Programlamaya Giriş
Karar Yapıları, Tekrarlı İfadeler(Döngüler)

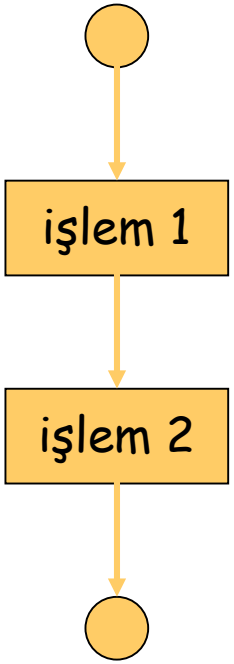
Konular

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

Program Blokları

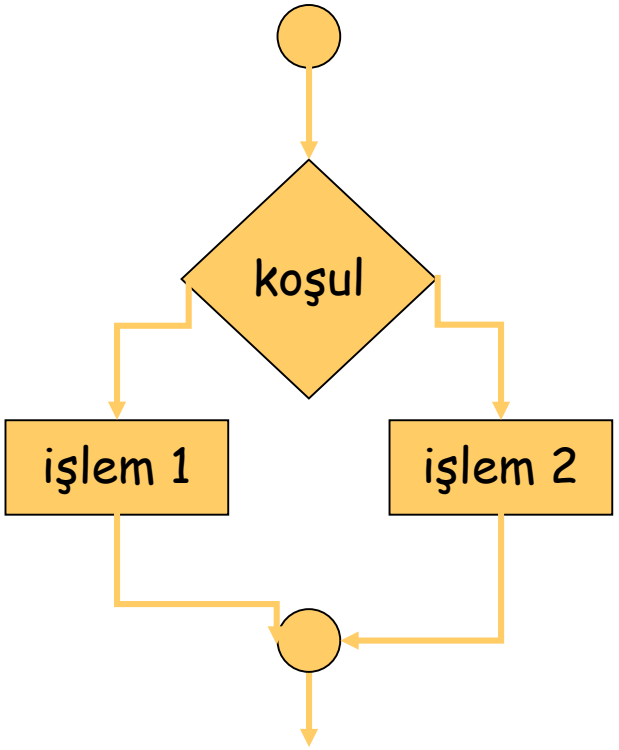
Programlar 3 bloktan oluşur:

- sıralı



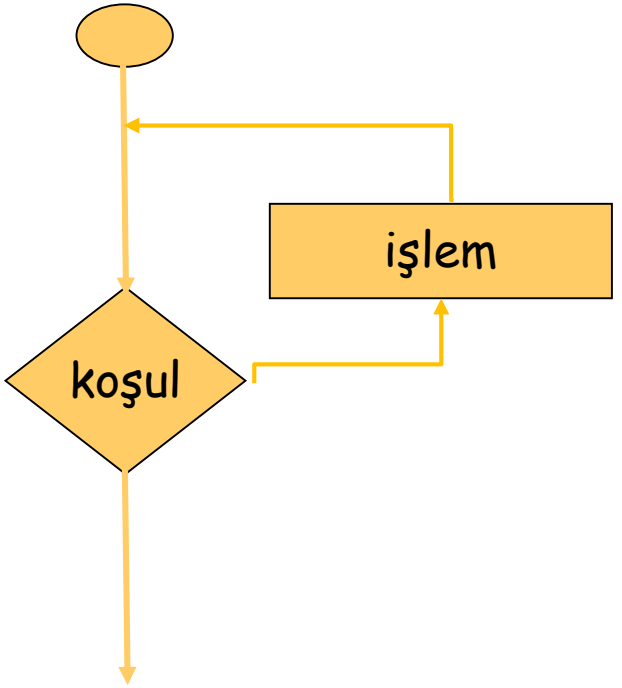
Bir dizi işlem birbiri ardından sırayla gerçekleştirilir.

- seçme



İki seçenekten hangisinin izleneceği koşula bağlıdır.

- döngü (tekrar)



Koşul sağlanana kadar işlemin tekrar ettiği durumdur

Sıra

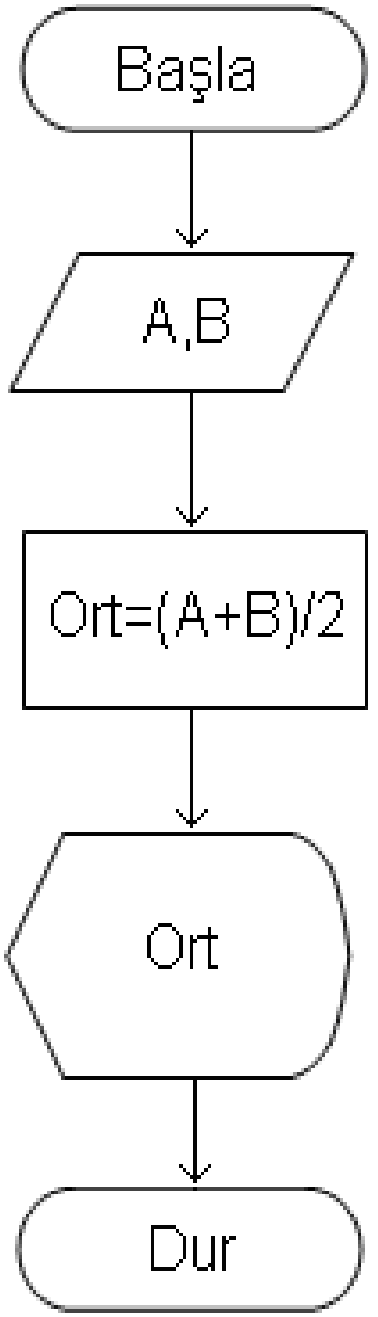
En basit akış şeması ifadeleri, bir dizi işlemin birbiri ardından sırasıyla yapılmasını şeklinde olan akış ifadeleridir. Bu tip akışlar oldukça yalın ve basittir.

Bu tarz akışlar genelde bir problemin bir parçasını çözümlemek ve ifade etmek için kullanılır.

Sorgu ve tekrar gerektirmeyen bazı basit ardışık problemler de bu tip akış kullanabilir.

✓ Örnek:

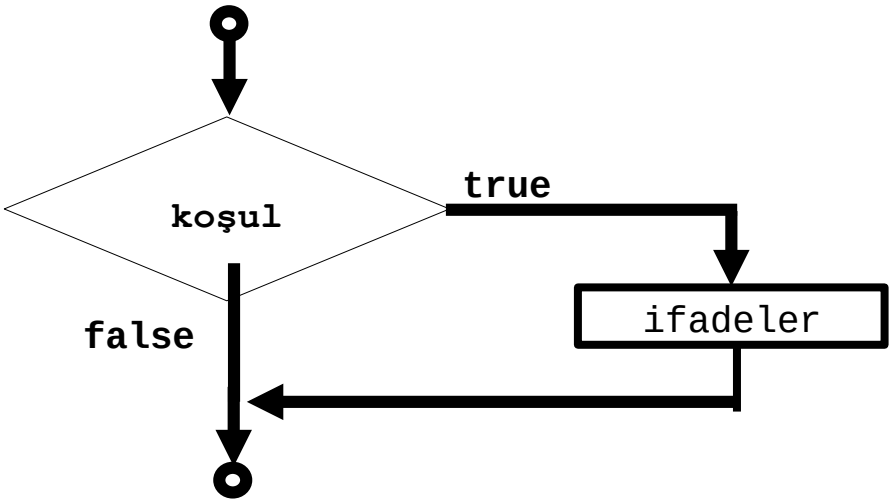
Klavyeden girilen iki sayıyı okuyup aritmetik ortalamasını hesaplayan ve sonucu ekrana yazan bir programın akışı yandaki şekilde gibi ifade edilebilir.



Kontrol Yapıları

Karar Yapıları (İf)

Karar yapıları öne sürülen koşulun doğru veya yanlış sonuç vermesine göre farklı kod bloklarını yürüten fonksiyonlardır.



Koşul boolean bir ifadedir

(1(true), 0(false))

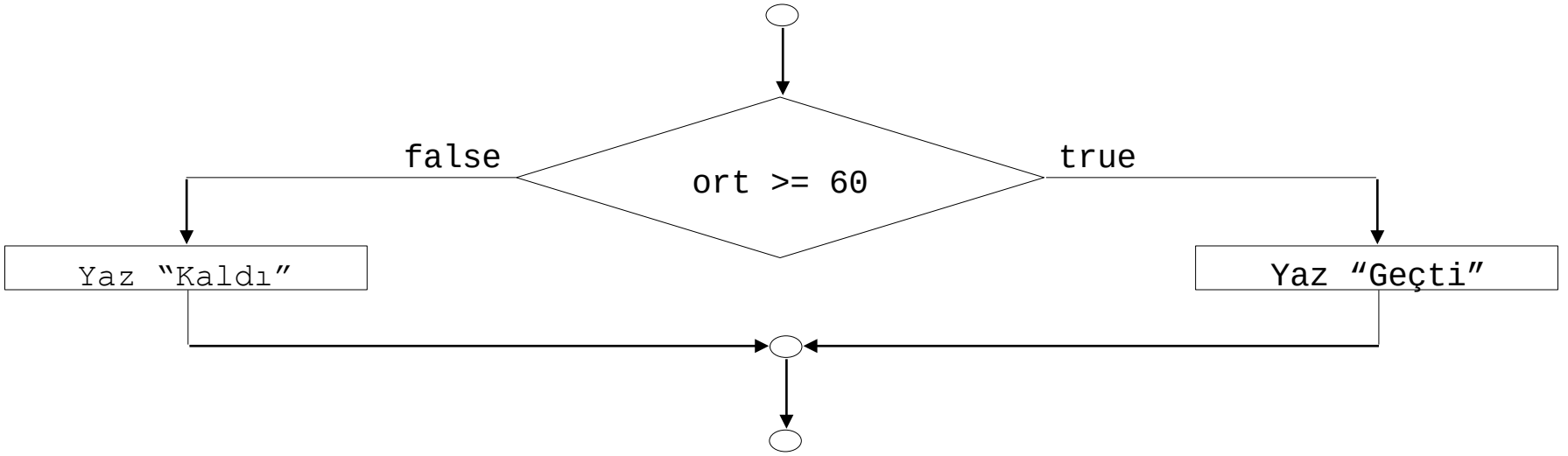
```
if ( koşul )
   {
   ifadeler..
    }
```

İki sayı girilecek ve bunların birbirleriyle ilişkisi ekrana yazdırılacak.

not>70 && not<80 ise C yaz

Karsilatirma.cpp

Kontrol Yapıları (İf/Else)




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

- ✓ Kısa if/else komutu (Ternary conditional operatör) (**?:**)

```
Sonuc= ( not >= 60 ? "Geçti" : "Kaldı" );
```

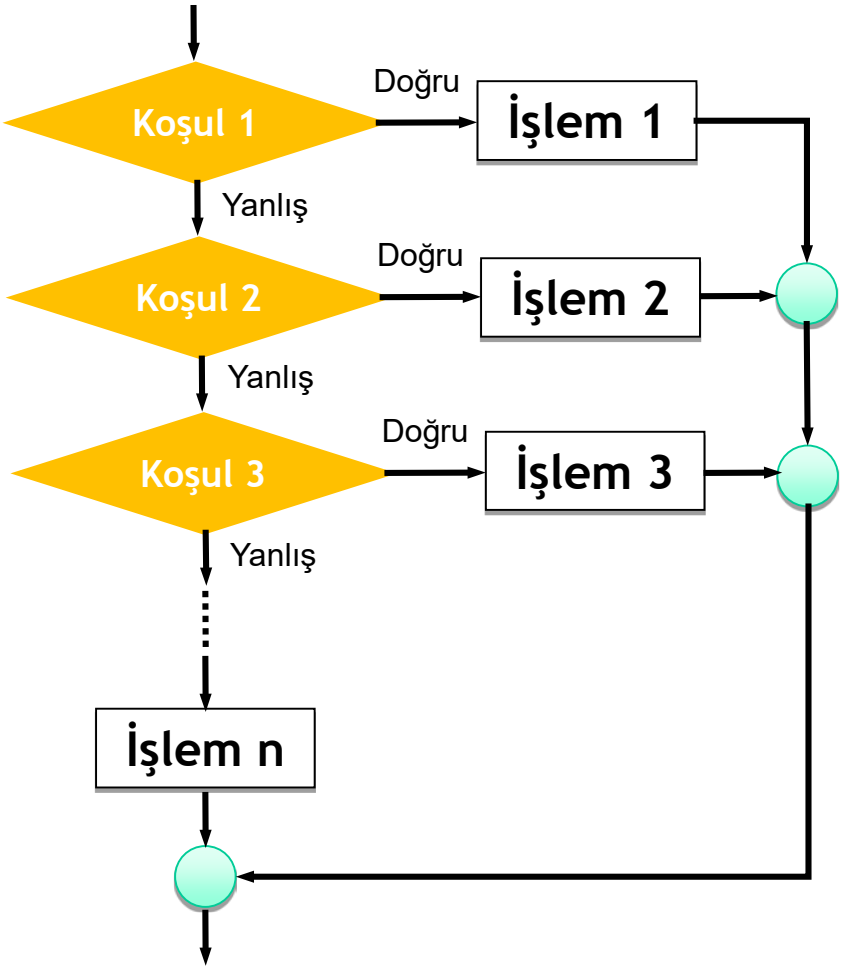
Koşul — Doğru (true) durum işlemi — Yanlış(false) durum işlemi

if — else

NotHesapla.cpp

Kontrol Yapıları (İç İçe İf/Else)

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



HarfDonusum.cpp

Kontrol Yapıları
İf/else Özel Yapıları

Kullanılmaları tavsiye edilmeyen ancak geçerli olan if yapıları

if (0) :Her zaman YANLIŞ (false)

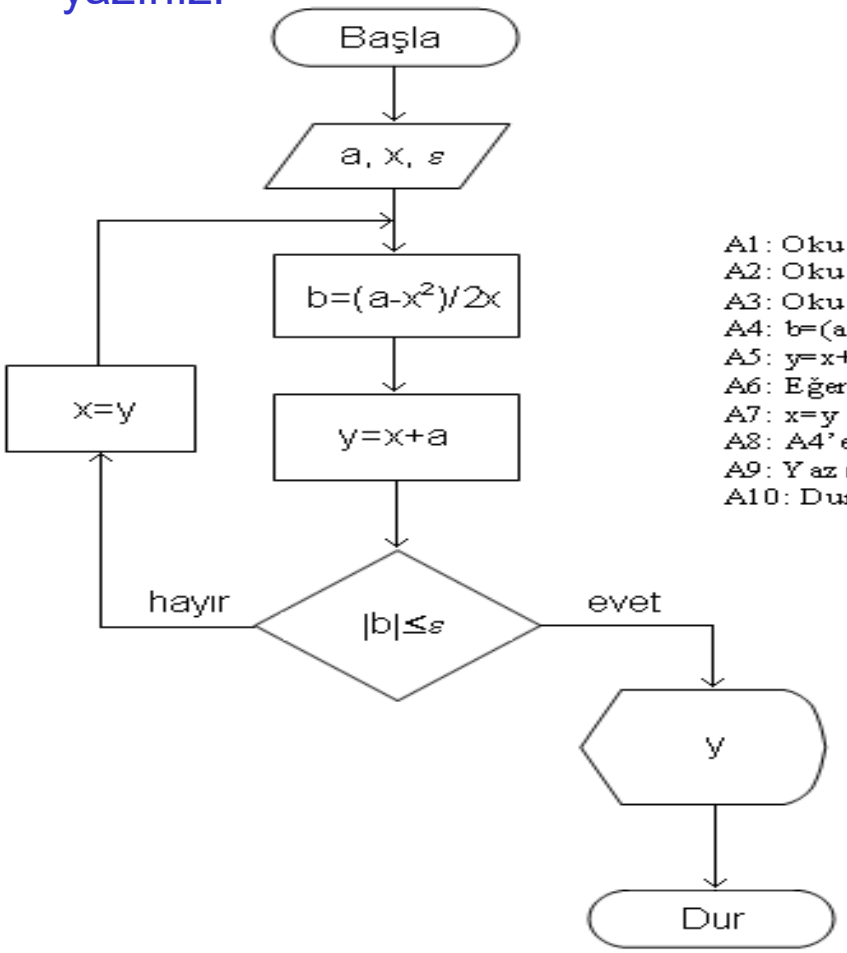
if (10) :Her zaman DOĞRU (true)

if (-10) :Her zaman DOĞRU (true)

if (a) : a sıfır ise false, diğer tüm durumlarda true

Örnek

✓Klavyeden okunan bir reel sayının karekökünü bulup sonucu ekrana yazan bir programı C++ programlama dili ile yazınız.



A1: Oku(a)
A2: Oku(x)
A3: Oku(ε)
A4: b=(a-x²)/2x
A5: y=x+a
A6: Eğer |b|≤ε ise A8'e git
A7: x=y
A8: A4'e git
A9: Yaz(y)
A10: Dur

```
// goto deyimi kullanılarak kodlanması
#include <iostream>
#include <conio.h>
#include <math.h>
using namespace std;

int main()
{
   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;
  A4:
   b=(a-x*x)/(2*x);   // hatayi hesapla
   y=x+b;     // yeni karakök değeri
   if (fabs(b)<=e)
                goto A9;
else
        x=y;goto A4;
A9:
        cout<<"karekök: "<<y;   // en son hesaplanan
                                // karekök değeri

    char ch=getch();
   return 0;
}
```

KotuOrnek.cpp **DahalyiOrnek.cpp**

```
//============================================================================
// Name        : KotuOrnek.cpp
// Author      :
// Version     : v.1.1
// Copyright   : Your copyright notice
// Description : Karekök Bulma // goto deyimi kullanılarak kodlanması (Kötü bir kodlama örneği)
//============================================================================

#include <iostream>
#include <math.h>

using namespace std;

int main()
{
   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;
   A4:
   b=(a-x*x)/(2*x);  // hatayi hesapla
   y=x+b;     // yeni karakök değeri
   cout<<endl<<y;
   if (fabs(b)<=e)
       goto A9;
    else
        x=y;goto A4;
    A9:
        cout<<"karekök: "<<y;  // en son hesaplanan
                               // karekök değeri
    cin.get();cin.get(); // Akışı duraklatma
   return 0;
}
```

KotuOrnek.cpp **DahalyiOrnek.cpp**

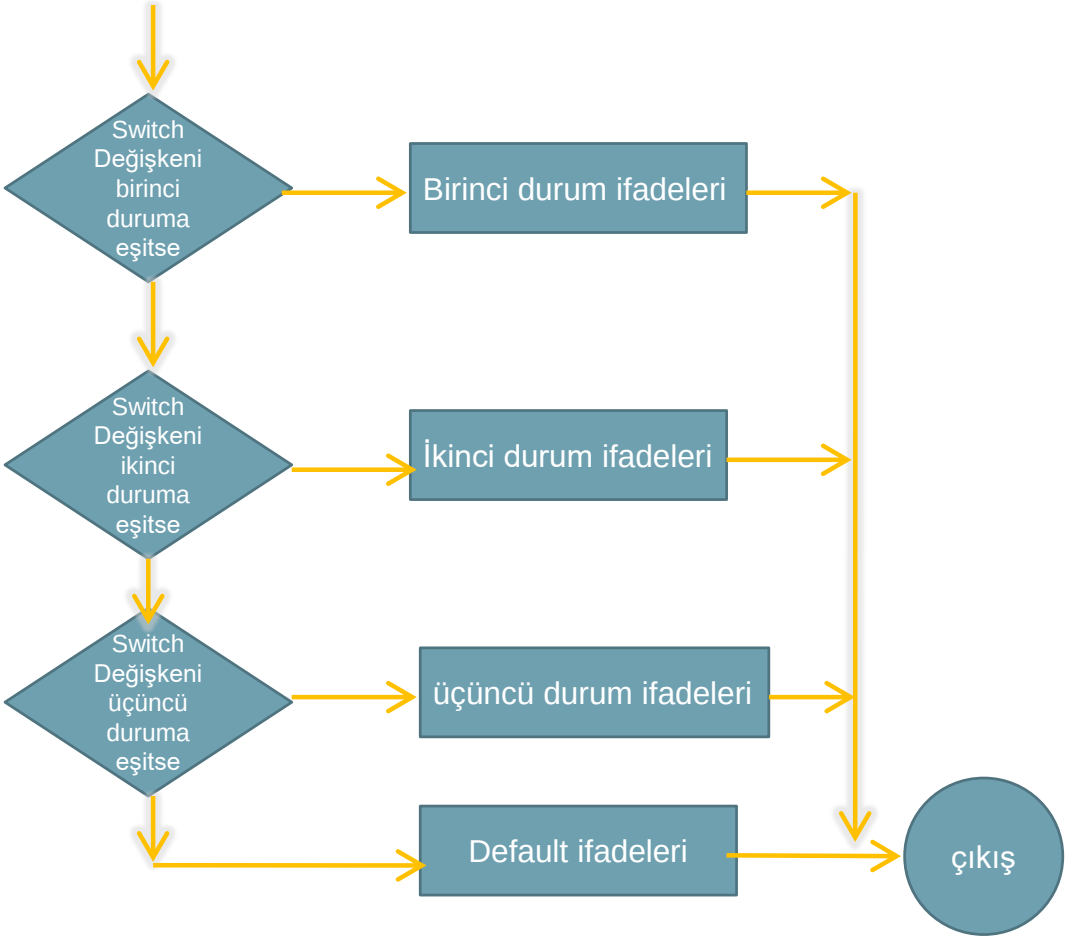
```
//============================================================================
// Name        : DahaIyiOrnek.cpp
// Author      :
// Version     : v.1.1
// Copyright   : Your copyright notice
// Description : Karekök Bulma // goto yerine döngü kullanımı (olması gereken budur...)
//============================================================================

#include <iostream>
#include <math.h>

using namespace std;

int main()
{
    float a,x,e,b;
    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
        x=x+b; // yeni karakök değeri
        cout<<endl<<x;
    }while(fabs(b)>e);

     cout<<endl<<"karekök: "<<x;  // en son hesaplanan

     cin.get();cin.get(); // Akışı duraklatma
    return 0;
}
```

KotuOrnek.cpp **DahaIyiOrnek.cpp**

Çoklu Dallanma Yapısı
Switch-Case

✓ Tamsayı tipinde olabilir.

```
switch (degisken) {
  case sabit1:
       ifadeler
break;
  case sabit2:
       ifadeler
break;


  .
  .
  .
  default:
       varsayılan ifadeler
  }
```



Switch-Case

- ✓ Switch-Case yapısı if/else if/else ile de gerçekleştirilebilir.

Switch-case

```
switch (x) {
  case 1:
    cout << "x is 1";
    break;
  case 2:
    cout << "x is 2";
    break;
  default:
    cout << "value of x unknown";
  }
```

İf/else karşılığı

```
if (x == 1) {
  cout << "x is 1";
  }
else if (x == 2) {
  cout << "x is 2";
  }
else {
  cout << "value of x unknown";
  }
```

~~**case ort<50 :**~~

İf (yol/zaman<80)

SonucGoruntule.cpp

Örnek

✓Klavyeden iki sayı ve aritmetik işlem sonucuna göre dört işlem gerçekleştiren hesap makinası uygulaması.

HesapMakinasi.cpp

```
// dört işlem hesap makinası
#include <iostream>
using namespace std;
int main()
{
      double sayi1, sayi2, sonuc;
      char islem;
         cout << " ilk sayi, islem, ikinci sayi: ";
         cin >> sayi1 >> islem >> sayi2;
         switch(islem)
         {
               case '+':  sonuc = sayi1 + sayi2;  break;
               case '-' :  sonuc = sayi1 - sayi2;  break;
               case '*' :  sonuc = sayi1 * sayi2;  break;
               case '/' :  sonuc = sayi1 / sayi2;  break;
               default:   sonuc = 0;
          }
          cout << "sonuc = " << sonuc;
          system ("pause");
     return 0;
}
```

Switch-Case

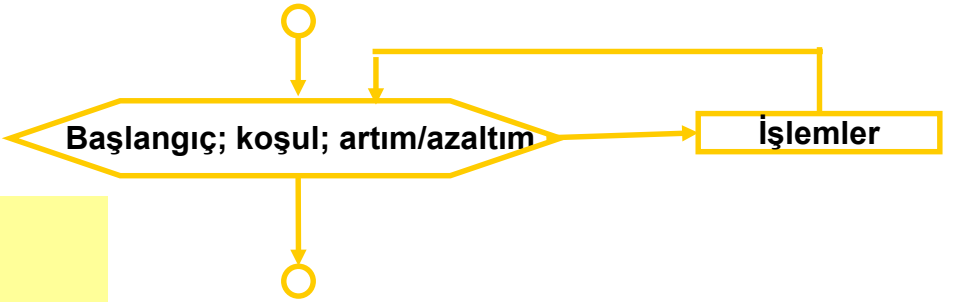
Harf dönüşüm programını geliştiriniz:
A için Mükemmel
B için Çok iyi
C için İyi
…….

Yazan kodları switch-case yapısı kullanarak ekleyiniz…

Haftanın günleri
0:pt 1: Salı 2:Çarşamba ….
…….

Girilen rakamı haftanın günlerine çeviren ve günü yazan programı switch-case kullanarak gerçekleştiriniz.

Döngüler-For



for (Başlatma; Koşul Testi; Artırma-Azaltma)
ifade;

For döngüsü, 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).

Sayaç başlangıç değeri
Koşul(sınama değeri)
Artırım ifadesi

```
for (j=0; j<15; j++)
        ifade ;
```

tek ifadeli for döngüsü

```
for (j=0; j<15; j++)
    {
        ifade;
        ifade;
        ifade;
        ......
    }
```

birden fazla ifadeden oluşan for döngüsü

For.cpp **TekCiftToplam.cpp**

Döngüler-For
Kullanım Örnekleri ve Özel Durumlar

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

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

Örnek yazılım formatları:

```
for (k=1;k<50; k+=2)
for (k=5;k<=n; k++)
for (x=50;x>10;x--)
for ( ;x<10;x++)   /* başlangıç değeri daha önce atanmış olmalı */
for (x=2;x<n; )    /* x döngü sayacı döngü içinde değiştirilmeli */
```

Sonsuz Döngüler:

```
for(;;) { }
for(;1;){ }          // Sıfırdan farklı bir sayı
for(;-9;){ }         // 9'un ve 1'in bir özelliği yok
```

Girilmez Döngü:

```
for(;0;) { }
```

Değişkene bağlı Döngü:

```
for(;a;) { }
```

Tehlikeli Form:

```
for(;a=3;)
```

Döngüler-For
Örnek

✓1 ‘den 10’a kadar sayıları toplayan program.

```
#include <iostream>
#include <conio.h>

using namespace std;

int main()
{
   int toplam=0;

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

   cout<<toplam;

   char ch=getch();
   return 0;
}
```

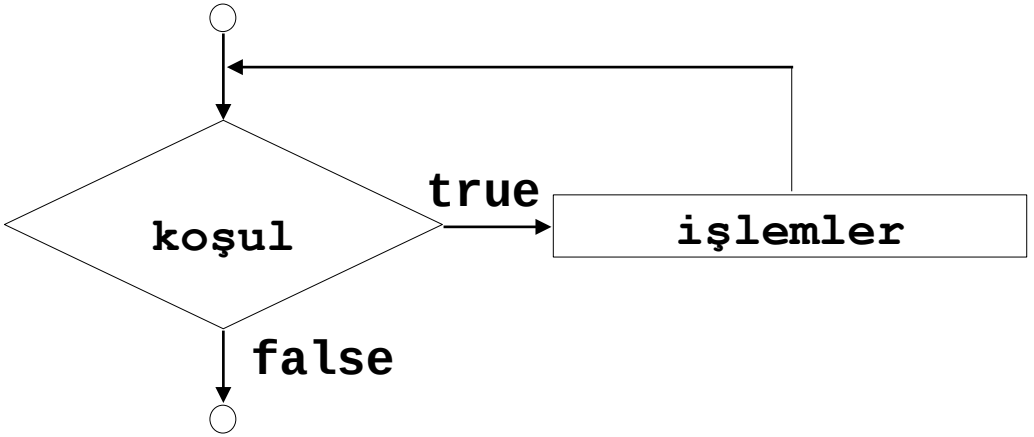
Döngüler-For
Örnek

✓Faktöriyel hesabı.

```
// faktöriyel hesaplamak için for döngüsü kullanımı

#include <iostream>
using namespace std;
int main()
{
      unsigned int sayi;
      unsigned long fakt=1;
      cout << "Sayı gir  : "; cin >> sayi; //sayıyı gir

      for(int j=sayi; j>0; j--)
         fakt *= j; //sayi, sayi-1, ..., 2, 1

      cout << "Faktöriyel  " << fakt <<" dır"<< endl;
      return 0;
}
```

Döngüler-While

```
while ( Koşul Testi){
   ifadeler…;
}
```



For döngüsü bir işi belli bir sayıda tekrarlamaya yarar.

while döngüsü ise döngüye girmeden ne kadar tekrarlamanın 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 (n!=0)
    ifade;
```
Koşul(sınama değeri)

tek ifadeli döngü

```
while (v2<45)
    {
      ifade;
      ifade;
      ifade;
      ......
    }
```
Koşul(sınama değeri)

Birden fazla ifadeli while döngüsü

While.cpp

Döngüler-While

```
#include <iostream>
#include <iomanip>                //for setw
using namespace std;

int main()
   {
   int pow=1;                    //power initially 1
   int numb=1;                    //numb goes from 1 to ???

   while( pow<10000 )            //loop while power <= 4 digits
      {
      cout << setw(2) << numb;          //display number
      cout << setw(5) << pow << endl;   //display fourth power
      ++numb;                           //get ready for next power
      pow = numb*numb*numb*numb;        //calculate fourth power
      }
   cout << endl;
   return 0;
   }
```

Döngüler-While

```
int main()
16   {
17      int total;        // sum of grades
18      int gradeCounter;  // number of grades entered
19      int grade;        // grade value
20
21      double average;    // number with decimal point for average
22
23      // initialization phase
24      total = 0;        // initialize total
1         gradeCounter = 0;  // initialize loop counter
27      // processing phase
28      // get first grade from user
29      cout << "Enter grade, -1 to end: ";  // prompt for input
30      cin >> grade;                       // read grade from user
31
32      // loop until sentinel value read from user
33      while ( grade != -1 ) {
34         total = total + grade;           // add grade to total
35         gradeCounter = gradeCounter + 1;  // increment counter
36
37         cout << "Enter grade, -1 to end: ";  // prompt for input
38         cin >> grade;                       // read next grade
39
40      } // end while
41
42      // termination phase
43      // if user entered at least one grade ...
44      if ( gradeCounter != 0 ) {
45
46         // calculate average of all grades entered
•          average = static_cast< double >( total ) / gradeCounter;
49         // display average with two digits of precision
50         cout << "Class average is " << setprecision( 2 )
51              << fixed << average << endl;
52
53      } // end if part of if/else
54
55      else // if no grades were entered, output appropriate message
56         cout << "No grades were entered" << endl;
57
58      return 0;   // indicate program ended successfully
59
60   } // end function main
```

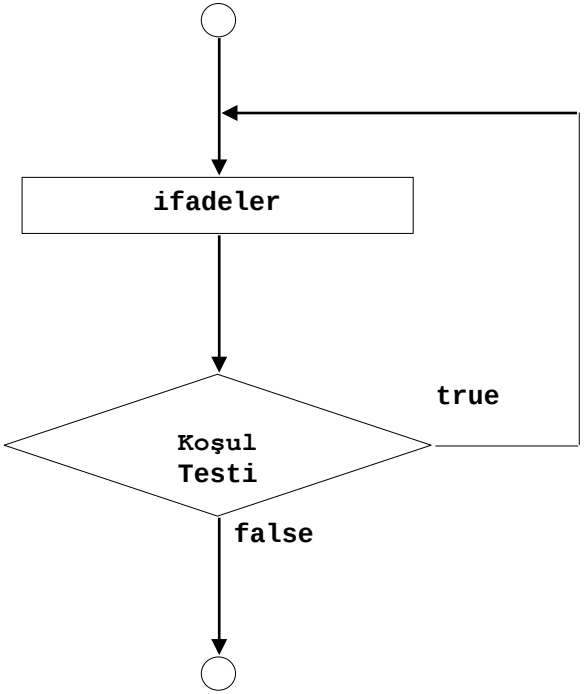
Döngüler-While
Örnek

✓Klavyeden girilen metinin kaç kelime ve harften oluştuğunu bulan program.

```
#include <iostream>
using namespace std;
#include <conio.h> // getche() için
int main()
{
      int harfsayac=0;
      int  kelimesayac=1;
      char ch = 'a'; //'\r' olmamalı
      cout << "bir cümle giriniz .: ";
      while( ch != '\r' ) //enter tusuna basana kadar dön
      {
            ch = getche(); //bir karakter oku
            if( ch==' ' ) // eğer karakter boşluk ise
                  kelimesayac++; // kelime sayısını bir artır
            else           // diğer durumda
                  harfsayac++; //harf sayısını bir artır
      }
      cout << "\n kelime sayısı=" << kelimesayac << endl;
      cout << "harf=" << (harfsayac-1) << endl;
      return 0;
}
```

Döngüler- Do - While

```
do {
   ifadeler
} while (Koşul Testi );
```



"do … while" döngüsü 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.

DoWhile.cpp

Döngüler- Do - While
Örnek

✓1 ‘den 10’a kadar sayıları toplayan program.

```
#include <iostream>
using namespace std;

int main()
{
    int sayac=0,toplam=0;        //sayac, döngü kontrol değişkeni
    do
    {
        toplam+=sayac;
        cout<<toplam<<“\t”;
        sayac++;                 //döngü kontrol değişkeni artımı
    } while(sayac<=10);          // döngü şartı kontrolü
    return 0;
}
```

Mantıksal İşleçler

Koşul ifadeleri içerisinde kullanılırlar.

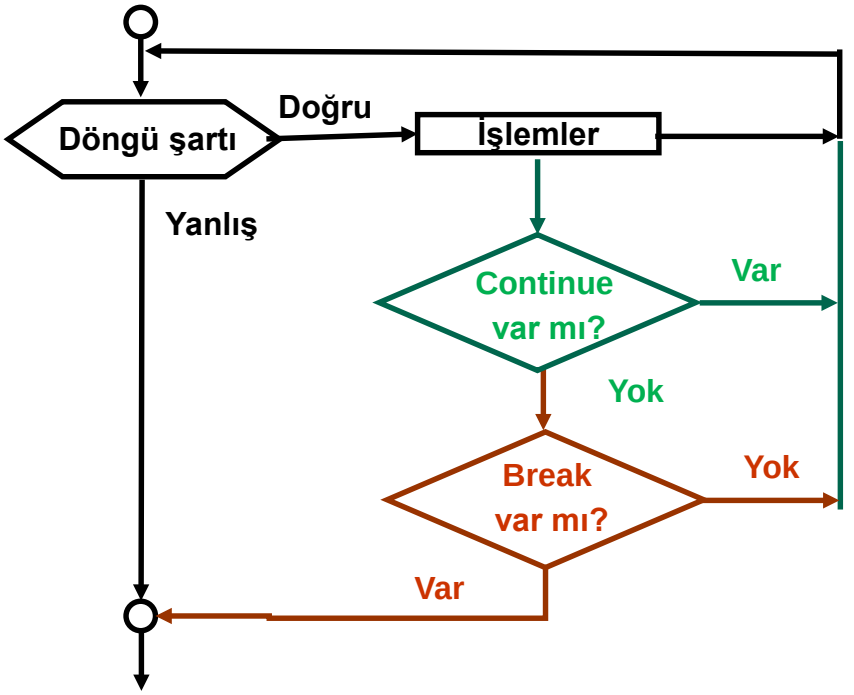
&& (logical **AND**)

|| (logical **OR**)

! (logical **NOT)**

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

Break-Continue İfadeleri

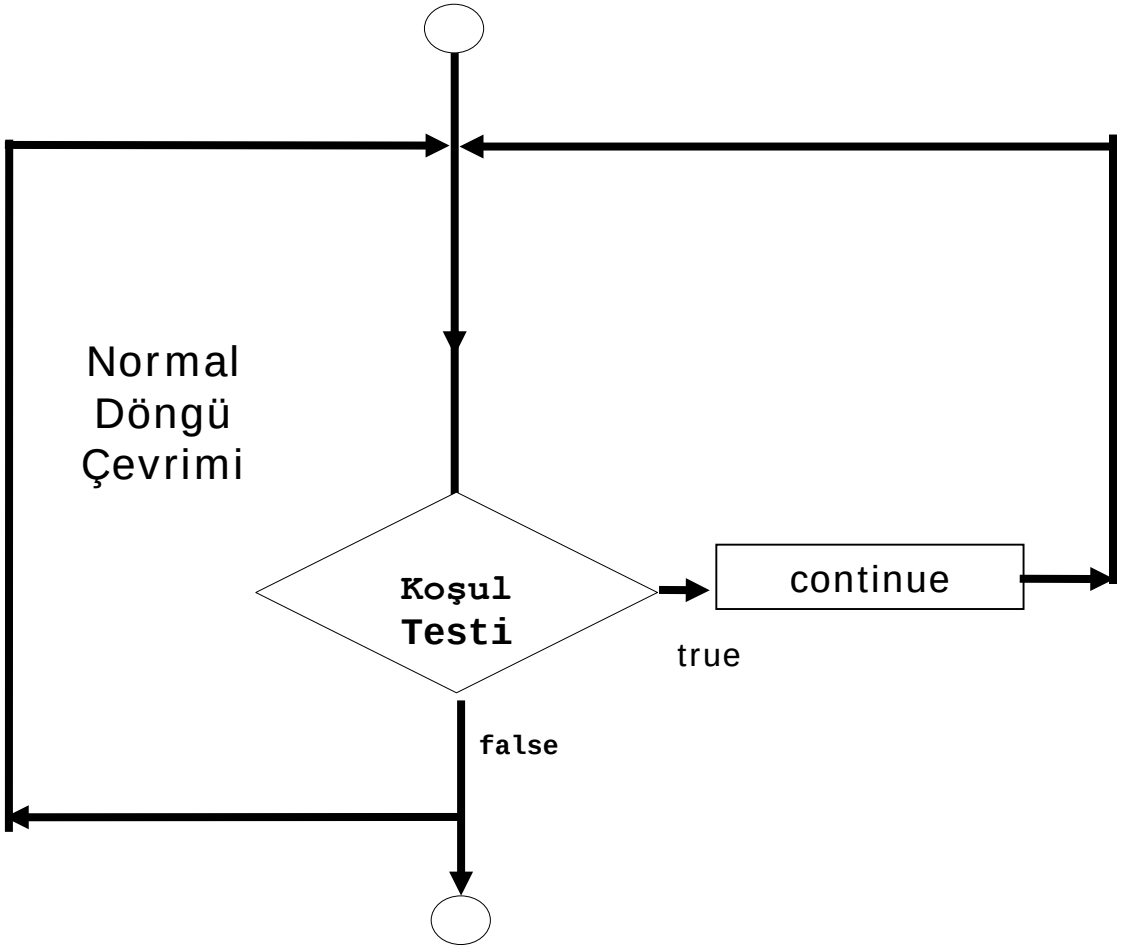


continue ifadesi:

- ✓**while**, **for**, **do/while**
  - ✓ -Döngünün kalanı atlanır
  - ✓ -Bir sonraki iterasyona geçilir
- ✓**While, do/while**
  - ✓ **continue** ifadesinden sonra koşul testine gidilir.
- ✓**for**
  - ✓ -**continue** ifadesinden sonra artırım ifadesi çalıştırılır.

Break-Continue İfadeleri

Continue



1 den 10 a kadar olan çift sayıların ekrana yazılması

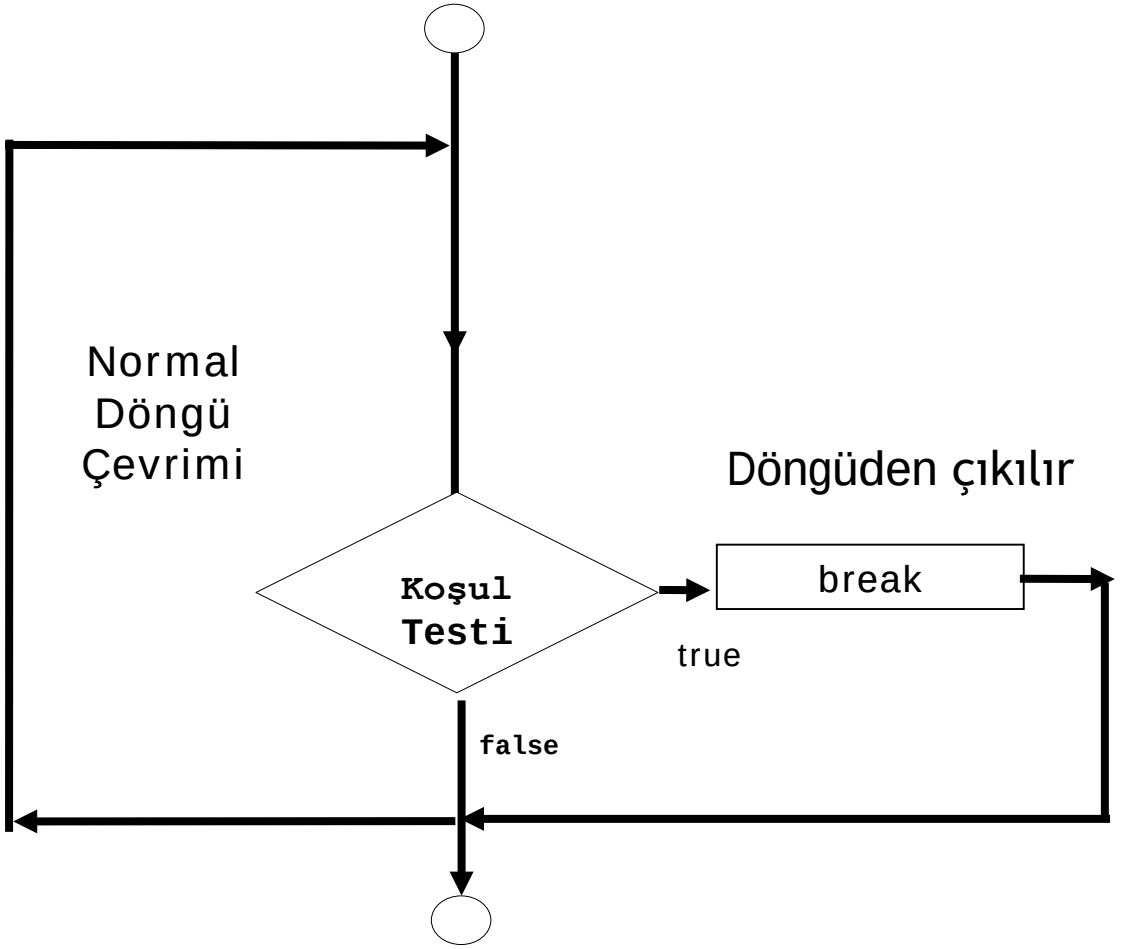
Break-Continue İfadeleri

Continue

```
3   #include <iostream>
4
5   using std::cout;
6   using std::endl;
7
8   // function main begins program execution
9   int main()
10  {
11      for ( int x = 1; x <= 10; x++ ) {
13
14       if ( x == 5 )
16         continue;
17
18      cout << x << " ";
19
20    }
21
25    return 0;
```

Break-Continue İfadeleri

Break



Break-Continue İfadeleri

Break

```
3     #include <iostream>
4
5     using std::cout;
6     using std::endl;
7
8     // function main begins program execution
9     int main()
10    {
11     for ( int x = 1; x<=10; x++ )
13
1    {
          if ( x == 5 )
16         {  break; }

18         cout << x << " ";
19
20       }
21
•     return 0;
•     }
```

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 deyimi çalışması için etikete (label) ihtiyaç vardır.

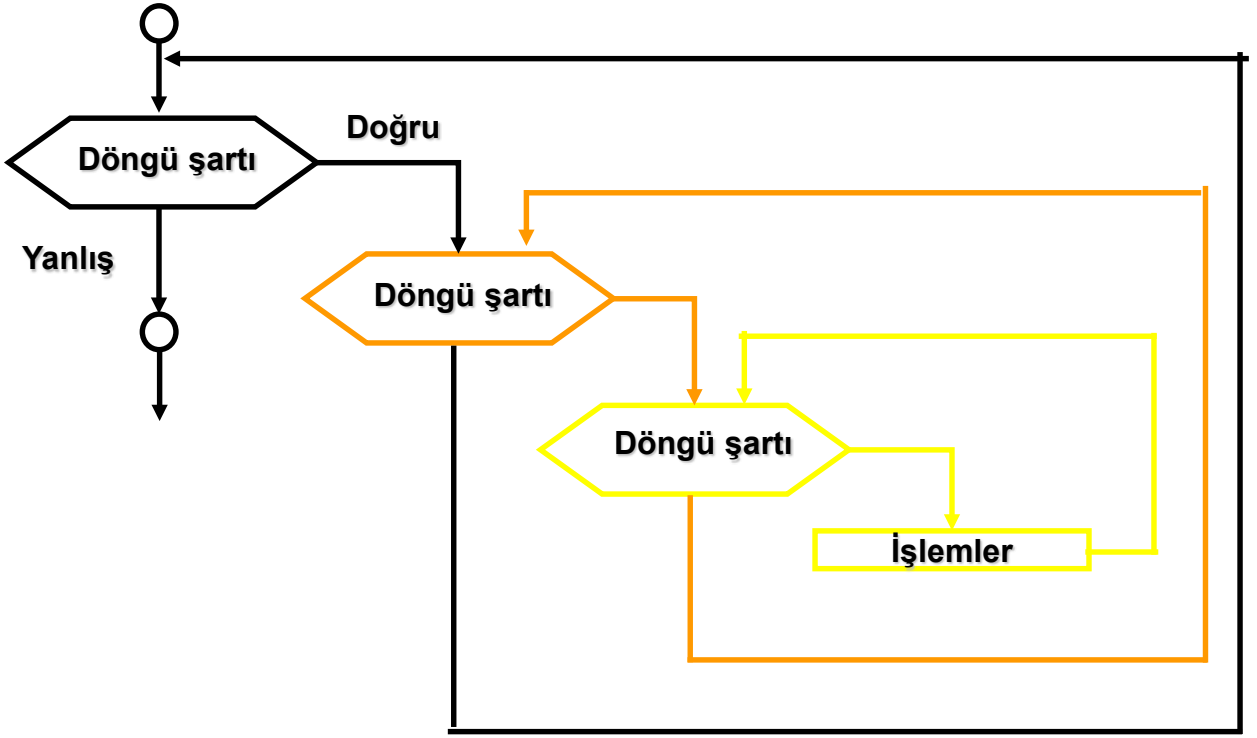
C++ dilinde tanımlı olmasına rağmen, yapısal programlama yönteminin ortaya çıkışından sonra, kullanımı kesinlikle tavsiye edilmemektedir. Programların anlaşılabilirliğini düşürmektedir (Spagetti programlama).

```
goto Etiket;
...
...
..
Etiket:
```

İç İçe Döngüler

Tüm döngüler iç-içe yapılandırılabilir

Kullanım Alanları
- Çok boyutlu dizilerde
- Seri hesaplamalarında
- İlişkili döngülerde



Örnekler

```
int main()
{
    float vize,fnl;
    char yanit='e';
    for( ;   ; )
    {
        ciz();
        cout << "Vize fnl giriniz ";
        cin >> vize>>fnl;
        float ortalama=vize*.4+fnl*.6;
        cout<<ortalama<<endl;
        ciz();
        do{
            cout<<"Devam (e/h)";
            cin>>yanit;
        }while(!((yanit=='e')||(yanit=='h')));
        if(yanit=='h')
            break;
        ciz();
    }
    return 0;
}
```

Örnekler

Yon.cpp

Asal.cpp

Hesapmakinesi.cpp

IntegralHesaplama.cpp

WhileTekCiftOrtalama.cpp

Sorular

1. 1-2000 arasındaki asal sayıları bulan program
2. Hesapmakinesi (switch-case) .. Kullanıcı hayır diyene kadar sürekli hesaplama yapmalı (Kullanıcının hatalı giriş yapamaması sağlanmalı)...
3. 1 ile N arasındaki sayıların ortalamasını alan program
4. Kullanıcı hayır diyene kadar girilen tüm sayıları toplayıp ortalamasını bulan programı yazınız...
5. İkinci dereceden bir denklemin köklerini bulan programı yazınızı...
6. y = x * x + 2 * x + 5 fonksiyonunun 0 - 4 arası integralini hesaplayan programı yazınız...
7. 0-2000 arasında rasgele olarak üretilen 500 sayıdan en büyük olanını bulan programı yazınız...

Kaynaklar

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