayı tipinde tanımlanmış değişkenler üzerinde uygulanabilecek format biçimleri aşağıdaki tablola belirtilmiştir.

Karakter	İsim	Örnek	Çıktı
C veya c	Currency	Console.Write("{0:C}", 2500);	2.500,00 TL
F veya f	Fixed-point	Console.Write("{0:F0}", 25); Console.Write("{0:F2}", 25); Console.Write("{0:F5}", 25);	25 25,00 25,00000
N veya n	Number	Console.Write("{0:N}", 2500000);	2.500.000,00

Tablo 1.3:Tam sayı tipindeki değişkenlere uygulanacak format listesi

Formatlı yazımda kullanılan parametrelerin açıklaması şöyledir:

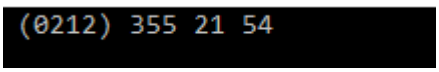
C: Sayıyı para birimi şeklinde gösterir.
D:Tek kullanıldığında bir anlam ifade etmez.Yanına sayı yazılarak kullanılır.Formatı alınacak sayının basamak değeri yanında yazılan sayıdan küçükse arta kalan değer kadar yanına sıfır eklenir.
E: Sayıyı 10 üzeri şeklinde gösterir.
F: Sayıların virgülden sonraki basamak sayısını ayarlamada kullanılır.
N: Sayıyı binlik basamaklara ayırarak yazar.
X: Sayıyı hexdecimal(16′ lık sayı sistemi) olarak yazar.

 # işareti: Formatlı yazımda her bir sayı için # işareti kullanılabilir.

Örnek 1.6-14:

```
staticvoid Main(string[] args)
        {

Console.Write("{0:(0###) ### ## ##}", 2123552154);

Console.ReadKey();
        }
```

Çıktısı:



Fotoğraf1.14:Örnek 1.6-14 ekran çıktısı

Tarih formatlama

Karakter	İsim	Örnek(DateTime.Now) için)
d	Kısa Tarih	01.04.2005
D	Uzun Tarih	01 Nisan 2005 Cuma
t	Kısa Saat	20:24
T	Uzun Saat	20:24:27
f	Tam Tarih ve Saat	01 Nisan 2005 Cuma 20:24
F	Tam Tarih ve Uzun Saat	01 Nisan 2005 Cuma 20:24:27
g	Varsayılan Tarih ve Saat	01.04.2005 20:24
G	Varsayılan Tarih ve Uzun Saat	01.04.2005 20:24:27
M	Gün ve Ay	01 Nisan
r	RFC1123 Tarih Metni	Fri, 01 Apr 2005 20:24:27 GMT
s	Sortable Tarih Metni	2005-04-01 20:24:27
u	Universal sortable, Yerel Zaman	2005-04-01 T20:24:27Z
U	Universal sortable, GMT	01 Nisan 2005 Cuma 17:24:27
	Ay ve Yıl	Nisan 2005

Tablo 1.4: Tarih tipindeki değişkenlere uygulanacak format listesi

Özel tarih formatlama

Karakter	İsmi	Örnek	Örnek Çıktısı
dd	Gün	`Console.Write("{0:dd}", DateTime.Now);`	01
ddd	Gün İsmi	`Console.Write("{0:ddd}", DateTime.Now);`	Cum
dddd	Tam Gün İsmi	`Console.Write("{0:dddd}", DateTime.Now);`	Cuma
hh	Saat	`Console.Write("{0:hh}", DateTime.Now);`	08
HH	Saat(24 Saat)	`Console.Write("{0:HH}", DateTime.Now);`	20
mm	Dakika 00-59	`Console.Write("{0:mm}", DateTime.Now);`	53
MM	Ay 01-12	`Console.Write("{0:MM}", DateTime.Now);`	04
MMM	Ay İsmi	`Console.Write("{0:MMM}", DateTime.Now);`	Nis
MMMM	Tam Ay İsmi	`Console.Write("{0:MMMM}", DateTime.Now);`	Nisan
yy	Yıl, Son İki Karakter	`Console.Write("{0:yy}", DateTime.Now);`	05
yyyy	Yıl	`Console.Write("{0:yyyy}", DateTime.Now);`	2005
:	Ayırıcı	`Console.Write("{0:hh:mm:ss}", DateTime.Now);`	08:53:56
/	Ayırıcı	`Console.Write("{0:dd/MM/yyyy}", DateTime.Now);`	01/04/2005

Tablo 1.5: Özel tarih format listesi

1.7. Giriş İşlemleri

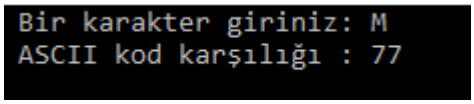
1.7.1. KlavyedenDeğişkene Değer Atama

Console.Read(): Console sınıfının Read() metodu kullanıcının klavyeden giriş yapmasını sağlar, tek karakter okur ve geriye tam sayı tipinde bir değer döndürür. Bu değer okunan karakterin 'ascii' kod karşılığıdır.

Örnek 1.7-1:

```
staticvoid Main(string[] args)
        {
int x;
Console.Write("Bir karakter giriniz: ");
            x = Console.Read();
Console.WriteLine("ASCII kod karşılığı : {0}", x);
Console.ReadKey();
        }
```

Çıktısı:



Fotoğraf1.15:Örnek 1.7-1 ekran çıktısı

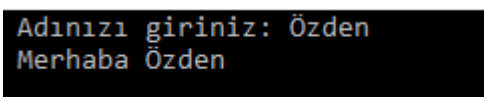
Read() metoduyla klavyeden istenen kadar değer okutulabilir ama geriye sadece ilk karakterin ascii kod karşılığı döndürülecektir. Burada dikkat edilmesi gereken bir başka nokta ise Read() metodu geriye tam sayı bir değer döndürdüğü için atamatam sayı tipinde bir değişkene yapılmalıdır. Aksi hâlde hata mesajı alırız(**Cannot implicitly convert type 'int' to 'string'**).

Console.ReadLine(): Console sınıfının ReadLine() metodu kullanıcının klavyeden bir değer girmesini sağlar ve bu değeri metin(string) bir ifade olarak geri döndürür.

Örnek 1.7-2:

```
staticvoid Main(string[] args)
        {
string x;
Console.Write("Adınızı giriniz: ");
            x = Console.ReadLine();
Console.WriteLine("Merhaba " + x);

Console.ReadKey();
        }
```

Çıktısı:



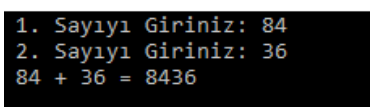
Fotoğraf 1.16:Örnek 1.7-2 ekran çıktısı

ReadLine() metodu ile geri döndürülen değer metin(string) tipindedir. Dolayısıyla okunan değer metin(string) tipinde bir değişkene atanmalıdır. Matematiksel bir işlem yapılacaksa değeri sayısal ifadeye çevirmek gerekir. Bu işlem için 'Convert' sınıfı veya veri türlerinin 'Parse' özelliğinden faydalınır.

Örnek 1.7-3:

```
staticvoid Main(string[] args)
        {
            string x, y;
Console.Write(" 1. Sayıyı Giriniz: ");
            x = Console.ReadLine();
Console.Write(" 2. Sayıyı Giriniz: ");
            y = Console.ReadLine();
Console.WriteLine(" "+x+" + "+y+" = "+ x + y);

Console.ReadKey();
}
```

Çıktısı:



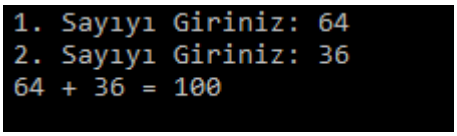
Fotoğraf 1.17: Örnek 1.7-3 ekran çıktısı

Görüldüğü üzere string ifadeler üzerinde matematiksel işlemler yapılamıyor. Matematiksel işlem yapılacaksa değişkenin tipi sayısal ifadeye çevrilmelidir.

Örnek 1.7-4:

```
staticvoid Main(string[] args)
        {
                    int x, y, toplam;
      Console.Write("1. Sayıyı Giriniz: ");
                  x = Convert.ToInt16(Console.ReadLine());
      Console.Write("2. Sayıyı Giriniz: ");
                  y = Convert.ToInt16(Console.ReadLine());
                  toplam = x+y;
      Console.WriteLine(" "+x+" + "+y+" = " +(x+y));

Console.ReadKey();
}
```

Çıktısı:



Fotoğraf 1.18:Örnek 1.7-4 ekran çıktısı

int türünde bir değişken için değer okunacaksa bu değer okuma yukarıdaki örnekte olduğu gibi Convert.ToInt16(Console.ReadLine()) kullanmak gerekir. Bu satır okunan değeri int türüne çevirecektir.

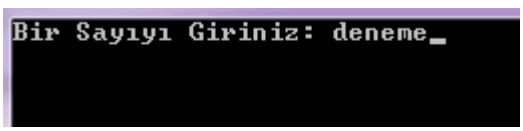
1.8. Giriş-Çıkış İşlemleri Hata Mesajları

Hazırlanacak programın en önemli özelliklerinden biri de stabil çalışmasıdır. Stabil çalışması programın hatalara karşı ne kadar hazırlıklı olduğuyla ve kullanıcıya verdiği geri dönütle eşdeğerdir. Programın çalışması sırasında oluşabilecek hatalar genellikle kullanıcı girişlerinden kaynaklanır. Bu yüzden kullanıcı girişlerini kontol altına alarak çalışma zamanında oluşabilecek hataları en aza indirmek programcıların görevidir.Hataoluştuğunda önce programın nasıl sonlandığı görülüp daha sonra bunun için bir çözüm aranmalıdır.

Örnek 1.8-1:

```
staticvoid Main(string[] args)
        {
int x;
Console.Write("Bir Sayıyı Giriniz: ");
            x = Convert.ToInt16(Console.ReadLine());


Console.ReadKey();
        }
```

Programı çalıştıralım ve sayı yerine string bir ifade girelim.



Fotoğraf 1.19: Örnek 1.8-1 ekran çıktısı

Programda istenen sayı yerine ‘string’ bir ifade girildiğinde çıktısı aşağıdaki gibi olacaktır. ‘String’ ifadeyi‘int’ tipine çevirmede zorlanılacağından program duracaktır. Giriş dizesinin doğru olmadığına dair bir hata verecektir**(Input string was not in a corrent format.)**.

```
static void Main(string[] args)
{
    int x;
    Console.Write("Bir Sayıyı Giriniz: ");
    x = Convert.ToInt16(Console.ReadLine());

    Console.ReadKey();
}
```

FormatException was unhandled

Input string was not in a correct format.

Troubleshooting tips:

Fotoğraf1.20: Giriş dizesi doğru biçimde değil hata mesajı

Örnek 1.8-2:

```
staticvoid Main(string[] args)
        {
int x, y;
Console.Write("1. Sayıyı Giriniz: ");
            x = Convert.ToInt16(Console.ReadLine());

Console.Write("2. Sayıyı Giriniz: ");
            y = Convert.ToInt16(Console.ReadLine());

Console.WriteLine("{0} / {1} = {2}", x, y, x / y);

Console.ReadKey();
        }
```

Program çalıştırılıp aşağıdaki gibi değerler girilmelidir.

```
1. Sayıyı Giriniz: 6
2. Sayıyı Giriniz: 0
```

Fotoğraf1.21: Örnek 1.8-2 ekran çıktısı

Girilen sayılardan biri sıfır olduğu zaman program sıfıra bölme hatasıyla karşılaşacağından dolayı duracaktır (**Attempted to divide by zero**).

```
static void Main(string[] args)
{
    int x, y;
    Console.Write("1. Sayıyı Giriniz: ");
    x = Convert.ToInt16(Console.ReadLine());

    Console.Write("2. Sayıyı Giriniz: ");
    y = Convert.ToInt16(Console.ReadLine());

    Console.WriteLine("{0} / {1} = {2}", x, y, x / y);

    Console.ReadKey();
}
```

DivideByZeroException was unhandled

Attempted to divide by zero.

Fotoğraf1.22: Sıfıra bölme hatası mesajı

Yukardaki kodlarda da görüldüğü gibi program esnasında kullanıcılar tarafından yapılan hatalı girişler programın hatayla karşılaşmasına ve durmasına sebep olmaktadır. Bu hatalar, program içinde yakalanıp kullanıcıya hata hakkında nasıl bir ileti (mesaj) verileceği aşağıdaki bölümlerde yer almaktadır.

.Net programcılıkta bu tür hatalara istisnalar (Exception) denir. İstisnalar, programın çalışma zamanında yani program çalışırken ortaya çıkan olağan dışı durumlardır..NET ortamında her şey gibi istisnalarda sınıflar kullanılarak oluşturulmaktave tüm istisnalar temel System.Exception nesnesinden türetilmektedir.

İstisnaları program esnasında yakalamak ve kullanıcıya hata mesajını vermek için try{} catch{} finally{} blokları kullanılır.

try{} bloku: İstisnanın çıkması muhtemel kodların yazıldığı bloktur.

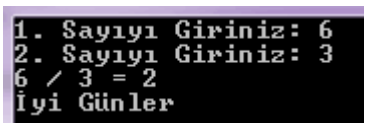
catch{} bloku: Oluşan istisnanın yakalandığı ve kullanıcıya sunulduğu bloktur.

finally{} bloku: Try bloku içinde hata olsa da olmasa da çalışması istenenkodların yazıldığı bloktur. Finally bloğu genellikle bazı kaynakları serbest bırakmak için kullanılır. Kullanılması isteğe bağlı bir bloktur. En sık kullanıldığı yerler açık olan veritabanı bağlantılarının program kırılsada kırılmasada kapatılması durumlarıdır.

Örnek 1.8-3:

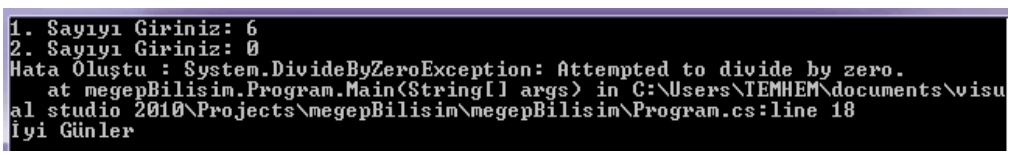
```
staticvoid Main(string[] args)
        {
int x, y;
Console.Write("1. Sayıyı Giriniz: ");
            x = Convert.ToInt16(Console.ReadLine());
Console.Write("2. Sayıyı Giriniz: ");
            y = Convert.ToInt16(Console.ReadLine());
try
            {
Console.WriteLine("{0} / {1} = {2}", x, y, x / y);
            }
catch (Exception e)
            {
Console.WriteLine("Hata Oluştu : {0}", e);
            }
finally
            {
Console.WriteLine("İyi Günler");
            }
Console.ReadKey();
        }
```

Program çalıştırılıp aşağıdaki gibi değerler girildiği zaman herhangi bir hatayla karşılaşılmadığından istenen sonucu üretecektir.



Fotoğraf1.23:Örnek 1.8-3 ekran çıktısı

Fakat prgorama aşağıdaki değerler girildiği zaman hatayla karşılaşılacak ve program olduğu yerde durup catch{} blokuna atlayacak ve buradan çalışmaya devam edecektir.

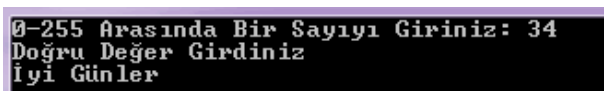


Fotoğraf1.24:Örnek 1.8-3 ekran çıktısı

Dikkat edilirse her iki durumda da finally bloku içine yazılan kodlar çalıştırılmaktadır.

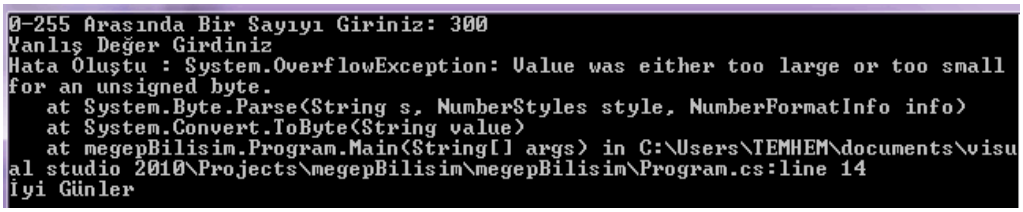
Örnek 1.8-4:

```
staticvoid Main(string[] args)
        {
byte x ;

try
            {
Console.Write("0-255 Arasında Bir Sayıyı Giriniz: ");
                x = Convert.ToByte(Console.ReadLine());

Console.WriteLine("Doğru Değer Girdiniz");
            }
catch (Exception e)
            {
Console.WriteLine("Yanlış Değer Girdiniz");
Console.WriteLine("Hata Oluştu : {0}", e);
            }
finally
            {
Console.WriteLine("İyi Günler");
            }

Console.ReadKey();
        }
```



Fotoğraf 1.25:Örnek 1.8-4 ekran çıktısı

Kullanıcıdan istenen 0-255 arası bir değer girilirse program herhangi bir hatayla karşılaşmayacağı için try bloku içindeki tüm kodlar icra edilip finally bloğuna atlar ve çalışmasına ordan devam eder (yukardaki ekran çıktısında görüldüğü gibi).



Fotoğraf1.26:Örnek 1.8-4 ekran çıktısı

Fakat istenen değer dışında bir değer girildiği zaman program hatayla karşılaşacağından hatanın oluştuğu satırda program durdurulur, catch blokuna atlanır ve çalışmasına ordan devam eder. Yukardaki ekran çıktısında görüldüğü üzere hata `x = Convert.ToByte(Console.ReadLine());` satırında oluştuğundan bir sonraki satır icra edilmeden catch blokuna atlanmış ve programa buradan devam edilmiştir.

1.9. Açıklama Satırları

Açıklama satırları programcıya kod içinde tanımlama metinleri yazma imkânı sağlar.Bu sayede kod parçacıklarının ne iş yaptıkları anlatılmış olur. Kodlar arasına açıklama satırları eklemek oldukça önemlidir. Az satırlı program kodlarında birşey ifade etmeyebilir fakat büyük programlarda kod bloklarının ne işe yaradıkları yazılarak programcının ileride karşılaşacağı problemleri kolay çözmesinde yardımcı olacaktır. Ayrıca açıklama satırları program derlenirken dosya içine alınmadığından oluşan dosyanın boyutunu yada çalışmasını etkilememektedir.

```
staticvoidMain(string[] args)
{
//Bu Satır ekrana Merhaba Dünya Yazar.
Console.Write("Merhaba Dünya");
}
```

// karakterlerinden sonra gelen ve satırın sonuna kadar olan sözcükler yorum satırlarıdır ve programlama dili derleyicisi tarafından görülmez. Aynı zamanda birden fazla satıra yorum eklenmek isteniyorsa /* */ karakterleri arasına yorum yazılır.

```
staticvoidMain(string[] args)
{
/* Bu Satır ekrana Merhaba Dünya Yazar.
Bu Kısım derleyici tarafından yok sayılır.
*/
Console.Write("Merhaba Dünya");
}
```

DEĞERLER ETKİNLİĞİ-1

İŞ AHLAKI

Yabancı bir kumaş taciri Osmanlı ülkesine gelerek bir kumaş imalathanesinin mallarını beğenip hepsini almak istedi. Ancak mal sahibinin kumaş toplarını denklerken bir top kumaşı ayırdığını görüp bu hareketinin sebebini sordu. Osmanlı esnafı "Onu sana veremem, kusurludur." cevabını verdi. Yabancı tacir "Ziyanı yok, önemli değil." dedi. Buna rağmen Osmanlı esnafı o kumaş topunu vermemekte direterek "Ben malımın kusurlu olduğunu söyledim biliyorsunuz fakat siz onu kendi memleketinizde satarken alıcılarınız orada benim bunları size söylediğimi bilmeyeceklerdir. Böylece de müşterilerinize kusurlu mal satmış olacağım. Neticede Osmanlının gururu, şeref ve haysiyeti rencide olacak. Bizi de hilekâr sanacaklardır. Onun için bu kusurlu kumaş topunu asla size veremem." diyerek kumaşı vermeyişinin sebebini açıkladı.

Böyle bir durumda siz ne yapardınız?

.....
.....

İş ahlakına önem veren tüccarla ilgili düşüncelerinizi yazınız.

.....
.....

Bu iki kişinin davranışları sizin iş ahlakına bakışınızı nasıl etkiledi? Kısaca açıklayınız.

……
…….....
.....
.....
.....
.....

UYGULAMA FAALİYETİ

Aşağıdaki uygulamaları yapınız.

İşlem Basamakları	Öneriler
➢ İş sağlığı ve güvenliği tedbirlerini alınız. ➢ Bir okul programında kullanılabilecek değişkenleri maddeler hâlinde yazınız.	➢ Değişken isimlendirme kurallarına dikkat etmelisiniz.
➢ Tanımladığınız her değişkene bir tip belirleyiniz.	➢ He değişkenin bir tipi olması gerektiğini unutmamalısınız.
➢ Tanımladığınız her değişkene bir değer atayınız.	➢ Tanımlanan her yerel değişkenin bir değeri olması gerektiğini unutmayınız.
➢ Tanımladığımız değişkenlerden sabit olarak ayırabileceklerimizi seçiniz.	➢ Sabitlerin değeri tanımlandıklarında atanır ve program boyunca değiştirilemez.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatle okuyunuz ve doğru seçeneği işaretleyiniz.

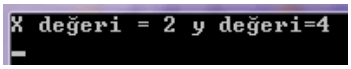
1. Aşağıdaki değişken isimlerinden hangisi hatalıdır?
A) ogrenciNo
B) OgrenciAd
C) ogrenci_soyad
D) ogrenci sinif
E) Ogrenci_1

2. Aşağıdakilerden hangisi veri tiplerinden değildir?
A) string
B) slong
C) int
D) byte
E) double

3. 8 bit işaretsiz tam sayı tipi aşağıdakilerden hangisidir?
A) Byte
B) int
C) long
D) short
E) string

4. Program boyunca sabit kalacak veri hangi kelime ile tanımlanır?
A) Float
B) double
C) bool
D) const
E) try

5. Aşağıdaki kullanımlardan hangisi ile önce arttırma işlemi yapılıp daha sonra atama işlemi yapılır?
A) +a+
B) +a+
C) +a
D) a+
E) ++a

6. Aşağıdaki yazımlardan hangisinin ekran çıktısı 

şeklindedir.

A) `Console.WriteLine("X değeri = x y değeri=y");`
B) `Console.WriteLine("X değeri = {0} y değeri={1}" + 2 + 4);`
C) `Console.WriteLine("X değeri = {0} y değeri={0}", 2, 4);`
D) `Console.Write("X değeri = {0} y değeri={1}", 2, 4);`
E) `Console.WriteLine("X değeri = {0} y değeri={1}", 2, 4);`

7. Hangi tarih formatı kullanılırsa ekran çıktısı "Nisan" şeklinde olur?

A) `Console.Write("{0:MMM}", DateTime.Now);`
B) `Console.Write("{0:MMMM}", DateTime.Now);`
C) `Console.Write("{0:yy}", DateTime.Now);`
D) `Console.Write("{0:dddd}", DateTime.Now);`
E) `Console.Write("{0:dd}", DateTime.Now);`

Aşağıda verilen cümlelerde boş bırakılan yerlere doğru sözcükleri yazınız.

8. Bir metin ifadesini ekrana yazdırmak için ..... veya ..... metodu kullanılır.

9. Metin bir alt satırdan başlanarak yazdırmak için ….. parametresi kullanılır.

10. Muhtemel istisna oluşması düşünülen kodlar ..... blokları arasına alınır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ–2

ÖĞRENME KAZANIMI

Verilen problemdeki işlemlere uygun operatörleri kullanabileceksiniz.

ARAŞTIRMA

- Matematik dersinde kullanılan işlem önceliği kurallarını liste hâlinde hazırlayınız.
- Kendinize bir hedef belirleyiniz. Bu hedefe ulaşmak için gerekli olan tüm şartları sıralayınız. Hangi şartları gerçekleştirdiğinizde hedefinize ulaşabileceğinizi işaretleyiniz.

2. OPERATÖRLER

Programlama dillerinde tanımlanmış sabit ve değişkenler üzerinde işlemler yapmayı sağlayan karakter ya da karakter topluluklarına **operatör**denir.

Örneğin;

```
int sayi = 2 + 3;
```

Yukardaki örnekte + ve = karakterleri birer operatördür. + karakteri 2 ve 3 sabitlerini toplama yapıyor ve = karakteri ise toplanan değeri tanımlanan değişkene atama işlemini gerçekleştiriyor.

2.1. Aritmetiksel Operatörler

Aritmetik işlemler yapılırken kullanılan operatörlerdir.

2.1.1. Dört İşlem

Operatör	Açıklama
+	Toplama İşlemi
-	Çıkarma İşlemi
*	Çarpma İşlemi
/	Bölme İşlemi

Tablo 2.1: Dört işlem operatör listesi

Örnek 2.1-1:

```
staticvoid Main(string[] args)
        {
int x = 10;
int y = 5;

Console.WriteLine("x + y = " + (x + y));
Console.WriteLine("x - y = " + (x - y));

Console.WriteLine("x * y = " + (x * y));
Console.WriteLine("x / y = " + (x / y));

Console.ReadKey();        }
    }
```

Çıktısı:

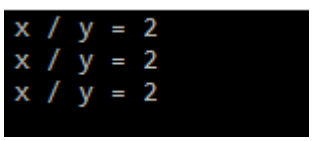
```
x + y = 15
x - y = 5
x * y = 50
x / y = 2
```

Fotoğraf2.1: Örnek 2.1-1 ekran çıktısı

Örnek 2.1-2:

```
staticvoid Main(string[] args)
{
int x = 10;
int y = 4;

int intSonuc = x / y;
float floatSonuc = x / y;
double doubleSonuc = x / y;

Console.WriteLine("x / y = "+ intSonuc);
Console.WriteLine("x / y = " + floatSonuc);
Console.WriteLine("x / y = " + doubleSonuc);

Console.ReadKey();
}
```

Çıktısı:



Fotoğraf2.2: Örnek 2.1-2 ekran çıktısı

Aslında sonuç 2.5 olması gerekirken 2 olarak çıktı. Bunun sebebi ise x ve y değişkenleri int yani tam sayı tipinde tanımlandıklarından çıkan sonucun da kesirli kısmı atılıp tam sayı kısmı alınmaktadır.

Örnek 2.1-3:

```
      staticvoid Main(string[] args)
      {
int x = 10;
int y = 4;

int sonuc;
float floatSonuc;

            sonuc = x / y;
Console.WriteLine("x / y = " + sonuc);

            floatSonuc = (float)x / y;
Console.WriteLine("x / y = " + floatSonuc);

            floatSonuc = x / (float)y;
Console.WriteLine("x / y = " + floatSonuc);

            floatSonuc = (float)x / (float)y;
Console.WriteLine("x / y = " + floatSonuc);

            floatSonuc = (float)(x / y);
Console.WriteLine("x / y = " + floatSonuc);

Console.ReadKey();
      }
```

Çıktısı:

```
x / y = 2
x / y = 2,5
x / y = 2,5
x / y = 2,5
x / y = 2
```

Fotoğraf2.3: Örnek 2.1-3 ekran çıktısı

Birinci çıktı: sonuc = x / y = 10/4 bir tamsayı bölme işlemi olduğundan bölümün tam sayı kısmı alınmıştır.

İkinci çıktı: floatSonuc = (float)x / y = (float)10 / 4 deyiminde, önce 10 sayısı float tipine dönüştürülüyor, sonra 4 sayısına bölünüyor. Bir float tipin bir tam sayıya bölümü yine float tipindendir. Dolayısıyla (float)10 / 4 = 2.5'tir.

Üçüncü çıktı: Bu çıktı ikinci çıktının simetriğidir. floatSonuc = x / (float)y = 10/ (float)4 deyimindeönce 4 sayısı float tipine dönüştürülüyor.Sonra 10 tam sayısı4.0 sayısına bölünüyor. Bir tam sayı tipin bir float tipine bölümü yine float tipindendir. Dolayısıyla 10 / (float)4 = 2.5'tir.

Dördüncü çıktı:İkinci ve üçüncüçıktının birleşimidir. floatSonuc= (float)x / (float)y = (float)10 / (float)4 deyiminde önce 10 ve 4 sayılarının her ikisi de float tipine donüştürülür. Sonra iki float tipin birbirine bölümü yapılır. Bu işlemin sonucu doğal olarak bir float tipidir. Dolayısıyla (float)10 / (float)4 = 2.5'tir.

Beşinci çıktı: floatSonuc = (float)(x / y) = (float)(10/4) deyiminde, önce (10 / 4) bölme işlemi yapılır. Bu bir tam sayı bölme işlemi olduğu için birinci çıktıda olduğu gibi çıkan sonuç 2'dir. (float) 2 = 2.0000000 olduğundançıktı 2'dir.

2.1.2. Mod Alma

Bir sayının başka bir sayıya bölümünden kalan sonucu alma işlemine **mod alma** denir. Bu işlemi yapmak için % karakteri kullanılır.

Örnek 2.1-4:

```
            staticvoid Main(string[] args)
                    {
double x = 10;
            double y = 4;

            double sonuc = x % y;

            Console.WriteLine("x = " + sonuc + "(mod "+ y +")");

            Console.ReadKey();
            }
```

Çıktısı :

```
x = 2(mod 4)
```

Fotoğraf 2.4 örnek çıktısı

Örnek 2.1-5:

```
staticvoid Main(string[] args)
        {
           int x, y;

Console.Write(" 1. Değeri Giriniz :");
            x = Convert.ToInt16(Console.ReadLine());
Console.Write(" 2. Değeri Giriniz :");
            y = Convert.ToInt16(Console.ReadLine());

Console.WriteLine(x +" = "+ x%y + " (mod "+y+") ");

Console.ReadKey();
}
```

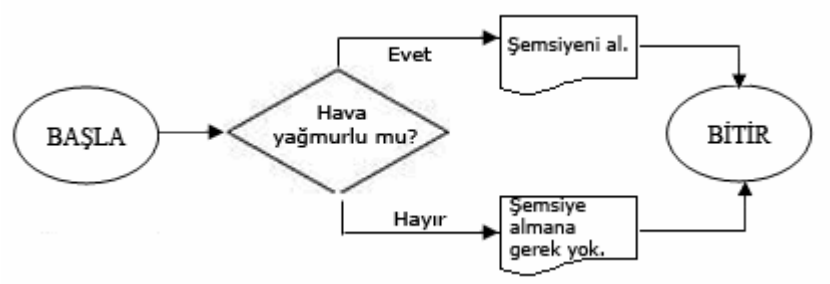
Çıktısı :

```
1. Değeri Giriniz :36
2. Değeri Giriniz :8
36 = 4 (mod 8)
```

Fotoğraf2.4: Örnek 2.1-5 ekran çıktısı

2.2. İlişkisel Operatörler

İlişkisel operatöler iki değerin karşılaştırılması işlemi için kullanılır. Programda koşul ifadelerinde kullanılarak programın akışını değiştirmeyi sağlar. Karşılaştırma sonucunda **doğru(true)** ve **yanlış(false)** olmak üzere **boolen** bir değer döndürür. Bu tür operatörler kullanılırken düz bir şekilde okunursa içerik daha iyi anlaşılır.



Şema 2.1: İki değerin karşılaştırılması

Hava yağmurlumu? sorusu bir şarttır ve bu şart içinde havanın durumu yağmurla karşılaştırılıyor. Bu soruya dışarıda yağmur yağıyorsa evet(true) yağmıyorsa hayır(false) şeklinde boolen bir cevap veriliyor. Dolayısıyla verilen cevaba görede programın akışı yön değiştirmektedir.

Operatör	Açıklama
==	Eşittir
!=	Eşit Değildir
>	Büyüktür
<	Küçüktür
>=	Büyük Eşittir
<=	Küçük Eşittir

Tablo 2.2: İlişkisel operatör listesi

== Operatörü: Aynı türdeki iki değerin birbirine eşitliğini karşılaştırmak için kullanılan operatördür.

```
int x = 10;
int y = 4;

string str1 = "megep";
string str2 = "megep";

x == y //false// x içeriği y içeriğine eşit mi?
str1 == str2 // true// str1 içeriği str2 içeriğine eşit mi?

3 == 3 // true
3 == "3"// hatalı kullanım. int tipi ile string tipi karşılaştırılamaz.
```

Tek eşittir (=) değişkene atama yaparken çift eşittir (==) karşılaştırmak için kullanılmaktadır.

!= Operatörü: Aynı türdeki iki değerin birbirine eşit olmadığının(eşit değil) kontolü için kullanılan operatördür.

```
int x = 10;
int y = 4;

string str1 = "megep";
string str2 = "megep";

x != y // true // x içeriği y içeriğinden farklı mı?
str1 != str2 // false //str1 içeriği str2 içeriğinden farklı mı?

3 != 3 // false
1 != 3 // true
```

>Operatörü: Bir değerin aynı türdeki başka bir değerden büyüklüğünün kontrolünün yapıldığı operatördür.Bu operatör string işlemlere uygulanmaz.

```
int x = 10;
int y = 4;

x > y // true // x y'den büyük mü?
y > x // false // y x'den büyük mü?

3 > 3 // false
5 > 3 // true
```

<Operatörü: Bir değerin aynı türdeki başka bir değerden küçüklüğünün kontrolünün yapıldığı operatördür. Bu operatör string işlemlere uygulanmaz.

```
int x = 10;
int y = 4;

x < y // false // x y'den küçük mü?
y < x // true // y x'den küçük mü?

3 < 3 // false
1 < 3 // true
```

>= Operatörü: Bir değerin aynı türdeki başka bir değerden büyük veya eşitliği kontrolünün yapıldığı operatördür.Bu operatör string işlemlere uygulanmaz.

```
int x = 10;
int y = 4;

x >= y // true // x y'den büyük mü yada eşit mi?
y >= x // false // y x'den büyük mü yada eşit mi?

3 >= 3 // true
```

<= Operatörü: Bir değerin aynı türdeki başka bir değerden küçük veya eşitliği kontrolünün yapıldığı operatördür. Bu operatör string işlemlere uygulanmaz.

```
int x = 10;
int y = 4;

x <= y // false // x y'den küçük mü yada eşit mi?
y <= x // true // y x'den küçük mü yada eşit mi?

3 <= 3 // true
```

2.3. Mantıksal Operatörler

Mantıksal operatörler birden fazla şartın olduğu durumlarda kullanılır. Birden çok boolen değeri tek bir boolen değere indirmek için kullanılır.

Operatör	Açıklama
&&	Ve
‖	Veya
!	Değil

Tablo 2.3:Mantıksal operatör listesi

&&Operatörü:‘Ve’ anlamındadır.Sorgulanan tüm şartlar doğru(true) olduğu zaman doğru(true), şartlardan birinin yanlış(false) olması durumunda yanlış(false) değerini döndürür. Yani şartların içinde bir tane **false** değeri varsa diğer şartların bir anlamı yoktur.

```
int x = 10, y = 4;
string str1 = "megep";

x == y &&"megep" = str1 // x y’ye eşit mive megep kelimesi str1’e eşit
mi? Birinci şartımız false olduğu için sonuç false çıkacaktır.

x == 10 && y == 4 &&"megep" = str1 // x 10’a eşit mi ve y 4’e eşit mive
megep kelimesi str1’e eşit mi? Bütün şartlar true olduğu için sonuç
true’dur.
```

‖Operatörü:‘Veya’ anlamındadır.Sorgulanan şartlardan birinin **doğru(true)** olması durumunda doğru(true), şartların hepsinin yanlış(false) olması durumunda yanlış(false) değerini döndürür.Yani şartların içinde bir tane **true** değeri varsa diğer şartların bir anlamı yoktur.

```
int x = 10, y = 4;
string str1 = "megep";

x == y ||"megep" == str1 // x y’ye eşit mi veya megep kelimesi str1
değişkenine eşit mi? İkinci şartımız true olduğu için sonuç true’dur.

x == 4 || y == 10 ||"megep" == str1 // x değişkeni 4’e eşit mi veya y
değişkeni 10’a eşit mi veya megep kelimesi str1 değişkenine eşit mi? İlk
iki sonucumuz false olsa da üçüncü modülümüz true olduğu için sonuç
true’dur.

x == 4 || y == 10 ||"megep1" == str1 // x değeri 4’e eşit mi veya y
değeri 10’a eşit mi veya megep1 kelimesi str1 değişkenine eşit mi? 3
şartımız da false olduğu için sonuç false’dur.
```

! Operatörü:‘Değil’ anlamındadır. ! işareti değeri tersine çevirir.

(!true) `// sonuç = false`
(!false) `// sonuç = true`

2.4. İşlem Önceliği

İşlem öncelik sırası aşağıdaki tabloda en yüksekten en düşüğe doğru sıralanmıştır.

En Yüksek
()
!
*, /, %
+, -
<, >
==, !=
&&
||
En Düşük

Tablo 2.4:İşlem önceliği listesi

Yapılan işlemde yukarıdaki sıra tamamlandıktan sonra eğer aynı tür işlemler kaldıysa işlem soldan sağa doğru yapılır.

Örnek 2.4-1: 3+5*2 işleminin sonucu nedir?

Yukardaki işlemde önce 3 ile 5 toplanır ve sonuç 2 ile çarpılırsa sonuç yanlış çıkar. İşlem önceliğine göre önce 2 ile 5’i çarpıp çıkan sonuçla 3 toplandığı zaman sonuç doğru çıkar.

3+5*2 = 8*2 = 16 // yanlış cevap
3+5*2 = 3 + 10 = 13 // doğru cevap

Örnek 2.4-2:10/5*2 işleminin sonucu nedir?

Yukardaki işlemde önce 5 ile 2’yi çarpar ve 10 çıkan sonuca bölünürse yanlış cevap çıkar. Bölme ve çarpmanın işlem önceliği eş değer olduğu için işlem soldan sağa doğru işleyecektir.

10/5*2 = 10 / 10 = 1 // yanlış cevap
10/5*2 = 2 * 2 = 4 // doğru cevap

Örnek 2.4-3: (5+2)*4-6/2 işleminin sonucu nedir?

(5+2)*4-6/2 = 7*4-6/2 = 28-6/2 = 28-3 = 25 // doğru cevap

DEĞERLER ETKİNLİĞİ-2

Sabah erken saatlerde yanına aldığı veziriyle çarşıda olan Sultan Fatih, girdiği ilk dükkândan birkaç şey almak ister. Dükkân sahibi kendisini tanımamakla beraber, arzu ettiği şeylerden sadece birini hazırlayıp verir. Bunun üzerine Sultan diğer istediği şeylerin de hazırlanmasını söyler. Dükkân sahibi "Efendim ben sabah siftahımı yaptım, komşum da dükkânını yeni açtı. Diğer isteklerinizi ondan alınız." der. Sultan, yan dükkâna girer, bu sefer de yeni girdiği dükkânın sahibi istediklerinden yine sadece birini hazırlar ve yan dükkâna gitmesini, çünkü komşusunun bu sabah siftah yapmadığını, diğer alacaklarını da ondan almasını ister. Bu durumbir süre böyle devam eder.

Bu hikâyeye uygun başlık oluşturunuz.

……

Siz esnaf olsanız komşu esnafınız gelen müşteriyi size yönlendirse ne hissedersiniz?

.....
.....
.....

Siz müşterinizi siftah yapmamış komşu esnafa yönlendirir misiniz?

.....
.....
…….....
.....

UYGULAMA FAALİYETİ

Aşağıdaki işlemleri uygulayınız.

İşlem Basamakları	Öneriler
➢ Tanımlanan değişkenlere sayısal hesaplamalar yaparak değer aktarınız.	➢ İş sağlığı ve güvenlik tedbirlerini alarak önce basit matematiksel işlemleri denemelisiniz(yaşınızı hesaplamak gibi)
➢ Elde edilen değeri ekranda gösteriniz.	➢ Sadece sonucu göstermek anlamlı olmaz. Mesela: "Yaşım: 16" gibi yazdırmalısınız.
➢ Karmaşık matematiksel formüllerde gerekli yerlere parantez ekleyerek işlem önceliklerini belirleyiniz.	➢ Deneyeceğiniz formülleri hesap makinesi ve hesap tablosu programları ile deneyip sonucu karşılaştırmalısınız. ➢ Güvenilir sonuçlar elde etmeye çalışmalısınız.
➢ Metin birleştirme operatörü ile birden fazla metin değeri birleştiriniz.	➢ Birden fazla metnin değerini birleştirmelisiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki soruları dikkatle okuyunuz ve doğru seçeneği işaretleyiniz.

1. Mod alma işlemini aşağıdaki karakterlerden hangisi gerçekleştirir?
A) >
B) !
C) %
D) &
E) ||

2. Aşağıdakilerden hangisi ilişikisel operatörler arasında yer almaz?
A) ||
B) <=
C) !=
D) >
E) >=

3. Ve mantıksal operatörü aşağıdakilerden hangisidir?
A) !=
B) ==
C) &
D) ||
E) &&

4. (5+3)*4-6/2*3-(2*3) işleminin sonucu aşağıdakilerden hangisidir?
A) 33
B) 17
C) 25
D) 19
E) 22

Aşağıda verilen cümlelerde boş bırakılan yerlere doğru sözcüğü yazınız.

5. 4 > 2 işlemi ..... sonucunu üretir.

6. Koşul ifadelerinin karşılaştırılmasında kullanılan operatörlere ..... denir.

7. ……. operatörü 'değil' anlamındadır. Değeri tersine çevirir.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise **Modül Değerlendirme**ye geçiniz.

MODÜL DEĞERLENDİRME

Aşağıdaki soruları dikkatle okuyunuz ve doğru seçeneği işaretleyiniz.

1. Aşağıdaki yerel değişken tanımlamalarından hangisi doğru yapılmıştır?
A) int16 sayi
B) metin ad
C) byte yas
D) float 12
E) string 1.sayi

2. Aşağıdakilerden hangisi referans tipler arasında yer alır?
A) String
B) float
C) double
D) bool
E) int

3. Aşağıdaki atama operatörlerinden hangisi doğru kullanılmıştır?
A) a= = 5
B) a+-
C) a++= 2
D) a+= 3
E) a+>7

4. Aşağıdaki yazımlardan hangisi doğrudur?
A) Console.WriteLine("Boyunuz : {0} Kilonuz : {1}", boy, kilo);
B) Console.WriteLine("Boyunuz : boy Kilonuz : kilo");
C) Console.WriteLine("Boyunuz : [0] Kilonuz : [1]", boy, kilo);
D) Console.WriteLine("Boyunuz : {0} Kilonuz : {1} " + boy + kilo);
E) Console.WriteLine("Boyunuz : {0} Kilonuz : {0}", boy, kilo);

5. Console.Write("{0:C}", 3000); komutunun ekran çıktısı aşağıdakilerden hangisidir?
A) 3.000 TL
B) 3000 TL
C) 3.000,00 TL
D) 0003 TL
E) 0,3000 TL

6. 10 % 3 işleminin sonucu kaçtır?
A) 4
B) 1
C) 3
D) 2
E) 5

7. (8/2*4)+4-10*2 işleminin sonucu kaçtır?
A) 0
B) -15
C) -4
D) 15
E) 6

Aşağıdaki cümlelerin başında boş bırakılan parantezlere, cümlelerde verilen bilgiler doğru ise D, yanlış ise Y yazınız.

8. ( ) Tanımlanan bir değişkene, ancak tanımlandığı blok içinde ulaşılabilir.

9. ( ) ogrNo ile OgrNO aynı hafıza bölgesini temsil eder.

10. ( ) Birden fazla şartın olduğu durumlarda aritmetiksel operatörler kullanılır.

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki bireysel öğrenme materyaline geçmek için öğretmeninize başvurunuz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ 1'İNCEVAP ANAHTARI

1	**D**
2	**B**
3	**A**
4	**D**
5	**E**
6	**E**
7	**B**
8	**Console.Write()** **Console.WriteLine()**
9	**\n**
10	**Try ..Catch**

ÖĞRENME FAALİYETİ 2'NİN CEVAP ANAHTARI

1	**C**
2	**A**
3	**E**
4	**B**
5	**true**
6	**İlişkisel**
7	**!**

MODÜL DEĞERLENDİRME'NİN CEVAP ANAHTARI

1	**C**
2	**A**
3	**D**
4	**A**
5	**C**
6	**B**
7	**A**
8	**DOĞRU**
9	**YANLIŞ**
10	**YANLIŞ**

KAYNAKÇA

- **http://www.eba.gov.tr (04.12.2017 15:00)**