

Programlamaya Giriş

HAFTA 4

Karar Yapıları, Tekrarlı İfadeler (Döngüler)

(C.Öz, C.Bayılmış, G.Çit Ders Notları)

Konu & İçerik

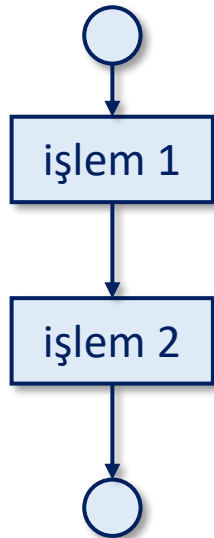
- Program Blokları
- Sıra Yapısı
- Kontrol Yapıları
 - Karar Yapıları (If)
 - If/Else
 - İç İçe If/Else
- Switch-Case (Çoklu Dallanma Yapıları)
- Döngüler
 - For
 - While
 - Do
- Mantıksal İşleçler(And, Or, Not)
- Break-Continue İfadeleri
- Goto
- Sorular
- Kaynaklar

Program Blokları

➤ Programlar 3 bloktan oluşur

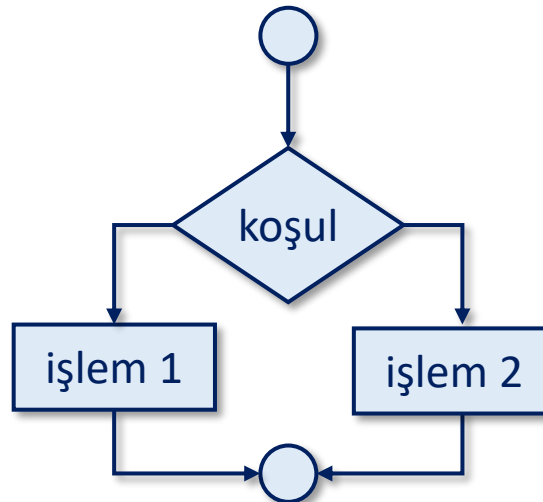
➤ Sıralı

Bir dizi işlem birbiri ardından sırayla gerçekleştirilir.



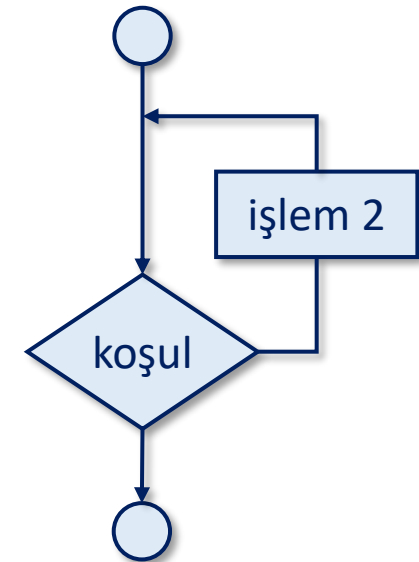
➤ Seçme / Kontrol

İki seçenekten hangisinin izleneceği koşula bağlıdır.



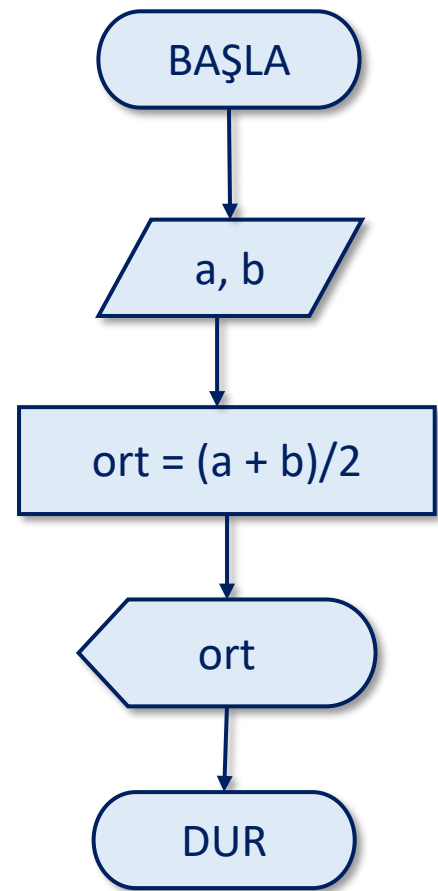
➤ Döngü

İki seçenekten hangisinin izleneceği koşula bağlıdır.



Sıra Yapısı

- En basit akış şeması ifadeleri, bir dizi işlemin birbiri ardından sırasıyla yapılmasını şeklinde olan akış ifadeleridir. Bu tip akışlar oldukça yalın ve basittir.
- Bu tarz akışlar genelde bir problemin bir parçasını çözmek ve ifade etmek için kullanılır.
- Sorgu ve tekrar gerektirmeyen bazı basit ardışık problemler de bu tip akış kullanabilir.
- **ÖRNEK:**
 - Klavyeden girilen iki sayıyı okuyup aritmetik ortalamasını hesaplayan ve sonucu ekrana yazan bir programın akışı yandaki şekilde gibi ifade edilebilir.



Seçme/Kontrol Yapıları

➤ **Karar yapıları**

➤ Öne sürülen koşulun doğru veya yanlış sonuç vermesine göre farklı kod bloklarını yürüten fonksiyonlardır.

➤ **if**

➤ **if / else**

➤ İç içe **if / else**

Seçme/Kontrol Yapıları...

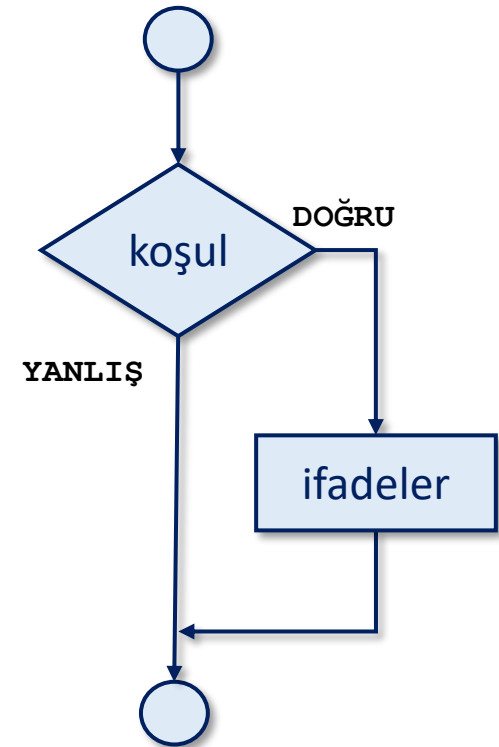
➤ if Kontrol Yapısı

➤ Koşul **boolean** bir ifadedir

➤ 1(**true**) / 0(**false**)

➤ Eğer ((not>70) && (not<80)) ise C yaz

```
if ( koşul )  
{  
    ifadeler...  
}
```



Seçme/Kontrol Yapıları...

Örnek:

➤ Klavyeden girilen iki sayıyı karşılaştırınız.

Seçme/Kontrol Yapıları...

➤ if Kontrol Yapısı...

➤ ÖRNEK: ⇒ [1]_karsilastirma.cpp

```
int sayi1, sayi2;

cout << "iki sayi giriniz: ";
cin >> sayi1 >> sayi2;

if (sayi1 < sayi2)
{
    cout << sayi1 << "\t" << sayi2 << "den kucuk...\n";
}

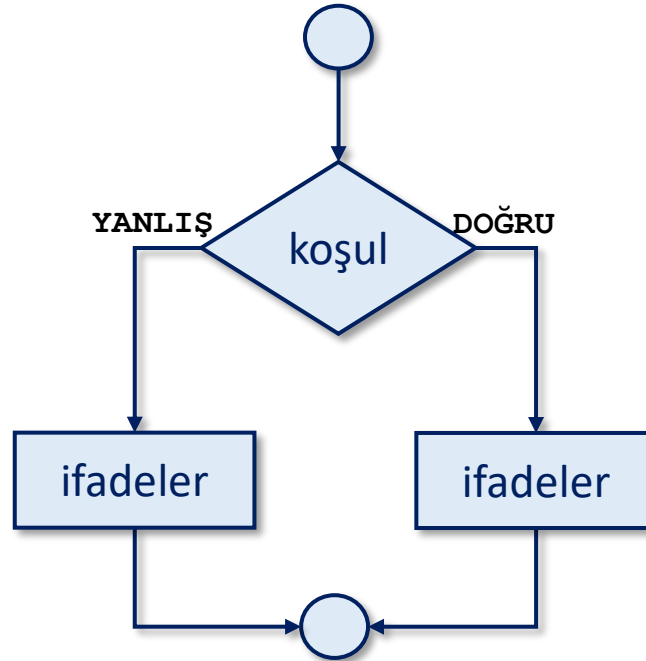
if (sayi1 > sayi2)
{
    cout << " sayi1 sayi2 den buyuk...\n";
}

if (sayi1 == sayi2)
{
    cout << " sayi1 sayi2 ye esit...\n";
}
```


Seçme/Kontrol Yapıları...

➤ if/else Kontrol Yapısı

```
if ( koşul )  
{  
    ifadeler...  
}  
else  
{  
    ifadeler...  
}
```



Seçme/Kontrol Yapıları...

➤ if/else Kontrol Yapısı

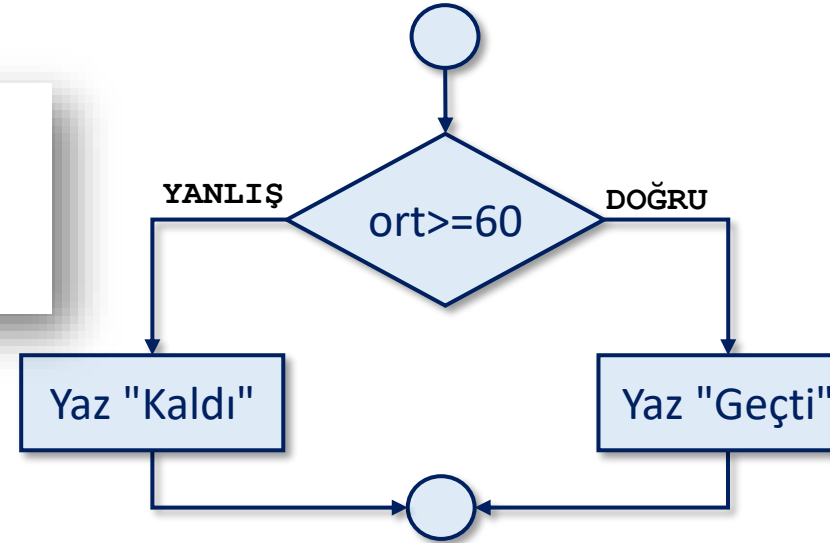
➤ if (ort >= 60)

"Geçti 😊" yaz

else

"Kaldı 😞" yaz

```
if (ort >= 60)
    cout << "Geçti";
else
    cout << "Kaldı";
```



➤ Kısa if/else operatörü (Ternary conditional operatör) (**?:**)

```
sonuc = (ort < 50 ? "Kaldi" : "Gecti");
```

Koşul

if

Doğru (true)
durum işlemi

else

Yanlış(false)
durum
işlemi

Seme/Kontrol Yapıları...

Örnek:

- Öğrencinin Vize ve Final notlarını kullanıcıdan alarak ortalama 50 üzerinde ise «Geçti» değil ise «Kaldı» yazdırınız.

Seçme/Kontrol Yapıları...

➤ if/else Kontrol Yapısı

➤ ÖRNEK:⇒ [2]_nothesapla.cpp

```
int    vize, final;

cout << "Vize ve final notunu giriniz:";
cin >> vize >> final;

float  ortalama = (vize*0.4 + final*.6);

cout << "Ortalamanız:" << ortalama << endl;

if (ortalama < 50 || final < 50)
{
    cout << "Kaldı..." << endl;
}
else
{
    cout << "Geçti..." << endl;
}
```

Seçme/Kontrol Yapıları...

Örnek:

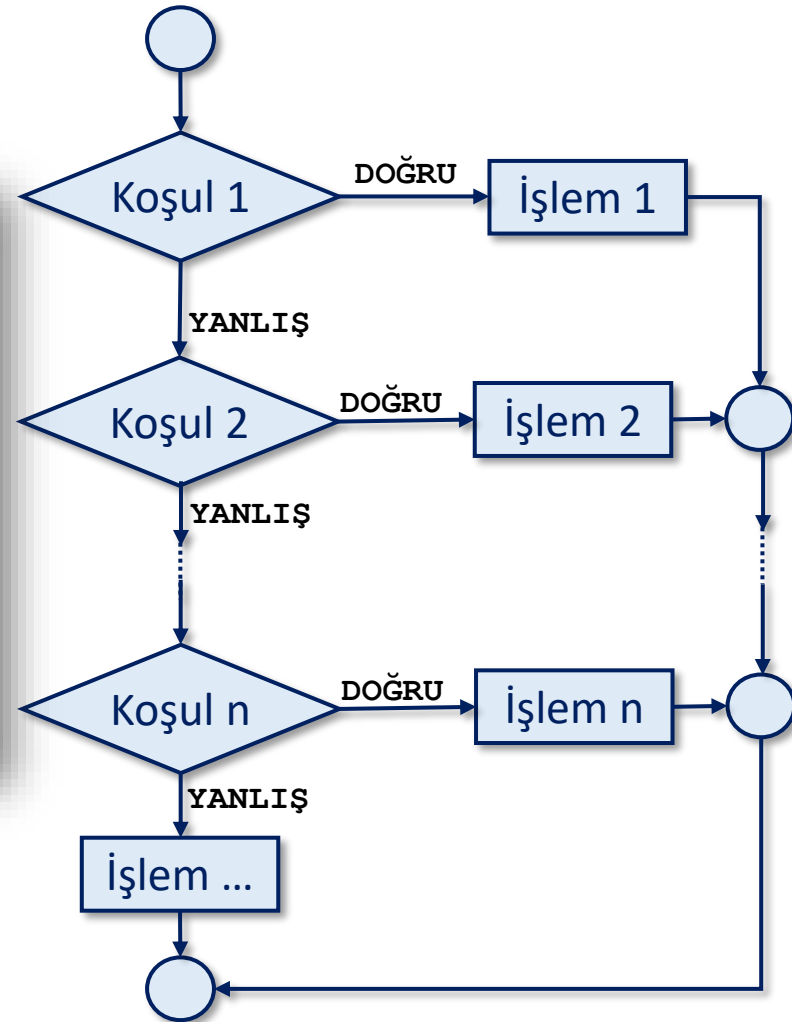
- Ortalama sonucuna göre Harf Notunu ekrana yazdıran programı yazınız.

Seçme/Kontrol Yapıları...

➤ İç içe if/else Kontrol Yapısı

➤ ÖRNEK:⇒ [3]_harfnotuhesapla.cpp

```
if (ortalama >= 90)           // 90 - 100
    cout << "A";
else if (ortalama >= 80)      // 80-89
    cout << "B";
else if (ortalama >= 70)      // 70-79
    cout << "C";
else if (ortalama >= 60)      // 60-69
    cout << "D";
else if (ortalama >= 50)      // 50-59
    cout << "E";
else                          // 0 - 49
    cout << "F";
```



Seçme/Kontrol Yapıları...

- Kullanılmayan ve tavsiye edilmeyen ancak geçerli olan **if** yapıları

- **if (a)**

- a sıfır ise **false**, diğer tüm durumlarda **true**

- ÖRNEK:

- if (0) ⇒ Her zaman YANLIŞ (false)
 - if (10) ⇒ Her zaman DOĞRU (true)
 - if (-10) ⇒ Her zaman DOĞRU (true)

Seçme/Kontrol Yapıları...

➤ **ÖRNEK:** Klavyeden okunan bir reel sayının karekökünü bulup sonucu ekrana yazan bir programı C++ programlama dili ile yazınız.

A1: Başla

A2: Oku (a)

A3: Oku (x)

A4: Oku (ϵ)

A5: $b = (a - x^2)/2x$

A6: $y = x + b$

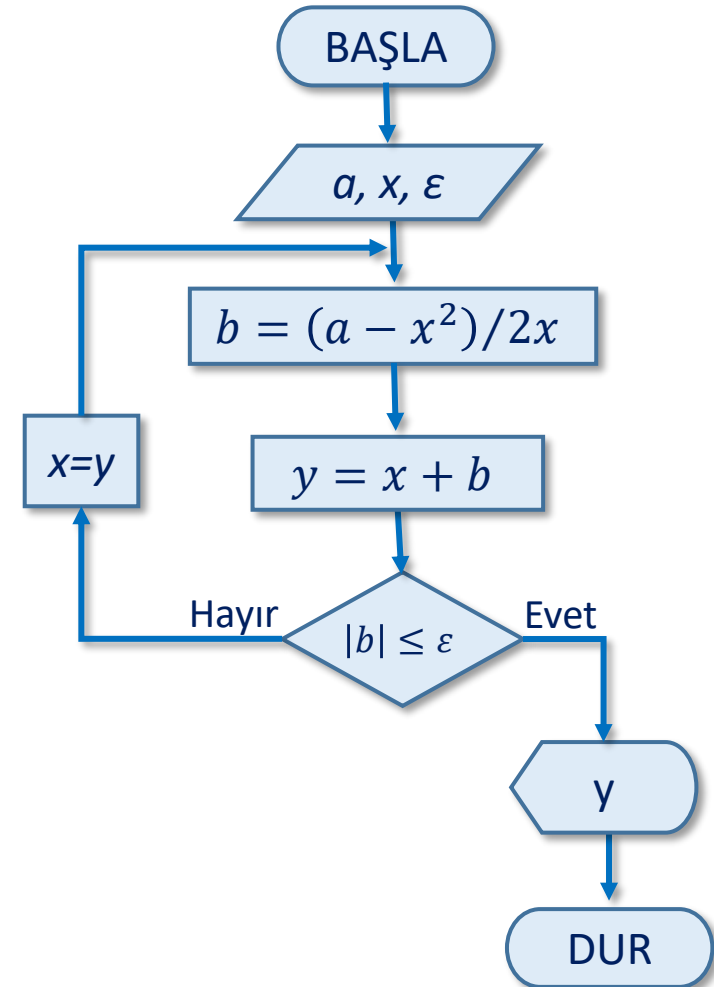
A7: Eğer $|b| \leq \epsilon$ ise
A10'a git

A8: $x = y$

A9: A5'e git

A10: Yaz (y)

A11: Dur



Seçme/Kontrol Yapıları...

➤⇒ [4_1]_karekok.cpp

```
float a, x, e, b, y;

cout << "karekökü bulunacak sayiyi giriniz: ";
cin >> a;
cout << "tahmini karakör değerini giriniz: ";
cin >> x;
cout << "kabul edilebilir hata değerini giriniz: ";
cin >> e;
A5:
b = (a - x * x) / (2 * x); // hatayi hesapla
y = x + b; // yeni karakök değeri
cout << y << endl;
if (fabs(b) <= e)
    goto A10;
else
    x = y; goto A5,
A10:
cout << "Karekök: " << y << endl; // en son hesaplanan karekök değeri
```

Seçme/Kontrol Yapıları...

➤⇒ [4_2]_karekok.cpp

```
cout << "karekökü bulunacak sayiyi giriniz: ";
cin >> a;
cout << "tahmini karakör değerini giriniz: ";
cin >> x;
cout << "kabul edilebilir hata değerini giriniz: ";
cin >> e;
do
{
    b = (a - x * x) / (2 * x);           // hatayi hesapla
    y = x + b;                          // yeni karakök değeri
    cout << endl << x;
    x = y;
} while (fabs(b) > e);

cout << "Karekök: " << y << endl;      // en son hesaplanan karekök değeri

system("pause");

return 0;
```

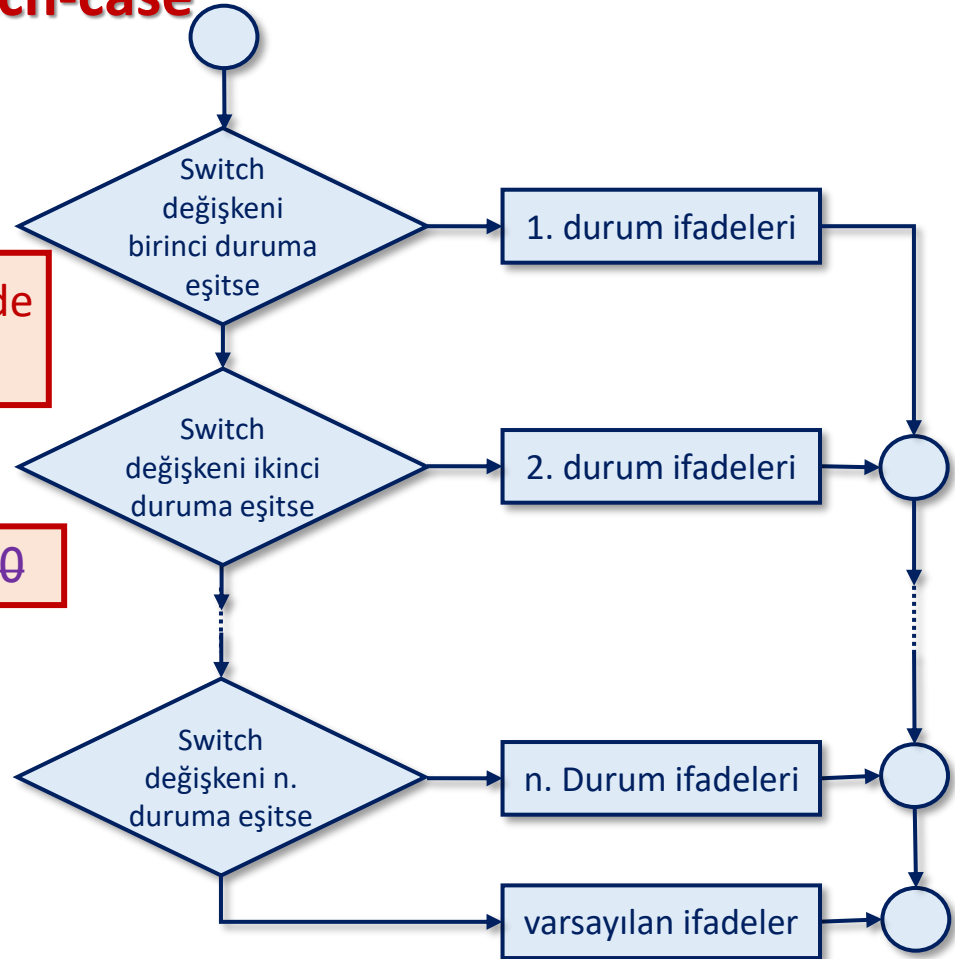
Seçme/Kontrol Yapıları...

➤ Çoklu Dallanma Yapısı → switch-case

```
switch ( değişken ) {  
  case sabit1:  
    ifadeler  
    break;  
  case sabit2:  
    ifadeler  
    break;  
  .  
  .  
  .  
  default:  
    varsayılan ifadeler...  
}
```

Tamsayı tipinde olabilir

~~Case~~ **ort < 50**



Seçme/Kontrol Yapıları...

Örnek:

- Dört işlemi gerçekleştiren hesap makinesi programını kullanıcının seçimine göre karar yapılarını kullanarak gerçekleştiriniz.

Seçme/Kontrol Yapıları...

➤ Çoklu Dallanma Yapısı → switch-case

➤ ÖRNEK: ⇒ [5]_dortislem.cpp

```
switch (islem)
{
case '+':
    cout << sayi1 << " + " << sayi2 << " = " << sayi1 + sayi2;
    break;
case '-':
    cout << sayi1 << " - " << sayi2 << " = " << sayi1 - sayi2;
    break;
case '*':
    cout << sayi1 << " * " << sayi2 << " = " << sayi1 * sayi2;
    break;
case '/':
    cout << sayi1 << " / " << sayi2 << " = " << sayi1 / sayi2;
    break;
default:
    // Kalvyeden girilen karakter dört işlemten hiç biri değilse (+, -, *, /)
    cout << "Hata!...";
    break;
}
```

Seçme/Kontrol Yapıları...

➤ Çoklu Dallanma Yapısı → switch-case...

➤ ÖRNEK: ⇒ [5]_dortislem.cpp...

➤ **switch-case** yapısı **if/else** yapısı ile de gerçekleştirilebilir.

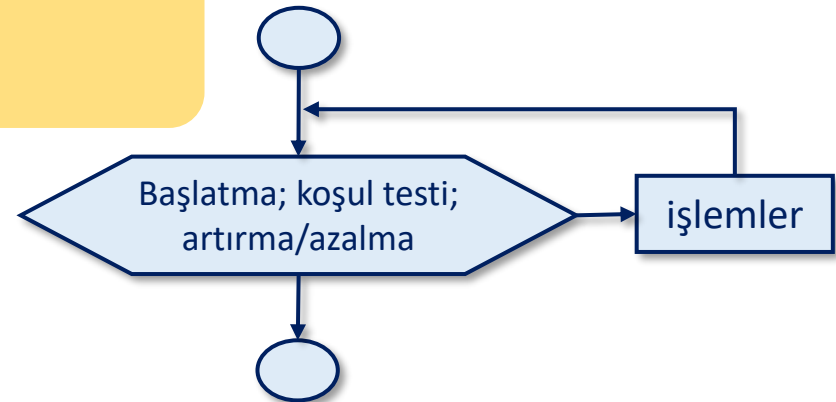
```
if (islem='+')
    cout << sayi1 << " + " << sayi2 << " = " << sayi1 + sayi2;
else if (islem = '-')
    cout << sayi1 << " - " << sayi2 << " = " << sayi1 - sayi2;
else if (islem = '*')
    cout << sayi1 << " * " << sayi2 << " = " << sayi1 * sayi2;
else if (islem = '/')
    cout << sayi1 << " / " << sayi2 << " = " << sayi1 / sayi2;
else
    cout << "Hata!...";
```

Döngü Yapısı

➤ For

- Programın bir parçasını sabit sayıda çalıştırır.
- Koşul sınaması çevrime girmeden yapılır.
- Döngüye girmeden önce sayaç başlangıç değeri alır ve daha sonra koşula bakılır.
- Döngü içerisindeki işlemler yapıldıktan sonra sayaç üçüncü parametrenin durumuna göre değiştirilir (artırılır/eksiltilir).

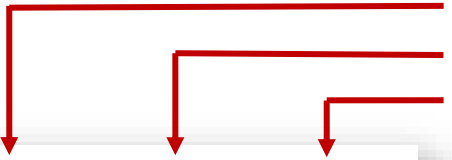
```
for ( başlatma; koşul testi; artırma/azaltma ) {  
    ifade(ler)  
}
```



Döngü Yapısı...

➤ For...

➤ Tek ifadeli **for** döngüsü



```
for (i = 0; i < 50; i++)  
    toplam++;
```

Sayaç başlangıç değeri
Koşul
Arttırma ifadesi

➤ Birden çok ifadeden oluşan **for** döngüsü

```
for (i = 0; i < 50; i++)  
{  
    sayi++;  
    toplam = toplam + i * 5;  
}
```


Seçme/Kontrol Yapıları...

Örnek:

➤ 1-100 arasındaki sayıları ve küplerini ekrana yazdırınız.

Döngü Yapısı...

➤ For...

➤ ÖRNEK: ⇒ [5]_for.cpp

```
#include <iostream>
#include <locale.h>           // Diller ve karakter setleri kütüphanesi
#include <math.h>
#include <iomanip>

using namespace std;

int main()
{
    // Sonuç ekranında Türkçe karakterleri kullanabilmek için
    setlocale(LC_ALL, "Turkish");

    int i;

    for (i = 1; i <= 100; i++)
        cout << setw(10) << left << i << setw(15) << pow(i, 3.0f) << endl;

    system("pause");

    return 0;
}
```

Seçme/Kontrol Yapıları...

Örnek:

- 1-1000 arasındaki tek ve çift sayıların sayısını, toplamını ve ortalamalarını ayrı ayrı olarak gösteren C++ kodunu yazınız.

Döngü Yapısı...

➤ For...

➤ ÖRNEK:⇒ [6]_for_sayilar.cpp

```
int tekToplam = 0, ciftToplam = 0;
int sayacTek = 0, sayacCift = 0;

for (int i = 1; i <= 1000; i++)
{
    if (i % 2 == 0)
    {
        ciftToplam += i;
        sayacCift++;
    }
    else
    {
        tekToplam += i;
        sayacTek++;
    }
}

cout << "[0-1000] arasındaki çift sayıların toplamı : " << ciftToplam << endl;
cout << "[0-1000] arasındaki tek sayıların toplamı : " << tekToplam << endl << endl;

cout << "[0-1000] arasındaki çift sayıların ortalaması : " << ciftToplam / sayacCift << endl;
cout << "[0-1000] arasındaki tek sayıların ortalaması : " << tekToplam / sayacTek << endl;
```

Döngü Yapısı...

➤ For...

➤ Kullanım Örnekleri ve Özel Durumlar

➤ `for (int i=1; i<=10; i++)`

➤ `for (int i=1; i<=10; i+=2)`

➤ `for (int i=10; i<=50; i++)`

➤ `for (int i=50; i<=10; i--)`

➤ `for (; i<=10; i++)` // başlangıç değeri daha önce atanmış olmalı

➤ `for (int i=2; i<=10;)` // i döngü sayacı döngü içinde değiştirilmeli

➤ `for (int i = 0, j = 0; j + i <= 10; j++, i++)`

Döngü Yapısı...

➤ For...

➤ Kullanım Örnekleri ve Özel Durumlar...

- `for (;;)` // sonsuz döngü
- `for (;1;)` // sonsuz döngü – 0'dan farklı bir sayı (1'in bir özelliği yok)
- `for (;-9;)` // sonsuz döngü – 0'dan farklı bir sayı (-9'un bir özelliği yok)
- `for (;0;)` // girilmez döngü
- `for (;a;)` // değişkene bağlı döngü
- `for (;a=3;)` // tehlikeli form

Döngü Yapısı...

➤ For...

➤ ÖRNEK:

➤ 1'den 10'a kadar olan sayıları toplayan program

➤ ⇒ [7]_for_toplam.cpp

```
int toplam = 0;

for (int sayac = 1; sayac <= 10; sayac++)
    toplam += sayac;

cout << "Toplam = " << toplam << endl;
```

Döngü Yapısı...

➤ For...

➤ ÖRNEK:

➤ Faktöriyel hesaplayan program

➤ ⇒ [8]_for_faktoriyel.cpp

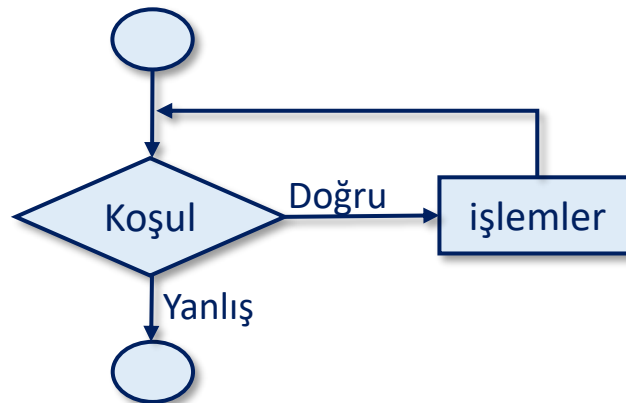
```
unsigned int sayi;  
unsigned long fakt = 1;  
  
cout << "Bir sayı girin : ";  
cin >> sayi;  
  
for (int i = sayi; i > 0; i--)  
    fakt *= i; //sayi, sayi-1, ..., 2, 1  
  
cout << sayi << "! = " << fakt << endl;
```


Döngü Yapısı...

➤ While

- For döngüsü bir işi belli bir sayıda tekrarlamaya yararırken while döngüsünde ise döngüye girmeden ne kadar tekrarlamamanın yapılacağı bilinmez.
- Bu döngüde de koşul sınaması çevrime girmeden yapılır.
- Koşul tek bir karşılaştırmadan oluşabileceği gibi birden çok koşulun mantıksal operatörler ile birleştirilmesi ile de oluşturulabilir.

```
while ( koşul ) {  
    ifade(ler)  
}
```



Koşul

```
while (i <= 10)  
{  
    toplam += i;  
    i++;  
}
```

Döngü Yapısı...

➤ While...

➤ ÖRNEK: ⇒ [9]_while.cpp

```
int i = 1;

// while döngüsü
while (i <= 100)
{
    cout << setw(10) << left << i << setw(15) << pow(i, 3.0f) << endl;
    i++;
}
```

Döngü Yapısı...

➤ While...

- ÖRNEK: Klavyeden girilen tek ve çift sayıları toplayıp toplamalarını ve ortalamasını ekrana yazdıran program

➤ ⇒ [10]_while_sayilar.cpp

```
cout << "Sayi giriniz... :";  
cin >> sayi;  
  
while (sayi > 0)  
{  
    if (sayi % 2 == 0)  
    {  
        ciftToplam += sayi;  
        sayacCift++;  
    }  
    else  
    {  
        tekToplam += sayi;  
        sayacTek++;  
    }  
  
    cout << "Sayi giriniz... :";  
    cin >> sayi;  
}
```

Döngü Yapısı...

➤ While...

- ÖRNEK: Dördüncü kuvvet değeri 10000'den küçük olan sayıları ve kuvvet değerlerini ekrana yazdıran program

➤ ⇒ [11]_while_kok.cpp

```
int kuvvet = 1;
int sayi = 1;

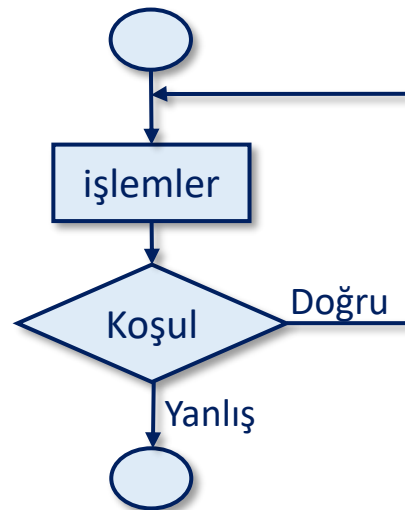
while (kuvvet < 10000)
{
    cout << setw(2) << sayi;
    cout << setw(5) << kuvvet << endl;
    ++sayi;
    kuvvet = sayi * sayi*sayi*sayi;
}
```

Döngü Yapısı...

➤ Do-while

- Diğer döngüler gibi aynı işlemleri birçok kez tekrarlamak için kullanılır.
- Farklı olarak, bu döngüde koşul sınaması yapılmadan çevrime girilir ve işlem kümesi en az bir kere işletilir. Bu deyim yapısında da koşul sağlandığı sürece çevrim tekrarlanır.
- Koşul tek bir karşılaştırmadan oluşabileceği gibi birden çok koşulun mantıksal operatörler ile birleştirilmesi ile de oluşturulabilir.

```
do {  
    ifade(ler)  
} while (koşul);
```



```
do  
{  
    toplam += i;  
    i++;  
} while (i <= 10);
```

↑ Koşul

Döngü Yapısı...

➤ Do-while...

➤ ÖRNEK: ⇒ [12]_dowhile.cpp

```
// do-while döngüsü
do
{
    cout << setw(10) << left << i << setw(15) << pow(i, 3.0f) << endl;
    i++;
} while (i <= 100);
```

Döngü Yapısı...

➤ Do-while...

➤ ÖRNEK: 1'den 10'a kadar olan sayıları toplayıp sonucu ekrana yazdıran program

➤ ⇒ [13]_dowhile_toplam.cpp

```
int sayac = 0, toplam = 0;           //sayac, döngü kontrol değişkeni

do
{
    toplam += sayac;
    sayac++;                         //döngü kontrol değişkeni artımı
} while (sayac <= 10);
```

Döngü Yapısı...

➤ Do-while...

- ÖRNEK: Klavyeden girilen tek ve çift sayıları toplayıp toplamalarını ve ortalamasını ekrana yazdıran program

➤ ⇒ [14]_dowhile_sayilar.cpp

```
do {  
    cout << "Sayi Giriniz: ";  
    cin >> sayi;  
  
    if (sayi > 0)  
    {  
        if (sayi % 2 == 0)  
        {  
            ciftToplam += sayi;  
            sayacCift++;  
        }  
        else  
        {  
            tekToplam += sayi;  
            sayacTek++;  
        }  
    }  
} while (sayi > 0);
```


Mantıksal İşleçler

➤ Koşul ifadeleri içerisinde kullanılırlar.

➤ **&&** (mantıksal **AND**)

➤ **||** (mantıksal **OR**)

➤ **!** (mantıksal **NOT**)

```
if (ortalama < 50 || fin < 50)
    cout << "Kaldı";
else
    cout << " Geçti";
```

continue ve break ifadeleri

➤ **continue** ifadesi

➤ **while, for, do/while**

- Döngünün kalanı atlanır
- Bir sonraki iterasyona geçilir

➤ **for**

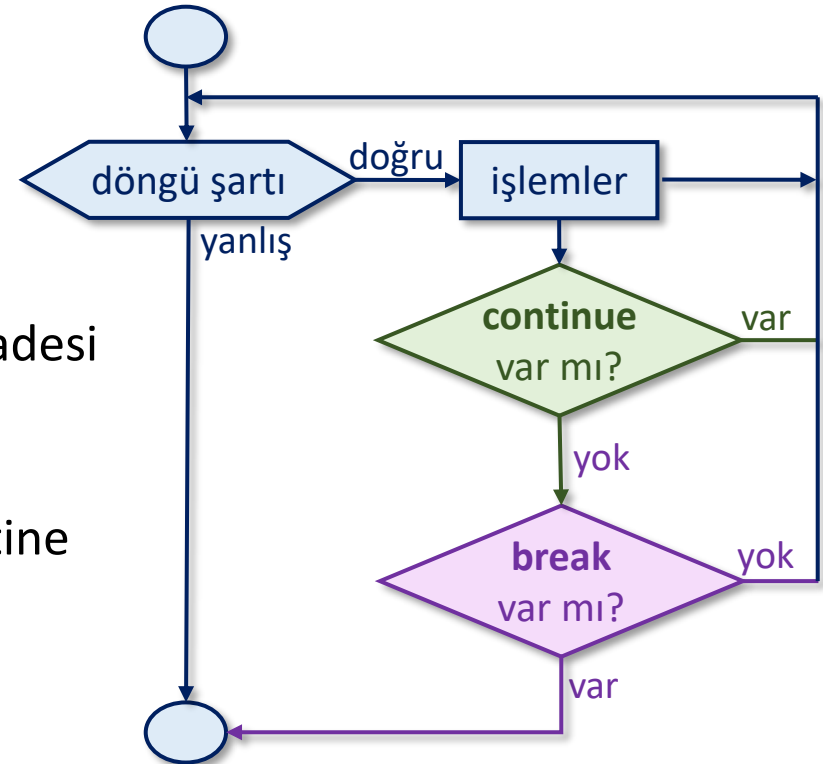
- **continue** ifadesinden sonra artırım ifadesi çalıştırılır.

➤ **while, do/while**

- **continue** ifadesinden sonra koşul testine gidilir.

➤ **break** ifadesi

- Döngüden çıkılır



continue ve break ifadeleri

➤ **continue** ifadesi

➤ ÖRNEK: ⇒ [15]_continue.cpp

```
int i = 0;

for (i = 1; i <= 10; i++)
{
    if (i == 5)
        continue;
    cout << setw(10) << left << i << setw(15) << pow(i, 3.0f) << endl;
}

cout << "i = " << i << endl;
```

```
1          1
2          8
3         27
4         64
5         216
6         343
7         512
8         729
9        1000
10         1000
i = 11
Press any key to continue . . .
```

continue ve break ifadeleri

➤ break ifadesi

➤ ÖRNEK: ⇒ [16]_break.cpp

```
int i = 0;

for (i = 1; i <= 10; i++)
{
    if (i == 5)
        break;
    cout << setw(10) << left << i << setw(15) << pow(i, 3.0f) << endl;
}

cout << "i = " << i << endl;
```

```
1      1
2      8
3     27
4     64
i = 5
Press any key to continue . . .
```

goto Deyimi ve Etiket

- Bir programın akışını herhangi bir koşula bağlı olmaksızın değiştirir.
- Program denetimi bir noktadan başka bir noktaya geçer ve oradan devam eder.
- **goto** deyiminin çalışması için **etikete** (label) ihtiyaç vardır.
- C++ dilinde tanımlı olmasına rağmen, yapısal programlama yönteminin ortaya çıkışından sonra, kullanımı kesinlikle tavsiye edilmemektedir. Programların anlaşılabilirliğini düşürmektedir (Spagetti programlama).

```
goto etiket
```

```
...
```

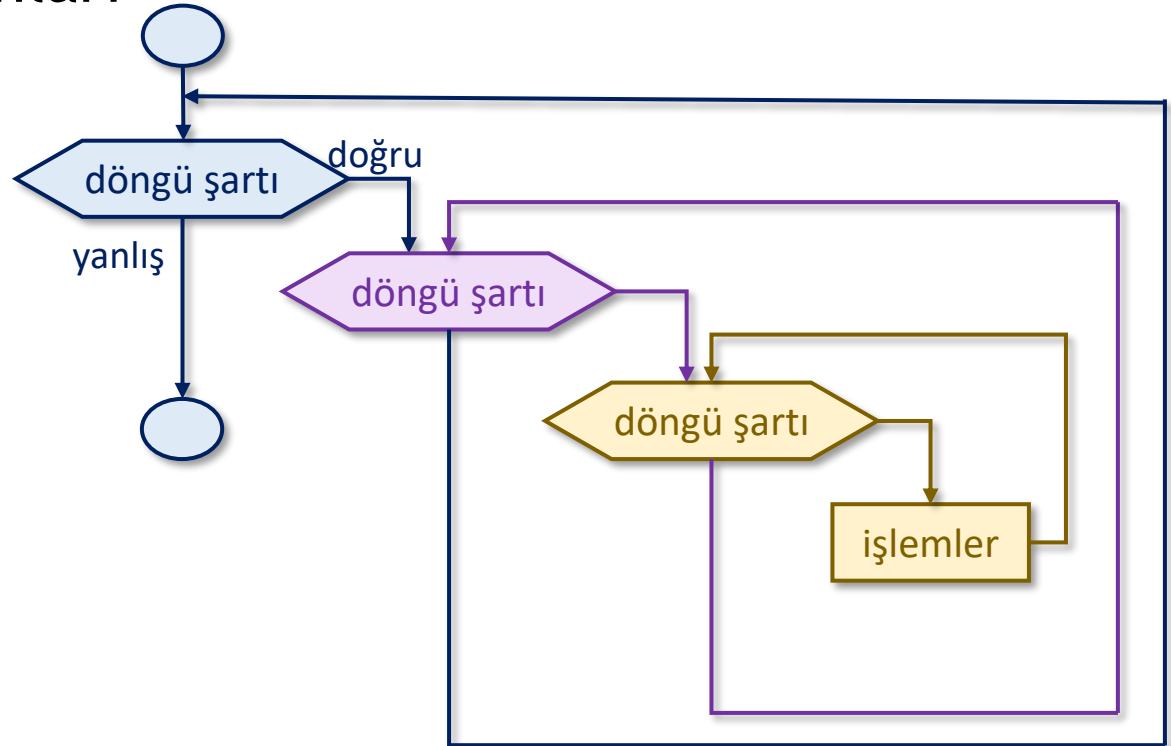
```
...
```

```
...
```

```
etiket:
```

İç İçe Döngüler

- Tüm döngüler iç-içe yapılandırılabilir
- Örnek kullanım alanları
 - Çok boyutlu diziler
 - Seri hesaplamaları
 - İlişkili döngüler
 - ...



Seçme/Kontrol Yapıları...

Örnek:

- Kullanıcı devam etmek istediği sürece yapmak istediği işlem türünü soran ve dört işlemi gerçekleştiren C++ kodunu yazınız.

İç İçe Döngüler

➤ ÖRNEK:⇒ [17]_dortislem_v2.cpp

```
do {
    cout << "Bir işlem girin (+, -, *, /): ";
    cin >> islem;

    cout << "İki sayı girin: ";
    cin >> sayi1 >> sayi2;

    switch (islem)
    {
    case '+':
        cout << sayi1 << " + " << sayi2 << " = " << sayi1 + sayi2;
        break;
    case '-':
        cout << sayi1 << " - " << sayi2 << " = " << sayi1 - sayi2;
        break;
    case '*':
        cout << sayi1 << " * " << sayi2 << " = " << sayi1 * sayi2;
        break;
    case '/':
        cout << sayi1 << " / " << sayi2 << " = " << sayi1 / sayi2;
        break;
    default:
        // Kalvyeden girilen karakter dört işlemden hiç biri değilse (+, -, *, /)
        cout << "Hata!...";
        break;
    }

    do {
        cout << "\nBaşka işlem (e/h) : ";
        cin >> ch;
    } while (!(ch == 'e' || ch == 'h'));
} while (ch != 'h');
```


ÖRNEKLER

- asal_v1.cpp
- asal_v2.cpp
- kelime_sayisi.cpp

Çalışma Soruları

- Haftanın günleri (0:Pzt, 1:Sa, 2:Çar, ...)
 - Girilen rakamı haftanın günlerine çeviren ve günü yazan programı switch-case kullanarak gerçekleştiriniz.
- 1 ile N arasındaki sayıların ortalamasını alan programı yazınız
- 1-10 arasındaki çift sayıların **continue** ifadesi kullanılarak ekrana yazılması
- Klavyeden önceden belirlenen bir çıkış tuşuna basılıncaya kadar girilen sayıların asal olup olmadığını kontrol edip sonucu ekrana yazdıran programı yazınız

Çalışma Soruları...

- Hesap makinesi programı yazınız
 - switch-case kullanımı, kullanıcı hayır diyene kadar sürekli hesaplama yapmalı, kullanıcının hatalı giriş yapmaması sağlanmalı, vs...
- Kullanıcı hayır diyene kadar girilen tüm sayıları toplayıp ortalamasını bulan programı yazınız.
- İkinci dereceden bir denklemin köklerini bulan programı yazınız.
- $y = x^2 + 2x + 5$ fonksiyonunun 0 – 4 arası integralini hesaplayan programı yazınız.
- 0-2000 arasında rasgele olarak üretilen 10 sayıdan en büyük olanını bulan programı yazınız.

KAYNAKLAR

- Deitel, C++ How To Program, Prentice Hall
- Horstmann, C., Budd, T., Big C++, Jhon Wiley & Sons, Inc.
- Robert Lafore, Object Oriented Programming in C++, Macmillan Computer Publishing