tekbilmyo.harran.edu.tr/tr/personel/dersler/

Programlamaya Giriş

Bir problemin bilgisayar ortamında bir programlama dili kullanılarak çözümünün yapılmasına programlama denir.

Programlama yapmak için bir programlama dili seçilmelidir.

Program yazmak için “program geliştirme ortamı “ denilen programlara ihtiyaç vardır. Yazılan programların hatalarının olup olmadığını kontrol eden bu programlara DERLEYİCİ(compiler) denir. Her dilin bir derleyici programı vardır. Günümüzde C,C++, C#, Java, Visual basic, PHP, ASP gibi programlama dilleri kullanılır.

Dersimizde C# Console programlama için Visual Studio 2010/2015 programını kullanacağız.

Programlama bir dil bilmek değildir.

Programlama=%70 Algoritma + %30 Kodlamadır.

Algoritma: Bir problemin çözüm adımlarıdır.

Algoritma yapılırken

1)Sözde kod yöntemi 2)Akış şeması yöntemi kullanılır.

Yapılan algoritmaları programa dökmek için bir dil seçilir ve o dilin derleyicisi(programı) kullanıır.

Visual Studio 2010 programı kullanılacak.

Bu derste Console uygulaması yapılacak: Console programları bir çalışma noktasından başlayıp, sonuna kadar çalışır birter.

Yazılan program derleme işlemi yapıldığında yazım hatalarını gösterir. Bir programda 2 türlü hata olabilir.

1)Yazım hatası: Kodların yazımında yapılan hatalardır. Derleyici bulabilir. Düzeltilmesi kolaydır. Yazım hatalı bir program çalışmaz.

2) Mantık hatası: Derleyici bulamaz. Algoritmada yapılan hatalardır. Program çalışırken ortaya çıkar. Program çalışır ancak yanlış çalışır.

Her program yazıldıktan sonra mutlaka derleyici ile derlenmesi gerekir. Derleyicinin görevi yazılan programda yazım hatası olup olmadığını kontrol etmektir.

C# Giriş

Her programlama dilinin kendine ait kuralları vardır.

*Her komut satırı ; ile biter.

*Küçük harf büyük harf ayrımı vardır.

*Program içine açıklama satırları // ile yazılır.

*C# komutları küçük harfle yazılır. ÖR: if, while, for…

*Değişkenler tanımlanırken: nasıl tanımlanmışsa öyle kullanılır. ÖR: Oran ~~oran~~

ÇIKIŞ KOMUTLARI(write / writeLine)

Program içinden ekrana (kullanıcıya) bilgi vermek istediğimizde bu komutlar kullanılır. Bilgi sabit metin olabilir. Değişkenlerin değeri olabilir.

Write: Bilgiyi ekrana yazar. İmleç o satıda bekler.daha sonra yazılan bilgi aynı satıra yazılır.

Writeline: Bilgiyi ekrana yazar, imleç bir alt satıra geçer. Sonra yazılan bilgiler alt satıra yazılır.

Metin bilgiler yazılacaksa “ “ içine yazılır.

GİRİŞ KOMUTU: (Readline())

Kullanıcıdan program içindeki değişkenlere bilgi almak için kullanılır. Enter tuşuna basılana kadar girilen bilgileri metin formatında program içine alır. Alınan bilgi programda kullanılması için DEĞİŞKEN denilen yapılarda tutulur. Yani Read line okunan bilgi değişkene konması gerekir. Ancak konmadan önce gerekiyorsa tip değiştirme yapılmalıdır.

DEĞİŞKENLER:

Programa Readline ile okunana veriler veya programda elde edilen veriler program boyunca hafızada saklanmalıdır.

İçine bilgi konulmadan önce isim verilerek ve tipi belirtilerek tanımlanmalıdır.

İsimlendirme kuralları: Türkçe karakter olmaz, tek kelimeden oluşmalıdır._ haricinde özel karakter olmaz. İlk harfi rakam olmaz.Değişken isimleri anlamlı olmak zorunda değil.

~~1sayi~~ sayi1 ~~maaş~~ maas

Oran ~~oran~~

Tip tanımlaması: Değişken içine konulacak bilginin türüne göre değişken tip tanımlaması yapılır.

1)Konulacak bilgi tam sayı ise : int türünde tanımlanmalıdır. ÖR int notu;

2) Konulacak bilgi ondalıklı sayı ise double, float türünde tanımlanır.

ÖR: double enf;

double x,a, ali, oran;

int not1, s1, y;

Değişkenlere tanım yerlerinde bilgi atanabilir. Sonradan da atanabilir.

int z=10;

int k;

k=30;

NOT: Değişkenlere bu şekilde doğrudan değer atayabildiğimiz gibi, Readline komutuyla da değer koyabiliriz. Koyacağımız değer sayısal ise Convert.Toint16() veya Convert.toDouble ile sayısal tipe çevrilmelidir.

3)metin bilgi konulacaksa string tipinde tanımlanmalıdır.

Ör: string ad, soyad, adres;

Not: Metin bilgiler “ “ içine yazılır.

ÖR:

string ad;

ad=”Cengiz”;

NOT: yazdığımız programın çıktısı(output penceresi)nı ekranda görülmesi için program sonuna Console.Readkey() eklenmelidir.

NOT:” “ içinde özel karakterler kullanılarak farklı çıktılar elde edebiliriz.

\t tab karakteri. 8 birim sağa kaydır

\n alt satıra geç.

NOT: Writeline içinde sabit metinler yazılabildiği gibi değişken değerleri de yazılabilir. + işareti sağındaki ve solundaki bilgilerden en az biri metin ise ekleme işlemi yapar. Her ikisi de sayı ise toplama işlemi yapar.

X=10;

```
Console.WriteLine("X değeri="+ x);
```

Çıktı: X değeri=10

```
Console.WriteLine("7"+ x);
```

Çıktısı: 710

```
Console.WriteLine(7+ x);
```

Çıktısı: 17

Not Ekranda değişken değerini yazdırmanın bir diğer yöntemi: Değerlerin yazılacağı yere (yer tutucu denilen) {0}, {1}, ….

İşaretlerini konur.

ÖR

```
int x,y;
            x = 10;
            y = 20;
            Console.WriteLine("X değeri={0}.. Y değeri={1}",x,y);
```

ÖR: Girilen bir sayının karesini bulan prog.

NOT: Bir sayının karekökünü bulmak için Sistemde bulunan Math.sqrt(x) hazır fonksiyon kullanılır. X in karekökünü verir.

1.s, karesi,kk tanımla

2. s oku

3.karesi=s*s

4.kk=Math.sqrt(s)

4.karesini ve kare kökünü yazdır

```
double s, k,kk;
            Console.Write("Bir Sayı Gir");
            s = Convert.ToDouble(Console.ReadLine());
```

```
k = s * s;
kk = Math.Sqrt(s);
Console.WriteLine("sayı=" + s);
Console.WriteLine("Karesi=" + k);
Console.WriteLine("Karekökü=" + kk);
```

ÖR: Kişinin adını ve doğum yılını alarak, yaşını hesaplayıp yazdıran prog.

1.ad,dy,yas tanımla

2.ad ve dy al

3.yas hesapla(yas=2019-dy)

4.ad ve yas yazdır

```
string ad;
            int dy, yas;
            Console.Write("AD Gir");
            ad = Console.ReadLine();
            Console.Write("Doğum Yılı gir");
            dy = Convert.ToInt16(Console.ReadLine());
            yas = 2019 - dy;
```

```
Console.WriteLine("AD: " + ad);
Console.WriteLine("YAŞI: " + yas);
```

ÖR: Kısa ve uzun kenarı girilen dik4genin alan ve çevresi bulan program.

1.Kk,uk,alan,cevre tanımla

2.KK ve UK oku

3.alan=KK*UK;

4.cevre=2*(uk+kk)

5alan ve cevre yazdır.

```
int kk, uk, c, a;
            Console.Write("UZUN KENAR GİR");
            uk = Convert.ToInt16(Console.ReadLine());
            Console.Write("KISA KENAR GİR");
            kk = Convert.ToInt16(Console.ReadLine());
            a = uk * kk;
            c = 2 * (uk + kk);
            Console.WriteLine("ALAN= " + a);
            Console.WriteLine("ÇEVRE= " + c+"
metre");
```

ÖR: Kısa ve uzun kenarı girilen dik4gen şeklindeki arsanın kenarına her 3 metreye 1 direk olacak şekilde direk dikilerek 4 sıra tel ile çevriliyor. Telin

metre fiyatı ve direk adet fiyatını da alarak ödemeyi hesaplayan prog.

1. Uk,kk, tmf,daf,da,tu, odeme,cevre tanımla
2. Uk,kk, tmf,daf al
3. Cevre=2*(uk+kk)
4. Da=cevre/3
5. Tu=cevre*4
6. Ödeme=da*daf+tu*tmf
7. Odeme yazdır

```
int kk, uk, c, tmf, daf, da, tu, odeme;
            Console.Write("UZUN KENAR GİR");
            uk = Convert.ToInt16(Console.ReadLine());
```

```
            Console.Write("KISA KENAR GİR");
            kk = Convert.ToInt16(Console.ReadLine());
            Console.Write("TELİN METRE FİYATI GİR");
            tmf =
Convert.ToInt16(Console.ReadLine());
            Console.Write("DİREK ADET FİYATI GİR");
            daf =
Convert.ToInt16(Console.ReadLine());
            c = 2 * (uk + kk);
            da = c / 3;
            tu = c * 4;
            odeme = da * daf + tu * tmf;
            Console.WriteLine("ÖDEME= " + odeme+"
TL");
```

```
            Console.ReadKey();
```

ÖDEV: R1,R2,R3 direnç değerlerini alarak, seri ve paralel bağlı eşdeğer direnci hesaplayan prog.

KARAR YAPILARI:

Programda belirtilen şartların doğru ya da yanlış olmasına göre program akışını belirleyen yapılardır.

Şart ifadesi sonucu yanlış/doğru olan karşılaştırma ifaderidir. İçinde >,>=,<,<=, ==, != operatörleri bulanan ifadelerdir. Ör x>10 . Basit olabilir (x>0). && (ve), || (veya) bağlaçları ile bağlanmış birden fazla şart ifadesi olabilir.

&&: Sonucun doğru olması için bütün şartların Doğru olması gerekir.

|| : Sonucun doğru olması için en bir tane Doüru olması gerekir.

(x>0 || y==x) && (z<100)

NOT: En fazla kullanılan karar yapısı if() yapısıdır.Belirtilen şart doğru ise program akışını ona göre yönlendirir.

3 Türlü if çeşidi vardır:

a)İf(şart)
{
……..
----- 	
}
	Şartın doğru olması durumunda çalışacak bir kod varsa, yanlış olması durumu önemli değilse(yanlış durumunda

	çalışacak ayrı bir kod yoksa) bu tür if kullanılır.

Ör: girilen sayı(n) pozitif ise ekrana DOĞRU Girdiniz….Mesajı yazmak için:

İf(n>0)

Console.write(“DOĞRU girdiniz.”);

b) if(şart)
{
…..
..
}
Else
{
…..
….
}
	Şartın doğru ve yanlış olduğu durumlarda çalışacak farklı farklı kodlar varsa bu tür kullanılır.
	ÖR: if(x>=0)
	
	Console.write(“POZİTİF”)
	Else
	Console.write(“NEGATİF”)

```
c)  if(şart1)
{
….
…
}
Else if(şart2)
{
….
….
}
Else
{
….
…
..}
```

Birbirine bağlı birden fazla şart ifadesi varsa bu if kullanılır. İflerin çalışması birbirine bağlıdır. Şart1 dogru ise şart 2 kontrol edilmez. Şart1 yanlış ise şart2 kontrol edilir. Doğru ise hemen altındaki yer, o da yanlış ise else kısmı çalışır.

İçiçe if() ler:

İf yapısı içinde başka iflerin de kullanılması durumudur.

```
İf(şart)
{
..
..
        İf(şart)
        {
        …
        ..
        ..
        }
.
}
```

NOT: algoritmaya göre bu karar yapılarından hangisi daha uygunsa istenilen sayıda kullanılır. Bir karar yapısı türü yerine diğerleri de kullanılır.

ÖRNEKLER:

1) Girilen bir sayı pozitif **ise** ekrana POZİTİF, negatif **ise** NEGATİF Sıfır **ise** ekrana SIFIR yazdıran program.

```
int x;

            Console.Write("Bir Sayı Gir");
            x = Convert.ToInt16(Console.ReadLine());
            if (x > 0)
            {
                Console.WriteLine("POZİTİF");
            }
```

```
            else if(x<0)
            {
                Console.WriteLine("NEGATİF");
            }
            else
                Console.WriteLine("SIFIR");
            Console.ReadKey();
```

2.çözüm:

```
int x;

            Console.Write("Bir Sayı Gir");
            x = Convert.ToInt16(Console.ReadLine());
            if (x > 0)
                Console.WriteLine("POZİTİF");
```

```
if(x<0)
    Console.WriteLine("NEGATİF");

if(x==0)
    Console.WriteLine("SIFIR");
```

ÖR:
Bir işçi normal maaşı yanı sıra ürettiği parça sayısına göre ek ücret alıyor. Ps<=4 ise parça başına 5 tl
4<ps<=10 ise parça başına 8 tl,
Ps>10 ise parça başına 10 tl alıyor. Normal maaşı ve parça sayısını alarak, toplam ücreti bulan prg.

```
double n, ps, ek=0, t;
        Console.Write("Normal maaşı gir");
        n = Convert.ToDouble(Console.ReadLine());
```

```
            Console.WriteLine("Parça sayısı Gir");
            ps =
Convert.ToDouble(Console.ReadLine());

            if (ps <= 4)
                ek = ps * 5;

            if (ps > 4 && ps <= 10)
                ek = ps * 8;
            if (ps > 10)
                ek = ps * 10;
            t = n + ek;
            Console.WriteLine("TOPLAM= " + t + "
TL");
            Console.ReadKey();
```

ÖR: Girilen 3 farklı sayının en büyüğünü bulan prog.

```
double x, y, z;
            Console.Write("1. sayıyı gr");
            x = Convert.ToDouble(Console.ReadLine());
            Console.Write("2. sayıyı Gir");
            y = Convert.ToDouble(Console.ReadLine());
            Console.Write("3. sayıyı Gir");
            z = Convert.ToDouble(Console.ReadLine());
            if (x > y && x > z)
                Console.WriteLine("EN BÜYÜK: " + x);
            if (x < y && y > z)
                Console.WriteLine("EN BÜYÜK: " + y);
            if (z > y && x < z)
                Console.WriteLine("EN BÜYÜK: " + z);
```

ÖR: Öğrencinin vize, kısa sınav ve final notunu alarak: Notu=30*vize/100+ 20*ks/100+ 50*final/100 göre hesaplayıp, notunu ve GEÇİP/KALDIğını yazdıran program.
ÖDEV: Girilen bir rakamı yazı ile yazdıran program.
ÖR: 5 girilirse ekrana BEŞ yazacak.

ÖR: 14 girilirse “RAKAM DEĞİL” yazacak.

B) switch(değişken) karar yapısı
İçinde belirtilen değişkenin durumu case ile belirtilen durumlardan hangisine uyuyorsa o kısım çalışır. Hiç birine uymuyorsa ve(varsa) default kısmı çalışır.

```
Switch(değişken)
{
Case  d1 : :  …..
              ….
            Break;
```

```
Case  d2:  :  …..
             ….
           Break;
Case d3:  :  …..
             ….
           Break;

.
.
Default: :  …..
             ….
           Break;
}
```

ÖDEV: Girilen bir rakamı yazı ile yazdıran program.
ÖR: 5 girilirse ekrana BEŞ yazacak.

ÖR: 14 girilirse “RAKAM DEĞİL” yazacak.

```
int rakam;

            Console.Write("Bir rakam Gir");
            rakam = Convert.ToInt16(Console.ReadLine());

            switch (rakam)
            {
                case 0: Console.WriteLine("Sıfır");
                        break;

                case 1: Console.WriteLine("Bir");
                        break;
```

```
                case 2: Console.WriteLine("iki");
                        break;
                case 3: Console.WriteLine("Üç");
                        break;
                case 4: Console.WriteLine("Dört");
                        break;
                case 5: Console.WriteLine("Beş");
                        break;
                case 6: Console.WriteLine("Altı");
                        break;
                case 7: Console.WriteLine("Yedi");
                        break;
                case 8: Console.WriteLine("Sekiz");
                        break;
                    case 9: Console.WriteLine("Dokuz");
                        break;
                    default: Console.WriteLine("RAKAM
DEĞİL");
```

```
                break;
            }

            Console.ReadKey();
```

ÖR: Girilen 2 sayı ve bir işlem işaretine göre sayılar arasında o işlemi yapıp sonucu yazan prg.
ÖR: 2 ve 6 işlem işareti de + girilmişe Sonuç:8 yazacak

```
double s1,s2,sonuc=0;
            string islem;

            Console.Write("Sayı 1 Gir");
            s1 = Convert.ToDouble(Console.ReadLine());
            Console.Write("Sayı 2 Gir");
            s2 = Convert.ToDouble(Console.ReadLine());
```

```
            Console.Write("işlem işareti (+,-,/,*) birini
gir");
            islem = Console.ReadLine();
            switch (islem)
            {
                case "+": sonuc = s1 + s2;
                          Console.WriteLine("Sonuç:"+sonuc);
                          break;
                case "-": sonuc = s1 - s2;
                          Console.WriteLine("Sonuç:" +
sonuc);
                          break;
                case "/": sonuc = s1 / s2;
                          Console.WriteLine("Sonuç:" +
sonuc);
                          break;
                case "*": sonuc = s1 * s2;
```

```
                        Console.WriteLine("Sonuç:" +
sonuc);
                        break;

                default: Console.WriteLine("YANLIŞ İŞARET
GİRDİNİZ");
                    break;
            }

            Console.ReadKey();
```

ÇÖZÜM2)

```
double s1,s2,sonuc=0;
            string islem;
```

```
            Console.Write("Sayı 1 Gir");
            s1 = Convert.ToDouble(Console.ReadLine());
            Console.Write("Sayı 2 Gir");
            s2 = Convert.ToDouble(Console.ReadLine());
            Console.Write("işlem işareti (+,-,/,*) birini
gir");
            islem = Console.ReadLine();
            if(islem=="+")
            { sonuc = s1 + s2;
              Console.WriteLine("Sonuç:"+sonuc);
            }
            else if(islem=="-")
            {sonuc = s1 - s2;
              Console.WriteLine("Sonuç:"+sonuc);


            }
            else if(islem=="/")
```

```
{sonuc = s1 / s2;
  Console.WriteLine("Sonuç:"+sonuc);


}
else if(islem=="*")
{sonuc = s1 * s2;
  Console.WriteLine("Sonuç:"+sonuc);

}
else
    Console.WriteLine("Yanlış işaret");


Console.ReadKey();
```

DÖNGÜLER:

Program içinde defalarca çalışması gereken kod kısımları defalarca yazılmayıp, döngü içine yazılarak, istenilen sayı kadar çalışması sağlanır.Soru içinde bir işlem defalarca yapılacaksa döngüer kullanılır. Ör: 40 sayının ortalaması, 20 öğrencinin notları, vb.
Çok defa çalışması beklenen kodlar döngü içine, bir defa çalışması gereken kodlar döngü dışına yazılmalıdır.
NOT: en çok kullanılan döngü ifadesi for() döngüsüdür.
Döngü çalışmadan önce kaçdefa çalışacağı belli olan durumlarda tercih edilir.Açıkça bir sayacı vardır. Sayaç başlangıçtan bitişe gelene kadar döngü döner.

```
for(sayaç=başlangıç  ; sayaç bitiş kontrol işlemi; sayaçdeğişim
işlemi)
{
…
….
}
```

Not: Başlangıçta kaç defa çalışacağı belli olan durumlarda tercih edilir. Açıkça bir sayacı vardır. Döngünün kaçıncı çalıştığını sayaç tutar.
ÖR: 1 den 50ye kadar 0lan sayıların yan yana yazdırılması.

```
int i;
        for (i = 1; i <= 50; i++)
        {

            Console.Write(i+" ");
        }
```

ÖR: Girilen 5 sayıyı yan yana yazdırma

```
int i, sayi;
            for (i = 1; i <= 5; i++)
            {
                Console.Write("Bir sayi giriniz");
                sayi = Convert.ToInt16(Console.ReadLine());

                Console.WriteLine(sayi+" ");
            }
```

ÖR: girilen 4 öğrenci notundan kaç tanesinin geçtiğini bulan prog

```
int i, notu,gs=0 ;
            for (i = 1; i <= 4; i++)
            {
```

```
            Console.Write("NOTU  giriniz");
            notu = Convert.ToInt16(Console.ReadLine());
            if (notu >= 50)
                gs = gs + 1;
        }
        Console.WriteLine("GEÇEN SAYISI" + gs);
        Console.ReadKey();
```

ÇÖZÜM 2: Döngü kullanmadan

```
int gs = 0, n1, n2, n3, n4;
        Console.Write(" NOT GİR");
        n1 = Convert.ToInt16(Console.ReadLine())
            Console.Write(" NOT GİR");
        n2 = Convert.ToInt16(Console.ReadLine());
        Console.Write(" NOT GİR");
        n3 = Convert.ToInt16(Console.ReadLine());
        Console.Write(" NOT GİR");
```

```
            n4 = Convert.ToInt16(Console.ReadLine());
            if(n1>50)
                gs=gs+1;
            if(n2>50)
                gs=gs+1;
            if(n3>50)
                gs=gs+1;
            if(n4>50)
                gs=gs+1;

            Console.WriteLine("GEÇEN SAYISI"+gs);
            Console.ReadKey();
```

ÖRNEK: 5 öğrencinin vize, ks, ve final notları ile ismini alarak, ort hesaplayıp geçenlerin isimlerini yazdıran prog.

```
double  v, ks, f, ort;
            int i;
            string ad;
```

```
            for (i = 1; i <= 5; i++)
            {
                Console.Write(" İSİM GİR ");
                ad = Console.ReadLine();
                Console.Write(" VİZE GİR ");
                v = Convert.ToInt16(Console.ReadLine());
                Console.Write(" KS GİR  ");
                ks = Convert.ToInt16(Console.ReadLine());
                Console.Write(" FİNAL GİR ");
                f = Convert.ToInt16(Console.ReadLine());

                ort = v * 30 / 100 + ks * 20 / 100 + f * 50
/ 100;

                if (ort >= 50)

                    Console.WriteLine(ad + " GEÇTİ");
            }
```

İÇ İÇE DÖNGÜLER

Tekrar eden bir durumun her bir tekrarında yine tekrar eden durumlar varsa içiçe döngüler kullanılır. Dıştaki döngünün her bir tekrarında içteki döngü baştan sona çalışır biter.

For(i=1;i<=5 i++)
{
 	
For(j=1;j<=4;j++)
{…
…
}
 	
}
	Genellikle satır ve sütun yapısının olduğu durumlarda kullanılır. Çarpım tablosu, matrisler vb.
	İçteki döngünün sayacı ile dıştaki döngünün sayacı farklı olmalıdır. İçteki döngü bitmeden dıştaki döngü bitemez.

ÖR: Çarpım tablosu.

```
for( int i=1;i<=10;i++)
```

```
            {
                for (int j = 1; j <= 10; j++)
                {
                    Console.Write(i+"*"+j+"="+(i *
j)+"\t");
                }
                Console.WriteLine();
            }
```

2.ÇÖZÜM

```
for( int i=1;i<=10;i++)
            {
                for (int j = 1; j <= 10; j++)
                {
                    Console.Write((i * j)+"\t");
                }
                Console.WriteLine();
            }
```

```
            Console.ReadKey();
```

NOT: Döngünün başlangıç ve bitiş değerlerini dışardan girebiliriz.
ÖR: Girilen m ve n değerlerine göre mxn çarpım tablosu.

```
int m, n;
            Console.Write("m değerini gir");
            m = Convert.ToInt16(Console.ReadLine());
            Console.Write("n değerini gir");
            n = Convert.ToInt16(Console.ReadLine());


            for( int i=1;i<=m;i++)
            {
                for (int j = 1; j <= n; j++)
                {
                    Console.Write((i * j)+"\t");
                }
```

```
            Console.WriteLine();
        }
        Console.ReadKey();
```

ÖR: Girilen bir mesajı yine girilen sayı kadar alt alta yazdıran prog.

```
string mesaj;
        int n, i;
        Console.Write("MESAJ GİR");
        mesaj = Console.ReadLine();
        Console.Write("KAÇ DEFA YAZILACAK");
        n = Convert.ToInt16(Console.ReadLine());
        for (i = 1; i <= n; i++)
        {
            Console.WriteLine(mesaj);
        }
```

```
            Console.ReadKey();
```

ÖR: Girilen 2 sayı arasındaki sayıların toplamını bulan prog.

```
int s1, s2, i,t=0;
            Console.Write("İlk sayı gir ");
            s1 = Convert.ToInt16(Console.ReadLine());
            Console.Write("İkinci sayı gir ");
            s2 = Convert.ToInt16(Console.ReadLine());
            for (i = s1; i <= s2; i++)
                t = t + i;
            Console.WriteLine("Sayıların Toplamı= " + t);
            Console.ReadKey();
```

Ör: Girilen bir sayının faktöriyelini bulan program.

```
            double s1, f=1,
                int   i;
               Console.Write(" sayı gir ");
               s1 = Convert.ToDouble(Console.ReadLine());

               for (i = 1; i <= s1; i++)
                   f = f * i;
               Console.WriteLine(s1+ "!= " + f);
               Console.ReadKey();
```

2) While(şart) Döngüsü:
Belirtilen şart doğru olduğu sürece çalışan döngü tipidir.Açıkça bir sayacı yoktur. Döngünün bir süre sonra sonlanması için şartı oluşturan ifadeler (değişkenler) mantıklı bir şekilde değiştirilmelidir.

```
While(şart)
{
..
……
}
```

ÖR: 1 den 10 kadar olan sayıları yazdırma.

```
int x;
            x = 1;
            while (x <= 10)
```

```
            {
                Console.WriteLine(x);
                x = x + 1;
            }

            Console.ReadKey();
```

ÖR: Negatif bir sayı girilene kadar girdiğimiz sayıların toplamını veren prog.

```
double s, t = 0;
            Console.Write("Bir sayı gir");
            s = Convert.ToDouble(Console.ReadLine());
            while (s > 0)
            {
                t = t + s;
                Console.Write("sayı gir");
                s = Convert.ToDouble(Console.ReadLine());
```

```
        }
        Console.WriteLine("TOPLAMI: " + t);
        Console.ReadKey();
```

RASTGELE SAYILAR ÜRETME:
Programda belirtilen aralıkta rastgele sayılar üretmek için Random fonksiyonu kullanılır. Önde Random türünde bir değişken tanımlanır. Daha sonra bu değişkenin next ifadesi ile sayı üretilir.

```
Random rasgele= new Random();

İnt x= rasgele.next(1,100)
```

ÖR: 10 rasgele sayı üreterek alt alta yazın

```
Random rasgele = new Random();
            int s;

            for(int i=1;i<=10;i++)
            {
            s = rasgele.Next(1, 100);
            Console.WriteLine("Rasgele üretilen sayı : " +
s);
            }
            Console.ReadKey();
```

ÖR: rasgele üretilen 10 sayının toplamını bulan program;

```
Random rasgele = new Random();
            int s;
            int t = 0;
```

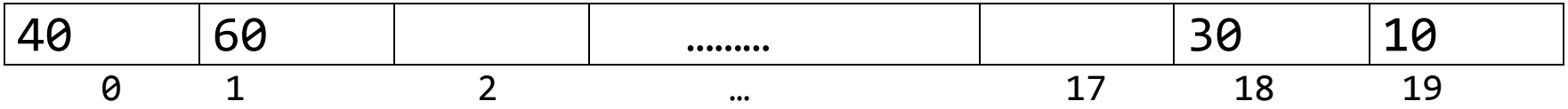
```
            for(int i=1;i<=10;i++)
            {
            s = rasgele.Next(1, 100);
            Console.WriteLine("Rasgele üretilen sayı : " +
s);
            t = t + s;
            }
            Console.WriteLine("Rasgele üretilen sayıların
toplamı : " + t);
            Console.ReadKey();
```

DİZİLER:
Normal bir değişken aynı anda 1 tek değer tutabilir. Değişkenin üzerine başka bir değer konduğunda eski değer kaybolur. Eğer girdiğimiz değerler kullanıldıktan sonra yeniden ihtiyacımız varsa dziye konmalıdır.
Dizi : aynı tipte bir çok bilgiyi aynı anda tutabilir. Diziler de bir dür değişkendir.
Eski bilgileri kaybetmemek için ya bir çok farklı değişkene koymak gerekir(ki bu uygun değildir), ya da diziye koymak gerekir. Bir çok bilgiyi bir değişken adı altında toplama işlemidir.

Tanımlama:

Tipi [] diziismi=new tipi [boyut];

Ör: int [] notlar=new int [20];
Notlar isminde içinde 20 sayı tutabilecek bir dizi tanımlandı.



Dizilerle çalışırken dizi ismi ve eleman numarası(indis) kullanılır. Ör notlar[1] .
Dizilerede indis 0 dan başlar. İlk kutu nosu 0, son kutu nosu boyut-1 ‘ dir.

Notlar[18]=90;
Bir değişkene nasıl dışarıdan veri alınıyorsa dizilere de alınabilir.
Ör
Notlar[2]=Convert.toint16(Console.readline())
komutu ile veri okunabilir.

ÇOK ÖNEMLİ NOT:Dizilerle çalışırken dizi ismi ve eleman numarası belirtmek zorunda olduğumuz için, dizi elemanlarına tek tek bilgi okuma ve yazma yapılmaz. Bunun yerine döngü içinde işlem yapılır.
Veri girişine örnek:

```
For(i=0; i<20 ; i++)
{
   Console.write(“Bir Not gir”);
   Notlar[i]= convert.toint16(Console.readline());

}
```

Veri yazdırma örnek:

```
For(i=0; i<20 ; i++)
{
   Console.writeline(Notlar[i]);

}
```

Dizi tanımlamaya örnek:

```
String [] isimler=new string[10];
```

10 kişinin isimlerini tutan isimler adında bir dizi.

FİNAL NOTU: 4 soru sorulacak. Bunlardan 1'i tanım, diğerleri program sorusu olacak.
Sınav Süresi : 70 dakika.
Soru sorma süresi: İlk 15 dakika.
Sınavda tek kağıt kullanılacaktır.

GENEL ÖRNEK:

Girilen 3 sayıya göre AZALAN , ARTAN, KARIşIK olduğunu yazdıran prog.

123 45 3 ise AZalan

34 678 12345 ise artan

123 23 100 ise karışık

```
int a, b, c;

            Console.Write("1. sayı gir");
            a = Convert.ToInt16(Console.ReadLine());
            Console.Write("2. sayı gir");
            b = Convert.ToInt16(Console.ReadLine());
            Console.Write("3. sayı gir");
            c = Convert.ToInt16(Console.ReadLine());

            if (a > b && b > c)
                Console.WriteLine("AZALAN SIRA");
```

```
else if(a<b && b<c)
    Console.WriteLine("ARTAN SIRA");
else
    Console.WriteLine("KARIŞIK SIRA");
```