ALGORİTMA VE PROGRAMLAMA I

Yrd. Doç. Dr. Deniz KILINÇ
deniz.kilinc@cbu.edu.tr

YZM 1101

Celal Bayar Üniversitesi Hasan Ferdi Turgutlu
Teknoloji Fakültesi

Genel Bakış…

- Döngüler
- for Döngüsü
- while Döngüsü
- do-while Döngüsü
- break Deyimi Kullanımı
- continue Deyimi Kullanımı
- İç İçe Geçmiş Döngüler
- Sonsuz Döngü

1. BÖLÜM

Döngüler

Döngü

- Döngü (loop) deyimleri, bir işlemi yerine getiren kod kümesinin belli bir koşul altında tekrar edilmesi için kullanılır.
- Bir değişken belirli bir değerden başlayıp, son değeri alıncaya kadar belirtilen işlemler tekrarlanır.
  - Örn: n! değerinin hesaplanması (faktöryel)
- C programlama dilinde:
  - for,
  - while,
  - do...while

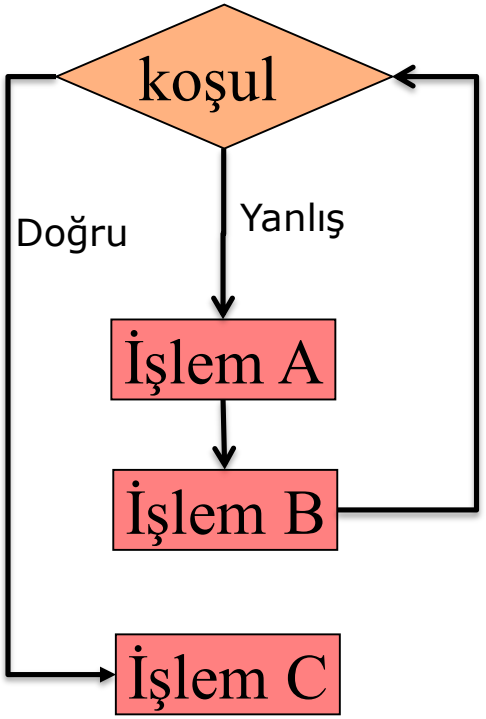
  olmak üzere üç tip döngü deyimi vardır.
- Diğer programlama dillerinde olduğu gibi, bu deyimlerle istenildiği kadar iç-içe döngü yapısı kullanılabilir.

for Döngüsü

- Belirli sayıda tekrarı çalıştırmak için kullanılır.
- Döngü sayısını denetlemek için bir başlangıç ve bir bitiş değeri belirtilmektedir.
- Normal durumda sayaç birer birer artmaktadır.
- Genel yazım biçimi aşağıdaki gibidir:

```
for (başlangıç; koşul; artım)
{
        ...
         döngüdeki deyimler;
        ...
}
```

for Döngüsü (devam…)




```
for (int i = başla ; koşul; i=i+artış miktarı)
{
        işlem A;
        işlem B;
        .....
 }
işlem C;
```

Örnek1: 1-10 arasındaki sayıların yazdırılması


```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int i;

    for (i=1; i<=10; i++){
        printf("i: %d \n",i);
    }

    return 0;
}
```

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

for Döngü Yapısı Örnekleri

- 1'den 100'e kadar birer birer arttırma

```
for ( i = 1; i <= 100; i++)
```

- 100'den 1'e kadar birer birer azaltma

```
for ( i = 100; i >= 1; i--)
```

- 7'den 77'ye kadar yedişer yedişer arttırma

```
for ( i = 7; i <= 77 ; i += 7)
```

- 2, 5, 8, 11, 14, 17, 20 değerlerini alacak biçimde değiştirme

```
for ( j = 2; j <=20; j += 3)
```

Örnek2: Başlangıç bitiş değişkenleri arasındaki sayıların toplamı

- Döngünün başlangıç ve bitiş değeri tam sayı olacaktır.
- Döngünün başlangıç ve bitiş değerleri dışarıdan girilmelidir. Kullanıcıya başlangıç ve bitiş değeri sorulmalıdır.
- Örnek:
  - Başlangıç değeri giriniz:2
  - Bitiş değeri giriniz: 5
  - Toplam: 14

Örnek2: Başlangıç bitiş değişkenleri arasındaki sayıların toplamı


```c
/*
    Girilen başlangıç ve bitiş değerleri arasındaki
    sayıların toplamını for döngüsü kullanarak toplar.
*/
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int sayac, baslangic, bitis;
    int toplam=0;

    printf("Baslangic sayisini giriniz:");
    scanf("%d",&baslangic);

    printf("Bitis sayisini giriniz:");
    scanf("%d",&bitis);

    for (sayac=baslangic; sayac<=bitis; sayac++){
        toplam = toplam + sayac;
    }

    printf("Toplam:%d",toplam);

    return 0;
}
```

```
Baslangic sayisini giriniz:2
Bitis sayisini giriniz:5
Toplam:14
```

Örnek3: Girilen bir sayının faktöriyel değerinin hesaplanması…

Uygulama dersinde yapacağız…

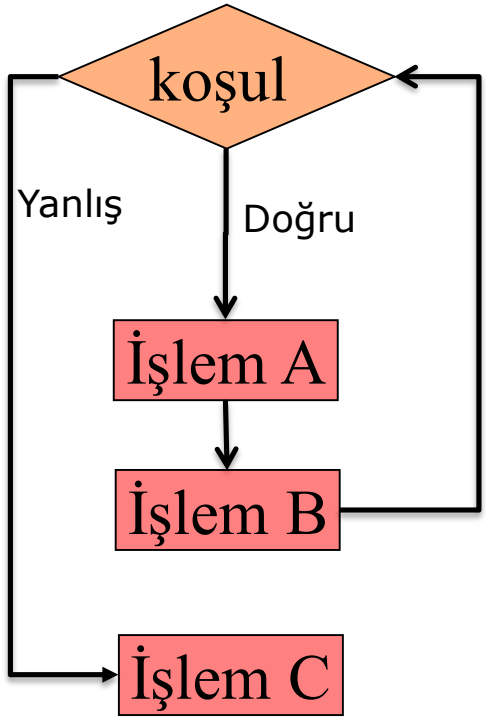
(While ve For ile ayrı ayrı yapalım…)

while Döngüsü

- Bir koşulun gerçekleşmesi durumunda belirli işlemlerin tekrarlanması söz konusu ise while döngülerinden yararlanılır.
- while döngüsünün çalışabilmesi için koşulun başlangıçta mutlaka doğru olması gerekir.
- Genel yazım biçimi aşağıdaki gibidir:

```
while (koşul)
{
    ...
    döngüdeki deyimler;
    ...
}
```

while Döngüsü (devam…)




```
while (koşul )
{
        İşlem A;
        İşlem B;
        ..........
}

İşlem C;
..........
```

Örnek4: 0 girilene kadar, girilmiş tüm sayıların toplanması...

- Girilen sayı tam sayı olacaktır. Bu tam sayı dışarıdan okunacaktır.
- Girilen sayı 0 olmadığı sürece yeni sayı girilmeye devam edilecektir.
- Her yeni sayı bir önceki ile toplanacak ve toplam saklancaktır.
- Örnek:
  - Bir sayi giriniz:2
  - Bir sayi giriniz:3
  - Bir sayi giriniz:4
  - Bir sayi giriniz:0
  - Toplam: 9

Örnek4: 0 girilene kadar, girilmiş tüm sayıların toplanması...


```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int girilen_sayi, toplam = 0;

    while (girilen_sayi != 0){
        printf("Bir sayi giriniz:");
        scanf("%d",&girilen_sayi);
        toplam += girilen_sayi;
    }

    printf("Toplam:%d", toplam);

    return 0;
}
```

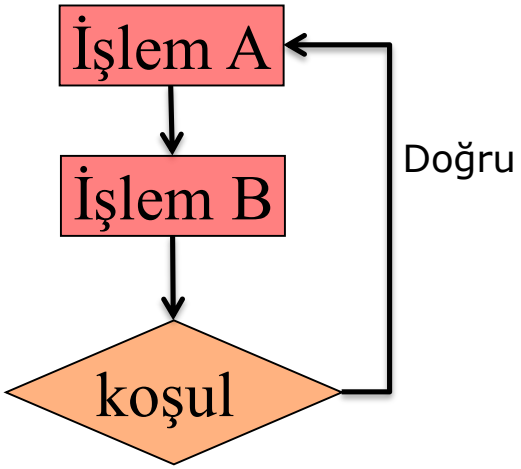
```
Bir sayi giriniz:2
Bir sayi giriniz:3
Bir sayi giriniz:4
Bir sayi giriniz:0
Toplam:9
```

do…while Döngüsü

- while döngüsü ile aynı mantıkta çalışır.
- Farkı, koşulun döngü sonunda sınanmasıdır.
- Yani koşul sınanmadan döngüye girilir ve döngü en az bir kez yürütülür. Koşul olumsuz ise döngüden sonraki satıra geçilir.
- Genel yazım biçimi aşağıdaki gibidir:

```
do
{
    ...
     döngüdeki deyimler;
    ...
} while (koşul);
```

do…while Döngüsü (devam…)




```
do {

        İşlem A;
        İşlem B;
        ..........

} while (koşul);
```

Örnek5: 0 girilene kadar, girilmiş tüm sayıların toplanması... (do - while)

- Girilen sayı tam sayı olacaktır. Bu tam sayı dışarıdan okunacaktır.
- Girilen sayı 0 olmadığı sürece yeni sayı girilmeye devam edilecektir.
- Her yeni sayı bir önceki ile toplanacak ve toplam saklancaktır.
- Örnek:
  - Bir sayi giriniz:2
  - Bir sayi giriniz:3
  - Bir sayi giriniz:4
  - Bir sayi giriniz:0
  - Toplam: 9

Örnek5: 0 girilene kadar, girilmiş tüm sayıların toplanması... (do - while)


```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int girilen_sayi, toplam = 0;

    do {
        printf("Bir sayi giriniz:");
        scanf("%d",&girilen_sayi);
        toplam += girilen_sayi;
    } while (girilen_sayi != 0);

    printf("Toplam:%d \n", toplam);

    return 0;
}
```

Örnek6: 0 girilene kadar, girilen sayıların karelerini bulma…(do - while)

- Girilen sayı tam sayı olacaktır. Bu tam sayı dışarıdan okunacaktır.
- Girilen sayı 0 olmadığı sürece yeni sayı girilmeye devam edilecektir.
- Örnek:
  - Bir sayi giriniz:2
  - Karesi: 4
  - Bir sayi giriniz:3
  - Karesi:9
  - Bir sayi giriniz:0
  - Karesi: 0

Örnek6: 0 girilene kadar, girilen sayıların karelerini bulma…


```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int girilen_sayi;
    do {
        printf("Bir sayi giriniz:");
        scanf("%d",&girilen_sayi);
        printf("Karesi:%d \n\n",(girilen_sayi*girilen_sayi));
    } while (girilen_sayi != 0);

    return 0;
}
```

```
Bir sayi giriniz:2
Karesi:4

Bir sayi giriniz:3
Karesi:9

Bir sayi giriniz:4
Karesi:16

Bir sayi giriniz:0
Karesi:0
```

break Deyimi

- Döngü işlemi devam ederken döngünün koşuluna bağlı olmaksızın döngüden çıkılmasını sağlayan deyimdir.
- Döngü içinde bu deyime sıra geldiğinde, break ardından döngü sonuna kadar olan tüm deyimler atlanır ve döngüye bir sonraki adımdan itibaren devam edilir.
- Tüm döngü türlerinde kullanılabilir.
- Kullanım biçimi aşağıdaki gibidir:

```
break;
```

Örnek7: 0 girilene kadar, girilen sayıların karelerini bulma… (break kullanarak)


```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int girilen_sayi;
    do {
        printf("Bir sayi giriniz:");
        scanf("%d",&girilen_sayi);
        if (girilen_sayi == 0)
            break;
        printf("Karesi:%d \n\n",(girilen_sayi*girilen_sayi));
    } while (girilen_sayi != 0);

    return 0;
}
```

```
Bir sayi giriniz:2
Karesi:4

Bir sayi giriniz:3
Karesi:9

Bir sayi giriniz:4
Karesi:16

Bir sayi giriniz:0
```

continue Deyimi

- Bir döngüyü terk etmeden bir adımın atlanması söz konusu olduğunda kullanılan deyimdir.
- Kullanım biçimi aşağıdaki gibidir:

```
continue;
```

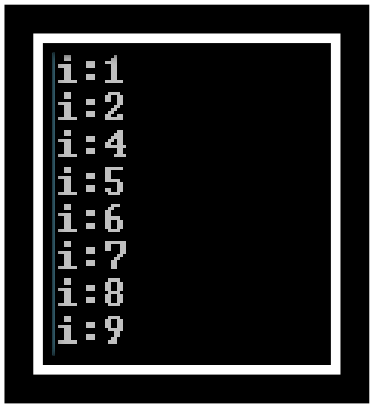
Örnek8: 1-10 arası sayıları yazdır, 3 değeri için devam et (continue kullanarak)


```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int i;

    for (i=1; i<10; i++){
        if (i==3)
            continue;
        printf("i:%d\n",i);
    }

    return 0;
}
```



İç İçe Geçmiş Döngüler

- Bir program içinde birbiri içine geçmiş birden çok döngü kullanılabilir. Bu durumda (bütün programlama dillerinde olduğu gibi) **önce içteki döngü**, **daha sonra dıştaki döngü** tamamlanır.

```
for (i=1; i<n; i++){
                    //1.dış döngü
        for (j=1; j<m; j++) {
                    //2.iç döngü
        }
}
```

İç İçe Geçmiş Döngüler


```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int i, j, k;

    for (i = 1; i <= 5; i++)
    {
        printf("i: %d\n", i);

        for (j = 1; j <= 10; j++)
        {
            printf("\t j: %d\n", j);
        }

    }
    return 0;
}
```

Örnek9: 1-50 arasındaki asal sayıların yazdırılması.

- Sadece kendisi ve 1 sayısına bölünebilen 1'den büyük pozitif tam sayılar biçiminde tanımlanırlar.
- Asal sayıların 1 ve kendisinden başka tam böleni yoktur.

Örnek: Girilen bir sayının asal olup olmadığını nasıl buluruz?

- 1'den başlayıp, sayının kendi değerine kadar devam edecek bir döngü tanımlarız.
- Döngü içerisinde eğer (Sayı mod Sayac) = 0 ise döngüden çıkarız.
- Eğer (Sayı == Sayac) ise bu sayı asal sayıdır.

Örnek9: 1-50 arasındaki asal sayıların yazdırılması (devam…)

Örnek: Girilen bir sayının asal olup olmadığını nasıl buluruz?

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int sayi, i;

    printf("Bir sayi giriniz:");
    scanf("%d",&sayi);

    for (i=2; i<sayi; i++) {
        if (sayi % i == 0)
            break;
    }
    if (sayi == i )
        printf("%d asal sayidir \n",sayi);
    else
        printf("%d asal sayi degildir \n",sayi);

    return 0;
}
```

Örnek9: 1-50 arasındaki asal sayıların yazdırılması (devam…)


```c
#include <stdio.h>
#include <stdlib.h>


int main()
{
    int i, j;
    for (i=2; i<50; i++){
        for (j=2; j<i; j++) {
            if (i % j == 0)
                break;
        }
        if (i == j )
            printf("Asal sayi: %d\n",i);
    }
    return 0;
}
```

```
Asal sayi: 2
Asal sayi: 3
Asal sayi: 5
Asal sayi: 7
Asal sayi: 11
Asal sayi: 13
Asal sayi: 17
Asal sayi: 19
Asal sayi: 23
Asal sayi: 29
Asal sayi: 31
Asal sayi: 37
Asal sayi: 41
Asal sayi: 43
Asal sayi: 47
```

Sonsuz Döngüler

- Bir döngü işlemini sonsuz kere tekrarlarsa bu döngü sonsuz döngü olarak adlandırılır.

Sonsuz Döngüler (devam...)

- Örnekler:

```
while(1) {
   printf("Sonsuz döngü...\n");
}

while(7>3) {
   printf("Sonsuz döngü...\n");
}

for(;;)
   printf("Sonsuz döngü...\n");
```

KAYNAKLAR

- N. Ercil Çağıltay ve ark., C DERSİ PROGRAMLAMAYA GİRİŞ, Ada Matbaacılık, ANKARA; 2009.
- Milli Eğitim Bakanlığı "Programlamaya Giriş ve Algoritmalar Ders Notları", 2007
- http://tr.wikipedia.org/wiki/Code::Blocks
- http://www.codeblocks.org
- http://www.AlgoritmaveProgramlama.com
- http://www1.gantep.edu.tr/~bingul/c

İYİ ÇALIŞMALAR…

Yrd. Doç. Dr. Deniz KILINÇ
deniz.kilinc@cbu.edu.tr