NESNE TABANLI PROGRAMLAMA

MARDİN ARTUKLU ÜNİVERSİTESİ

HAZIRLAYAN: Öğr.Gör. Hüseyin AHMETOĞLU

5.HAFTA

Döngüler

while döngüsü

- While döngüsü bir şart sağlanıyor *iken* sürekli içindeki komutları çalıştırır. Kelime anlamı olarak "olduğu müddetçe" anlamı çıkar. Yapısı şöyledir:

```
while (koşul) {
      //komut 1
      //komut 2
      //...
      //komutlar
}
```

while döngüsü

- While döngüsünde koşulun en başta olması ayırt edici bir özelliktir.
- Program akışı önce buradaki koşulu mantıksal bir süzgeçten geçirir. Buna göre eğer koşula yazılan ifade matematiksel olarak **true (doğru)** değer döndürüyorsa, süslü parantezlerle çevrili *blok*taki komutlar sırasıyla çalıştırılır.
- Ancak **false (yanlış)** değerini döndürüyorsa, bu komutlar hiçbir zaman çalışmayacaktır. Bu durumda program akışına döngünün bittiği yerden devam eder.

while döngüsü

- Koşulun **true (doğru)** döndürmesinin ardından bloktaki komutlar sırasıyla çalıştırılır ve son komut da çalıştırıldığında akış tekrardan koşulun olduğu satıra gelir ve koşulun **true (doğru)** değer döndürüp döndürmediği kontrol edilir. Koşul doğru olduğu müddetçe komutlar çalıştırılır.

MAUZEM

Örnek 5.1


```
public class WhileDongusu1 {
    public static void main(String[] args) {

        int sayac=10;
        while (sayac>=1) {
            System.out.println(sayac);
            sayac--;
        }
    }
}
```

```
10
9
8
7
6
5
4
3
2
1
```

Örnek 5.2


```
public class WhileDongusu2 {
    public static void main(String[] args) {

        int faktoriyel=1;
        int carpan=1;
        while(carpan<=10) {
            faktoriyel=faktoriyel*carpan;  //1*1, 1*2, 2*3, 6*4, 24*5...
            System.out.println(carpan+" faktoriyel: "+faktoriyel);
            carpan++;
        }
    }
}
```

```
1 faktoriyel: 1
2 faktoriyel: 2
3 faktoriyel: 6
4 faktoriyel: 24
5 faktoriyel: 120
6 faktoriyel: 720
7 faktoriyel: 5040
8 faktoriyel: 40320
9 faktoriyel: 362880
10 faktoriyel: 3628800
```

do ... while Döngüsü

- *while* deyiminin önemli bir özelliği, test koşulunun döngünün başlangıcında olmasıdır. Bunun anlamı, ilk anda koşulun yanlış olması halinde, *while* gövdesinin hiçbir zaman çalıştırılmayacağıdır. Ancak gövdeyi hiç olmazsa bir kere çalıştırmamız gereken durumlar vardır. Bu durumlar çok yaygın olmasa da gerektiği zaman ***do...while*** deyimini kullanmalıyız.

do ... while Döngüsü

- *do..while* döngü yapısının yazılışı şu şekildedir:

```
do {
     Deyim1;
     Deyim2;
     ...
     Deyim_n;
} while(İfade) ;
```

do ... while Döngüsü

- *do...while* ve *while* döngüleri arasındaki tek fark, test koşulunun (ifadenin) *do while* döngüsünde, döngünün sonunda yer almasıdır. Bunun anlamı, programın döngüyü hiç olmazsa bir kez çalıştırmasıdır (ilk kez). Sonra, ifadenin değerine bağlı olarak, ifade **doğru** ise tekrar *do*'ya dönülerek döngü sürebilir veya ifade **yanlış** ise bir sonraki deyimle devam edilebilir.

do ... while Döngüsü


```
int sayac = 1;
do {
 System.out.println("Sayaç: " + sayac);
 sayac++;
} while (sayac < 11);
```

for Döngüsü

- *for deyimi* ve for deyimi kullanılarak oluşturulacak döngü yapısı, işlemlerin tekrar sayısının önceden belli olduğu durumlarda kullanılır.

```
for(ifade1;ifade2;ifade3)
            { Deyim1;
              Deyim2;
                ...
              Deyim_n;
            }
            Deyim_x;
```

Örnek 5.3


```java
import java.util.Scanner;

public class PozTop{
    public static void main(String args[]) {
        Scanner giris = new Scanner(System.in);
        int n, toplam;
        int i;
        System.out.println("Kaca kadar sayilar toplansin? ");
        n = giris.nextInt();

        toplam = 0;

        for (i = 1; i <= n; i++) {
            toplam += i;
        }
        switch (n) {
        case 1 :
            System.out.println("\n" + n + "=" + toplam);
            break;
        case 2 :
            System.out.println("\n1+" + n + "=" + toplam);
            break;
        case 3 :
            System.out.println("\n1+2+" + n + "=" + toplam);
            break;
        case 4 :
            System.out.println("\n1+2+3+" + n + "=" + toplam);
            break;
        default :
            System.out.println("\n1+2+3+..+" + n + "=" + toplam);
        }
    }
}
```

```
Kaca kadar sayilar toplansin?
20

1+2+3+..+20=210
```

Örnek 5.4 (ALTERNE SERİ TOPLAMI)

- ***1.2/(3.4) - 5.6/(7.8) + 9.10/(11.12) - 13.14/(15.16)***
- (Yukardaki ifadede "." sembolü çarpma işlemi anlamındadır.)

```
public class Alterne {
    public static void main(String args[]){
        int i,p;
        double x,s;
        s=0;
        p=1;
        x=1.0;
        for (i=1;i<=4;i++)
        {
            s=s+(x*(x+1))/((x+2)*(x+3))*p;
            x=x+4;
            p=-p;
        }
        System.out.println("Toplam= "+s);
    }
}
```

```
toplam=-0.4455627705627706
```

Örnek 5.4 (ALTERNE SERİ TOPLAMI)


```
public class Alterne {
    public static void main(String args[]){
        int i,p;
        double x,s;
        s=0;
        p=1;
        x=1.0;
        for (i=1;i<=4;i++)
        {
            s=s+(x*(x+1))/((x+2)*(x+3))*p;
            x=x+4;
            p=-p;
        }
        System.out.println("Toplam= "+s);
    }
}
```

Programın Çalıştırılması Sırasında Alterne Seri Programındaki Değişkenlerin Değişimlerini Gösteren Tablo

i	x	s	p
Çevrim öncesi	1.0	0	1
1	1.0	(1.2)/ (3.4)	1
2	5.0	(1.2)/ (3.4) – (5.6)/ (7.8)	–1
3	9.0	(1.2)/ (3.4) – (5.6)/ (7.8) +(9.10)/ (11.12)	1
4	13.0	(1.2)/ (3.4) – (5.6)/ (7.8) +(9.10)/ (11.12) – (13.14)/(15.16)	–1

İç İçe Döngüler

- Bir döngü yapısının içine başka bir döngü yapısının yerleştirilmesiyle elde edilen yapıya **iç içe döngü (nested loop)** adı verilir.
- Java dilinde, *if* deyimlerini herhangi bir derinliğe kadar iç içe kullanmak nasıl mümkünse, döngü deyimlerini de iç içe kullanmak olasıdır. Şu kural iç içe döngüler için daima geçerlidir:
- ***İç içe döngülerde en içteki döngü en önce tamamlanır.***

Örnek 5.5


```
public class IcIce {
    public static void main(String args[]) {
        int a, i; //5 kez tekrarla
        for (a = 1; a <= 5; a++) {
            System.out.println("a= " + a); //3 kez tekrarla
            for (i = 1; i <= 3; i++) {
                System.out.println("i= " + i);
            }
            System.out.println();
        }
    }
}
```

```
a= 1
i= 1
i= 2
i= 3

a= 2
i= 1
i= 2
i= 3

a= 3
i= 1
i= 2
i= 3

a= 4
i= 1
i= 2
i= 3

a= 5
i= 1
i= 2
i= 3
```

Örnek 5.6


```
public class CarpimTablosu {
    public static void main(String args[]) {
        int altcizgi, i, j;
        System.out.println(" 1 2 3 4 5 6 7 8 9 10 "); //döngüyle uzunca bir çizgi çizelim
        for (altcizgi = 1; altcizgi <= 70; altcizgi++) {
            System.out.print("_");
        }
        System.out.println();
        for (i = 1; i <= 10; i++) {
            System.out.print(i + " | ");
            for (j = 1; j <= 10; j++) {
                System.out.print(i * j + " ");
            }
            System.out.println();
        }
    }
}
```

```
          1    2    3    4    5    6    7    8    9   10
______________________________________________________________________
1  |    1    2    3    4    5    6    7    8    9   10
2  |    2    4    6    8   10   12   14   16   18   20
3  |    3    6    9   12   15   18   21   24   27   30
4  |    4    8   12   16   20   24   28   32   36   40
5  |    5   10   15   20   25   30   35   40   45   50
6  |    6   12   18   24   30   36   42   48   54   60
7  |    7   14   21   28   35   42   49   56   63   70
8  |    8   16   24   32   40   48   56   64   72   80
9  |    9   18   27   36   45   54   63   72   81   90
10 |   10   20   30   40   50   60   70   80   90  100
```

Örnek 5.7


```
class FaktoriyelliSeri {
    public static void main(String args[]) {
        double s, fakt;
        int i, j;
        s = 0;
        for (i = 2; i <= 10; i++) {
            fakt = 1;
            for (j = 1; j <= i; j++) {
                fakt = fakt * j;
            }
            s = s + 1 / fakt;
            System.out.println(s);
        }
        System.out.println("Seri toplami " + s);
    }
}
```

```
0.5
0.6666666666666666
0.7083333333333333
0.7166666666666666
0.7180555555555554
0.7182539682539681
0.7182787698412697
0.7182815255731921
0.7182818011463844
Seri toplami 0.7182818011463844
```

break Deyimi

- ***break*** deyiminin Java programları içinde iki farklı kullanım alanı vardır:
- *switch* yapısındaki *case* seçeneklerinden birinde *switch* yapısını terk etmek ve *switch*'i izleyen deyime geçmek için kullanılır. (Bu tip kullanımla ilgili örnekler *switch* deyimi anlatılırken verilmiştir.)
- Bir döngü (loop) yapısı içinden, döngüyü kontrol eden koşul ifadesini beklemeksizin döngü dışına çıkmak için kullanılır.
- *break* deyimi bir döngü içinde yer almışsa bu durumda *break* deyimi ile karşılaşır karşılaşmaz döngü dışına çıkılır ve döngüyü izleyen deyime geçilir.

break Deyimi


```
public class BreakClass {
    public static void main(String args[]) {
        Scanner giris = new Scanner(System.in);
        double ort;
        int sayac, toplam, sayi;
        toplam = 0;
        sayac = 0;
        System.out.println("Alt alta pozitif sayilar gireceksiniz ve negatif sayi");
        System.out.println("girene kadar onlar toplanacak. Negatif girdiginizde");
        System.out.println("toplama isleminin sonucu size verilecektir.");
        while (true) {
            System.out.print("Sayi giriniz: ");
            sayi = giris.nextInt();

            if (sayi < 0) {
                break;
            }
            toplam += sayi;
            sayac++;
        }
        ort = toplam / sayac;
        System.out.println("Toplami " + toplam + " olan " + sayac + " tane sayi girdiniz.");
        System.out.println("Sayilarin ortalamasi da " + ort);
    }
}
```

```
Alt alta pozitif sayilar gireceksiniz ve negatif sayi
girene kadar onlar toplanacak. Negatif girdiginizde
toplama isleminin sonucu size verilecektir.
Sayi giriniz: 2
Sayi giriniz: 4
Sayi giriniz: 5
Sayi giriniz: 11
Sayi giriniz: 25
Sayi giriniz: 6
Sayi giriniz: -5
Toplami 53 olan 6 tane sayi girdiniz.
Sayilarin ortalamasi da 8.0
```

continue Deyimi

- **continue** deyimi, döngü içinde belirli bir koşulun kontrolüyle bir sonraki döngü adımına gidilmesini gerçekleştirir.

continue Deyimi


```
public class Cont{
    public static void main(String args[]) {
        Scanner giris = new Scanner(System.in);
        int toplam, i, ustlimit, sayac;
        System.out.println("Gireceginiz ust limite kadar olan ve");
        System.out.println("3'e bolunebilen sayilar bulunacaktir.");
        System.out.println();
        System.out.println("Ust limiti giriniz: ");
        ustlimit = giris.nextInt();
        toplam = 0;
        sayac = 0;
        for (i = 1; i <= ustlimit; i++) {
            if (i % 3 != 0) {
                continue;
            }
            System.out.println("3'e bolunebilen sayi: " + i);
            toplam += i;
            sayac++;
        }
        System.out.println("3-" + ustlimit + " arasindaki 3 ile bolunebilen "+sayac+" tane sayi var.");
        System.out.println("Bu tur sayilarin toplami " + toplam);
    }
}
```

```
Gireceginiz ust limite kadar olan ve
3'e bolunebilen sayilar bulunacaktir.

Ust limiti giriniz:
30
3'e bolunebilen sayi: 3
3'e bolunebilen sayi: 6
3'e bolunebilen sayi: 9
3'e bolunebilen sayi: 12
3'e bolunebilen sayi: 15
3'e bolunebilen sayi: 18
3'e bolunebilen sayi: 21
3'e bolunebilen sayi: 24
3'e bolunebilen sayi: 27
3'e bolunebilen sayi: 30
3-30 arasindaki 3 ile bolunebilen 10 tane sayi var.
Bu tur sayilarin toplami=165
```

KAYNAKLAR

https://gelecegiyazanlar.turkcell.com.tr/