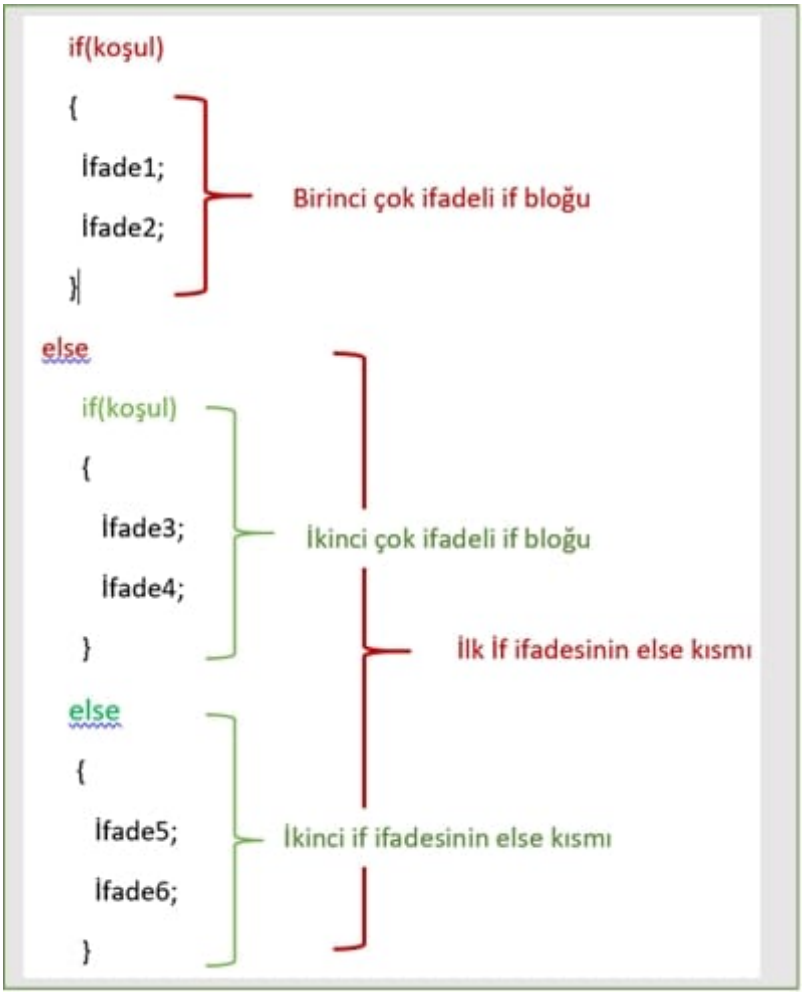
4. KÜMELENMİŞ *IF….ELSE* YAPILARI

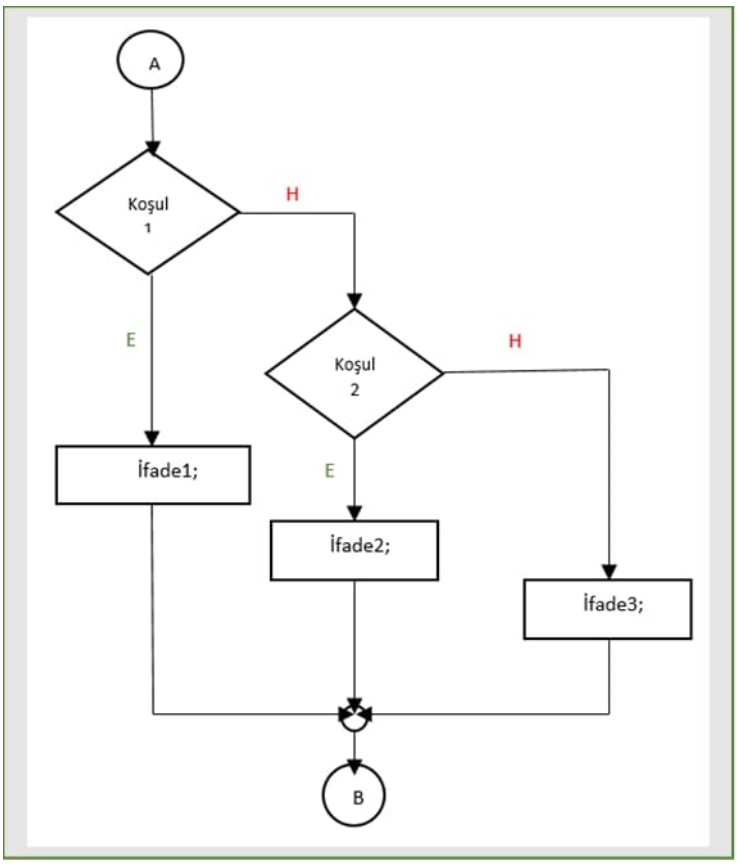
Giriş

Bölüm 3'te ***if*** ve ***if….else*** yapılarını inceledik. Hatırlanacağı gibi ***if….else*** yapısı ile program içerisinde bir koşulun, doğru ya da yanlış olmasına göre değerlendirme yapılabiliyordu. Yani sadece iki farklı durum değerlendirilebiliyordu. Ancak, bazen programlarda, koşula bağlı olarak değerlendirilmesi gereken farklı durum sayısı ikiden fazla olabilir. Örneğin klavyeden girilen üç sayıdan büyük olan sayıyı ekrana yazdıran bir C++ programı yazmamız gerektiğinde ***if….else*** yapısı problemin çözümü için yetersiz kalacaktır. Çünkü bu problemin çözümü için değerlendirilmesi gereken ikiden fazla, üç farklı durum vardır. Bu şekilde problemin çözümü için ikiden fazla durumun değerlendirilmesi gerektiğinde Kümelenmiş ***if….else*** yapıları kullanılır. **Kümelenmiş if** yapısının söz dizimi ve akış şeması aşağıda verilmiştir.



Şekil 4.1 *Kümelenmiş* ***if….else*** *yapısı*

İpucu !! *Kümelenmiş if….else yapılarında, if….else içerisine yerleştirilmiş bir if….else, onun da içerisine yerleştirilmiş bir if….else bulunur ve yapı bu şekilde devam eder. Kümelenmiş if else yapılarında ilk sınama koşulu doğru ise programın çalışması doğrudan kümelenmiş bloğun dışındaki ilk ifadeye aktarılır. İlk sınama yanlış(false) ise bir sonraki koşul test edilir. Eğer test edilen bütün koşulların sonucu yanlış(false) olursa o zaman en sondaki else içerisindeki*



Şekil 4.2 kümelenmiş if yapısının blok akış şeması

4.1. Kümelenmiş if….else Yapısı

4.1.1. Program 4.1

Üç sayıdan büyük olanı ekrana yazdıran algoritmanın akış şemasını çiziniz ve C++ programını yazınız?

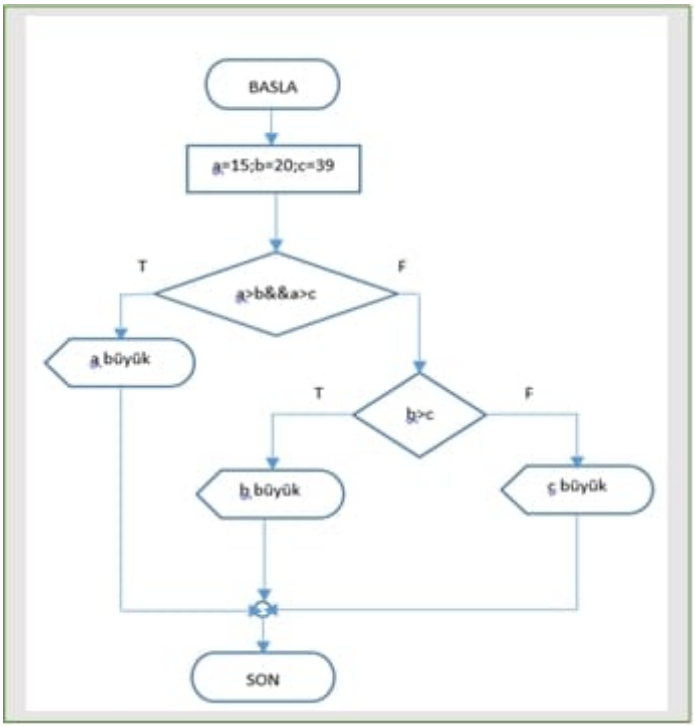
Not: *Program yazılırken a, b ve c sayılarının eşit olmadığı varsayılacaktır.*

Çözüm:

Önce **a** sayısının **b** ve **c** sayılarından büyük olmadığını kontrol edebiliriz (*birinci if*). Eğer a sayısı b ve c sayılarından büyükse program ekrana "*a büyük*" yazacak ve programın çalışması sona erecektir. Eğer a sayısı b veya c sayılarının birinden veya her ikisinden küçük ise bu defa b sayısının c sayısından büyük olup olmadığı araştırılır. Eğer b sayısı c'den büyükse bu durumda ekrana "*b büyük*" yazılır (*else if*). Son olarak, bir önceki değerlendirme sonucunda b sayısı c sayısından büyük çıkmamış ise programın else kısım çalışır ve ekrana "*c büyük*" yazılır.

İpucu !! *İç içe else yapılarında potansiyel bir tehlike vardır. Bu tehlike yanlışlıkla* ***else'i*** *yanlış* ***if*** *ile eşleme ihtimalidir.*

4.1.1.1. Program 4.1.'in Akış Şeması



Şekil 4.2 *Problem 4.1 akış şeması*

4.1.1.2. Program 4.1.'in C++ Kodu

```
#include <iostream>
#include <locale.h>
using namespace std;
int main() {
    setlocale(LC_ALL,"Turkish");
    int a=15,b=20,c=39;
    if(a>b && a>c)
    {
        cout<<"a büyük\n";
    }
    else if(b>c)

        cout<<"b büyük\n";

        else

            cout<<"c büyük\n";

        return 0;
}
```

Problem 4.1'in C++ kodu

Yukarıda verilen örneğe yakından bakıldığında programda iç içe iki ***if*** yapısı olduğu görülecektir. İkinci ***if*** yapısı birinci ***if*** yapısının bir parçasıdır. Birinci *if* yapısı çalıştırıldığında önce koşul sağlanıp sağlanmadığı kontrol edilir. Koşul saylanıyorsa buradaki ilk ifade çalıştırılır. Koşul sağlanmıyorsa else kısmından sonraki ikinci ***if*** yapısı çalıştırılır. Bu kısım yukarıdaki programda ***else if(b>c)*** ile başlar ve ***return 0;*** kadar devam eder.

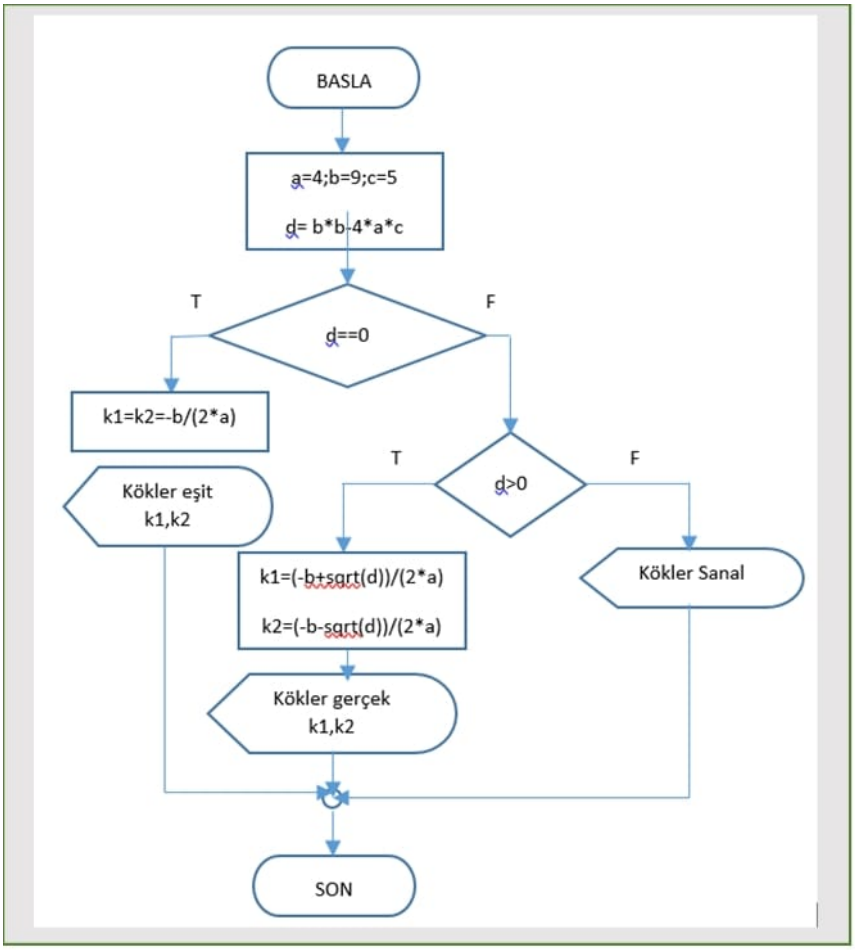
4.1.2. Program 4.2

Katsayıları verilen ikinci derece denklemin köklerini hesaplayan programı C++ 'ta yazınız?

Çözüm:

Şimdi yazacağımız program (**program 4.2**), ikinci derece bir denklemin köklerini hesaplayıp ekrana yazdıran bir program olacak. İkinci derece denklemin köklerinin hesabı **kümelenmiş if** yapılarının anlaşılması için iyi bir örnektir. Bu problemin çözümümde karekök alma işlemi yapılması gerektiği için programa *#include<math.h>* satırı eklenmiştir. Bu başlık dosyası içerisinde matematik fonksiyonları tanımlıdır. Bilindiği gibi ikinci derece bir denklemin köklerini hesaplamak için denklemdeki katsayıların bilinmesi gerekir. Programda bu katsayılar doğrudan atama yoluyla belirlenmiştir.

4.1.2.1 Program 4.2'nin akış şeması



Şekil.4.3 *Problem 4.2'nin akış şeması*

Şekil 4.3'te verilen akış şemasından da görülebileceği gibi önce denklemin diskriminant değeri hesaplanmıştır. Daha sonra diskriminant değerinin sıfıra eşit olup olmadığı test edilmiştir(*birinci if*). Bu karşılaştırmanın sonucu doğru(True) olarak elde edildiği durum için denklemin kökleri k1=k2=-b/(2*a) formülü ile hesaplanmıştır. Bulunan sonuçlar ekrana yazdırılmış ve akış if bloğunun sonuna aktarılmıştır. Bundan sonra karşılaştırmanın sonucu yanlış(False) olarak elde edildiği durum için diskriminantın sıfırdan büyük olup olmadığı test edilmiştir(ikinci if). Bu karşılaştırmanın sonucunun doğru(True) olması durumunda denklemin birbirinden farklı iki kökü vardır. Denklemin kökler k1=(-b+sqrt(d))/(2*a), k2=(-b-sqrt(d))/(2*a formülleri ile hesaplanmış hesaplanan sonuçlar ekrana yazdırılıp programın akışı if bloğunun dışına aktarılmıştır. ***else if(d>0)*** karşılaştırmasının sonucunun yanlış(false) olması durumunda denklemin kökleri sanaldır. Bu durumda programın else kısmındaki ifadeler çalışır ve ekrana "Kökler Sanal" yazılır. Programın akışı if bloğunun sonuna yönlendirilir.

4.1.2.2 Program 4.2'nin C++ Kodu

```
#include <iostream>
#include <locale.h>
#include <math.h>
using namespace std;

int main() {
    setlocale(LC_ALL,"Turkish");
    float b=9,a=4,c=5,d,k1,k2;
    d=b*b-4*a*c;
    if(d==0)
    {
        cout<<"Kökler geçek ve eşit\n";
        k1=k2=-b/2*a;
        cout<<"k1 = "<<k1<<endl;
        cout<<"k2 = "<<k2<<endl;
    }
    else if(d>0)
    {
        cout<<"Kökler geçek ve eşit değil\n";
        k1=(-b+sqrt(d))/(2*a);
        k2=(-b-sqrt(d))/(2*a);
        cout<<"k1 = "<<k1<<endl;
        cout<<"k2 = "<<k2<<endl;
    }
    else
    {
        cout<<"Kökler Sanal\n";
    }
    return 0;
}
```

Problem 4.2' ye ait C++ kodu

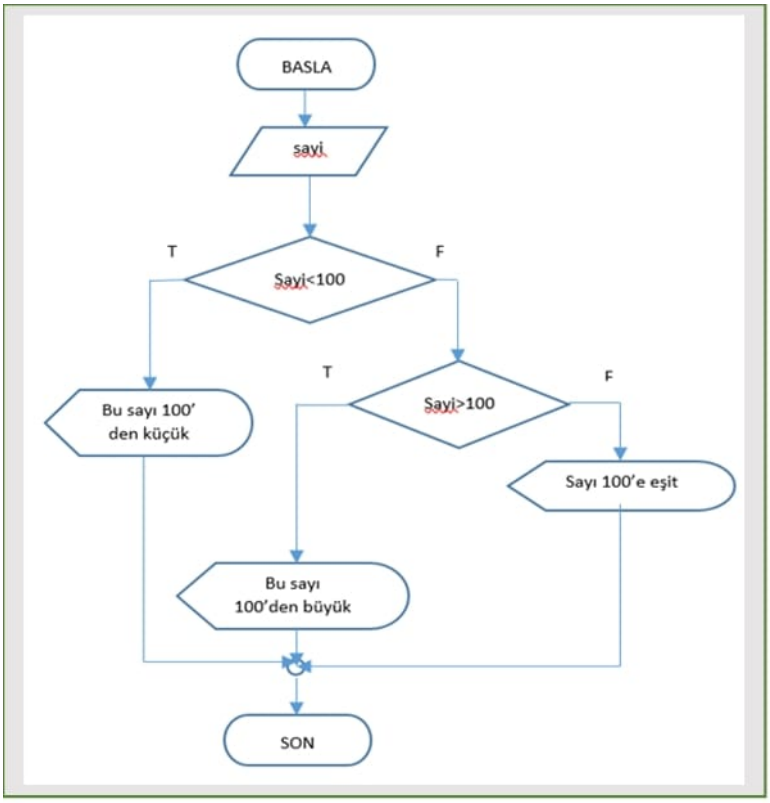
4.1.3. Program 4.3

Klavyeden girilen sayının 100'den küçük, 100'den büyük veya 100' e eşit olduğunu tespit eden ve sonucu ekrana yazdıran C++ programını yazınız?

Çözüm:

Klavyeden sayı girildikten sonra sayının 100'den küçük olup olmadığı kontrol edilecek (*birinci if*). Eğer test sonucunda sayının 100'den küçük olduğu tespit edilirse program ekrana "*Bu sayı 100'den küçük*" yazılmasını sağlayacak ve programın çalışması sona erecektir. Eğer bu ilk testin sonucunda sayının 100'den küçük olmadığı tespit edilirse sayının 100'den büyük olup olmadığı kontrol edilir(*else if*). Eğer sayı 100'den büyük ise program ekrana "*Bu sayı 100'den büyük*" yazılmasını sağlayacak ve programın çalışması sona erecektir. Bu ikinci kontrolde de sayının 100'den büyük olmadığı tespit edilirse programın else kısmı çalışacak ve program ekrana "Sayı 100'e eşit" yazılmasını sağlayacak ve programın çalışması sonlanacaktır.

4.1.3.1. Program 4.3'ün Akış Şeması



Şekil.4.4 *Problem 4.3'nin akış şeması*

4.1.3.2. Program 4.3'ün C++ Kodu

```
#include <iostream>
#include <locale.h>
using namespace std;

int main()
{
    setlocale(LC_ALL,"Turkish");
    int sayi;
    cout << "Bir sayi girin: ";
    cin >> sayi;
    if (sayi < 100)
    {
        cout << "Girdiginiz sayi: " << sayi << endl;
        cout << "Bu sayi 100'den kucuk." << endl;
    }
    else if (sayi > 100)
    {
        cout << "Girdiginiz sayi: " << sayi << endl;
        cout << "Bu sayi 100'den buyuk." << endl;
    }
    else
    {
    cout << "Girdiginiz sayi: " << sayi << endl;
    cout << "Sayılar 100'e eşit'" << endl;
    }

    system("PAUSE");
    return 0;
}
```

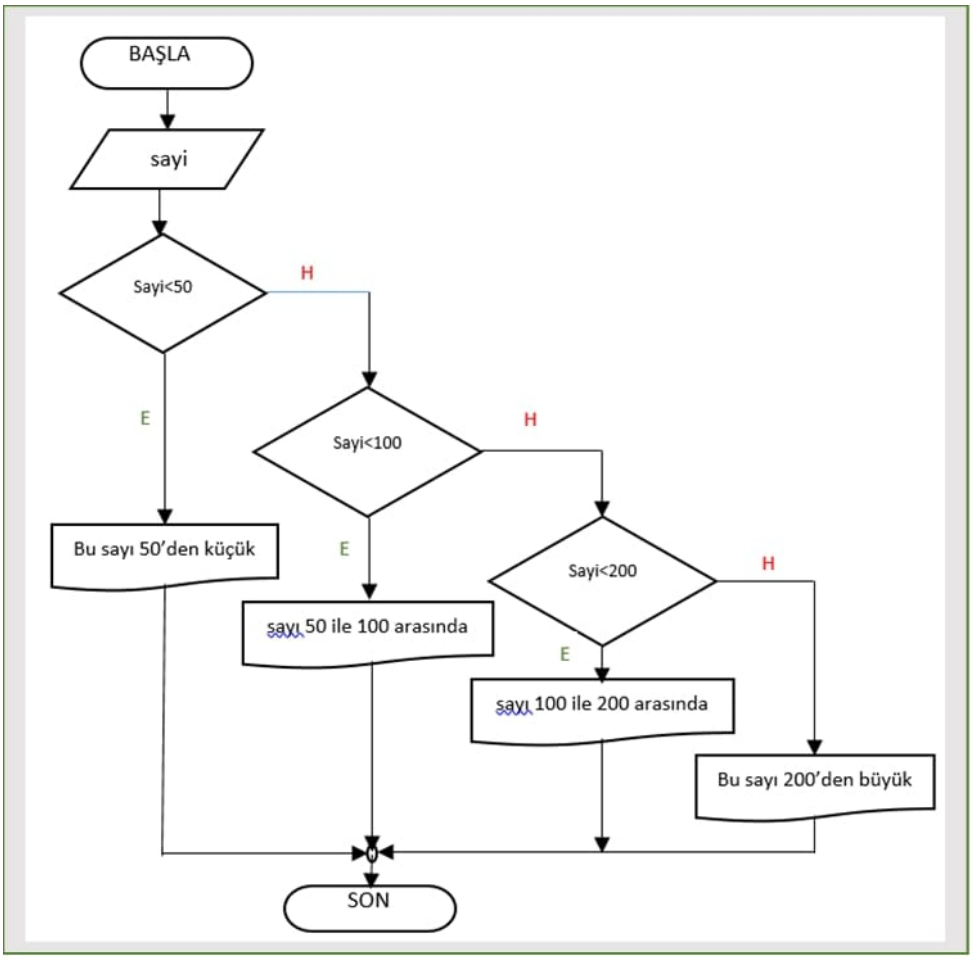
Problem 4.3' ye ait C++ kodu

4.1.4. Program 4.4

Klavyeden girilen bir sayının 50'den küçük veya 50 ile yüz arasında veya 100 ile 200 arasında veya 200'de büyük olduğunu tespit eden ve sonucu ekrana yazdıran algoritmanın akış şemasını çizin ve C++ programını yazınız?

Çözüm:

4.1.4.1. Program 4.4'ün Akış Şeması



Şekil 4.5 *Program 4.4'ün Akış Şeması*

4.1.4.2. Program 4.4'ün C++ Kodu

```
#include <iostream>
using namespace std;
int main()
{
    int sayi;
    cout << "Bir sayi girin: ";
    cin >> sayi;
    cout << "Girdiginiz sayi: " << sayi << endl;
    if (sayi < 50)
        cout << "Bu sayi 50'den kucuk." << endl;
    else if (sayi < 100)
        cout << "Bu sayi 50 ile 100 arasinda." << endl;
    else if (sayi < 200)
        cout << "Bu sayi 100 ile 200 arasinda." << endl;
    else
        cout << "Bu sayi 200'den buyuk." << endl;
    system("PAUSE");
    return 0;
}
```

Program 4.4'ün C++ Kodu

4.1.5. Program 4.5

Bir öğrencinin klavyeden sayısal notu girildiğinde, girilen sayısal notu harf notuna dönüştüren C++ programını yazınız?

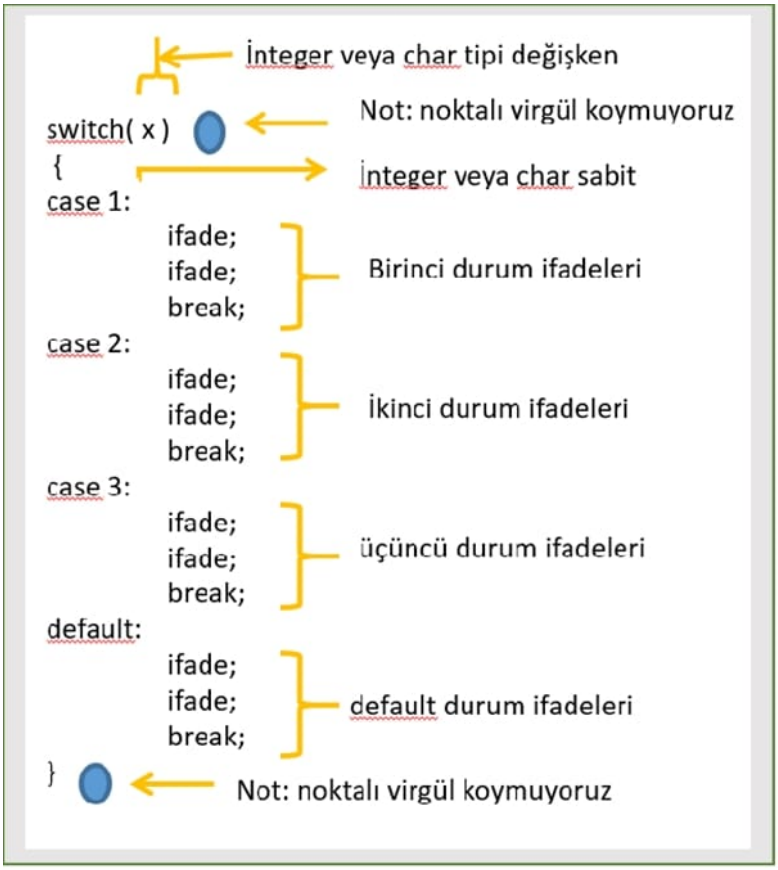
```
#include <iostream>
using namespace std;
int main()
{
    short int ogrnot;
    cout<<"not giriniz : =";
    cin>>ogrnot;
    cout<<"\n\n";
    if(ogrnot>= 90)
    cout<<"notunuz : AA\n";
    else if(ogrnot>=85)
    cout<<"notunuz : BA\n";
    else if(ogrnot>=80)
    cout<<"notunuz : BB\n";
    else if(ogrnot>=75)
    cout<<"notunuz : CB\n";
    else if(ogrnot>=65)
   cout<<"notunuz : CC\n";
    else if(ogrnot>=58)
   cout<<"notunuz : DC\n";
    else if(ogrnot>=50)
   cout<<"notunuz : DD\n";
    else
 cout<<"notunuz : FF\n";
  cout<<"\n\n";
  system("PAUSE");
  return 0;
}
```

4.2. Switch İfadesi

switch yapısının kullanılması kümelenmiş ***if….else*** veya ***else….if*** yapılarına benzer. Önceki bölümden hatırlanacağı gibi ***if….else*** veya ***else….if*** yapılarında, problemin çözümü için gereken sayıda koşulu iç içe yazarak uygun olan koşula bağlı ifade veya ifadeler çalıştırılıyordu. *Ele alınan problemin çözümü için büyük bir karar ağacı olması ve bütün kararların aynı değişkenin değerine bağlı olması durumda da bu problemler* ***if….else*** *veya* ***else….if*** *yapılarıyla çözülebileceği gibi* ***switch*** *yapısı kullanılarak da çözülebilir.* Hatta bu türden problemleri ***if….else*** veya ***else….if*** yapılarıyla çözmek yerine switch yapısı ile çözmek kodun okunabilirliğinin artması ve programcıya kod yazarken kolaylık sağlaması açısından avantaj olabilir.

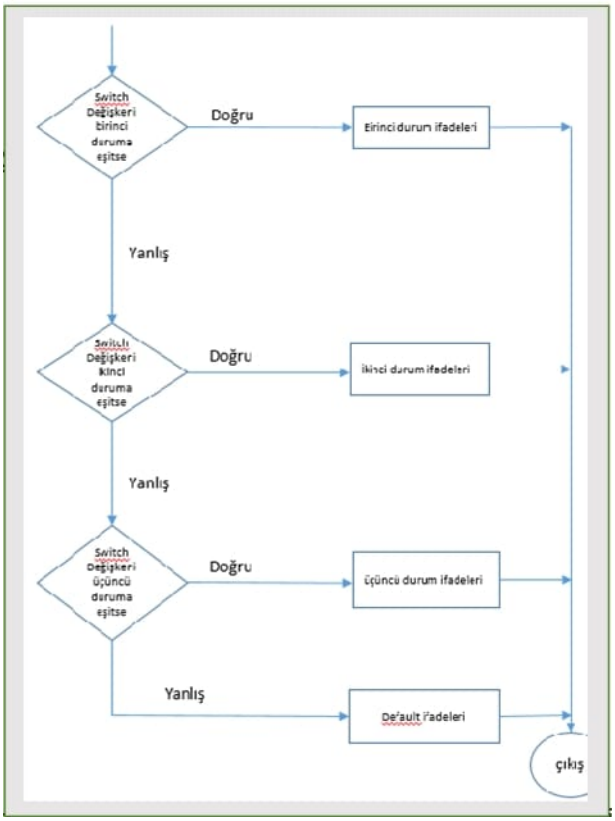
İpucu !! ***Switch*** yapısını, ***if….else*** veya ***else….if*** yerine kullanılacak daha kapsamlı bir yapı düşünmeyiniz. Switch yapısı, daha çok problemin çözümü için büyük bir karar ağacı olması ve bütün kararların aynı değişkenin değerine bağlı olması durumda kullanılır. Switch kullanarak çözülen problemlerin tamamı ***if….else*** veya ***else….if*** yapılarından biri ile çözülür.

Aşağıdaki şekilde ***switch*** yapısı gösterilmiştir.



***Switch** ifadesinin söz dizimi*

Aşağıda ***switch*** ifadesinin akış şeması verilmiştir.



***Şekil 4.5. switch** ifadesini akış şeması*

switch yapısında, ***switch*** anahtar kelimesinden sonra sınanacak bir değişken gelir. Aşağıda verilen örnekte bu değişkenin sayı olduğu görülmektedir.

switch(sayi)

Programda çalışma sırası switch(sayi) ifadesine geldiğinde parantez içerisindeki değişkenin değeri hesaplanır. Daha sonra elde edilen sonuca göre ***case*** ile başlayan ifadelerden biri çalıştırılır ve bu işlemden sonra programın çalışması ***break*** ifadesi kullanılarak ***switch*** bloğu dışına aktarılır. ***Break*** *ifadesi* ***switch*** yapısından çıkılmasını sağlar. Eğer başlangıçta yapılan hesaplama sonucunda elde edilen değişken değeri herhangi bir **case** değerine eşit olmaz ise programın çalışması, yapının ***default*** bölümüne aktarılır ve bu bölümdeki ifade veya ifadeler çalıştırılarak programın akışı ***switch*** bloğundan sonra gelen ilk ifadeye aktarılır.

İpucu !! ***Switch*** yapısında ***default*** ifadesinin kullanılması zorunlu değildir. Ancak, ***switch*** bloğunun başında hesaplanan değişken değeri yapı içerindeki hiçbir ***case*** değerine eşit olmadığı durumda programın çalışması ***switch*** bloğundan sonra gelen ilk ifadeye aktarılır. Bu durumda ***switch*** bloğunda herhangi bir ifade çalıştırılmamış olur.

Örnek: Klavyeden haftanın gün sırası girildiğinde günü ekrana yazdıran programı **switch** yapısı kullanarak yazınız. Program hatalı gün sırası girildiğinde ekrana hata mesajı vermelidir.

Çözüm:

```
#include <iostream>
#include <locale.h>
using namespace std;
int main() {
    setlocale(LC_ALL,"Turkish");
    int gun;
    cout<<"Haftanın Kaçıncı Günü :";
    cin>>gun;
    switch(gun)
    {
        case 1:
            cout<<"Pazartesi"<<endl;
            break;
        case 2:
            cout<<"Salı"<<endl;
            break;
        case 3:
            cout<<"Çarşamba"<<endl;
            break;
        case 4:
            cout<<"Perşembe"<<endl;
            break;
        case 5:
            cout<<"Cuma"<<endl;
            break;

        case 6:
            cout<<"Cumartesi"<<endl;
            break;
        case 7:
            cout<<"Pazar"<<endl;
            break;
        default:
            cout<<"Hatalı Gün"<<endl;
    }
    return 0;
}
```

İpucu !! her ***case*** değişkeninden sonra bir sabit olmalıdır. Sabitten sonra iki nokta işareti(:) gelir. ***case*** sabitlerinin veri tipleri ***switch*** değişkeninin veri tipi ile aynı almalıdır.

Yukarıdaki örnek çalıştırıldığında ve klavyeden gün sırası girildiğinde günün ismi ekrana yazdırılır. Örneğin klavyeden 6 girildiğinde ekrana *Cumartesi* yazdırılır. Eğer klavyeden 1'den küçük 7'den büyük bir sayı girilirse ekrana "*Hatalı Gün*" mesajı yazdırılır. Yukarıda verilen örneği bilgisayarınızda yazıp klavyeden farklı sayılar girerek çalıştırınız ve sonuçlarını inceleyiniz.

Örnek: *Klavyeden iki sayı ve bir dört işlem operatöründen biri girildiğinde operatöre göre dört işlem yapan ve sonucunu ekrana yazdıran bir C++ programı yazınız. (**switch** yapısını kullanınız).*

```
#include <iostream>
#include<conio.h>
#include<locale.h>
using namespace std;
int main()
{
    setlocale(LC_ALL,"Turkish");
    int sayi1,sayi2,sonuc;
    char islemtipi;
    cout<<"sayı giriniz :";
    cin>>sayi1;
    cout<<"sayı giriniz :";
    cin>>sayi2;
    cout<<"operator giriniz :";
    islemtipi=getche();
```

```
    switch(islemtipi)
        {
        case '*':
            sonuc=sayi1*sayi2;
            break;
        case '/':
            sonuc=sayi1/sayi2;
            break;
        case '+':
            sonuc=sayi1+sayi2;
            break;
        case '-':
          sonuc=sayi1-sayi2;
          break;
        default:
    cout<<"girdiğiniz karakter dört islem tipi değil\n";
        }
            cout<<"sonuc :"<<sonuc<<endl;
  system("PAUSE");
  return 0;
}
```

Yukarıda verilen örneği yazıp çalıştırınız. Verilen örnekte klavyeden iki sayı girildikten sonra, çarpma, bölme, toplama ve çıkarma operatörlerinden biri ***switch*** değişkeni olarak yine klavyeden girilmektedir. Bu örnekte ***switch*** değişkeni tipi **char** olarak seçilmiştir. Bu örnekten ***switch*** yapılarında değişken olarak ***char*** tipinin kullanılabileceği gösterilmiştir.

Klavyeden toplama işareti girildiğinde girilen sayılar toplanır ekrana yazdırılır. Benzer şekilde çarpma, bölme ve çıkarma operatörlerinin girişi yapıldığında ise operatörlere uygun sonuçlar ekrana yazdırılır.

Klavyeden bu dört temel operatörün dışında bir karakter girildiğinde ekrana ***default:*** bölümünde verilen hata mesajı yazdırılır.

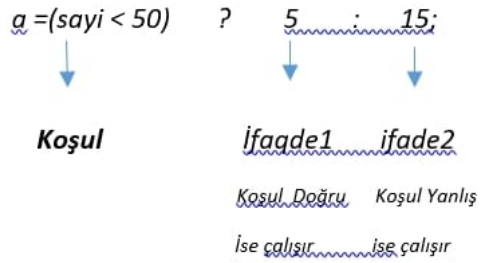
İpucu !! Bir else....if yapısında içinde alakasız değişkenlerin olduğu birçok deyim kullanılabilir. Ancak, switch ifadesinde seçeneklerin her biri aynı değişkene bağlı olmak zorundadır. Bir seçeneği diğerinden ayıran tek özellik değişkenin değeridir. Case sabiti tamsayı veya karakter sabiti olmak zorundadır.

4.3. Koşul Operatörü

Koşul operatörü, üç operand üzerinde işlem yapan iki sembolden meydana gelir. çalışması if….else yapısına benzer. Durum operatörü aşağıdaki şekilde kullanılır.

(ifade1) ? ifade2 : ifade3;

Kuşul operatöründe önce ifad1'in değeri hesaplanır. Eğer bu ifadenin değeri doğru ise ifade2, yanlış ise ifade3 çalıştırılır. Aşağıda koşul operatörünün kullanımını gösteren bir örnek verilmiştir.



Koşul Operatörü için Program:

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

main()
{
    int sayi=-9;
    int a=0;

    a=(sayi>0) ? 5 : 15;

    cout<<"a = "<<a<<endl<<endl;

    return 0;

}
```

Program çalıştırıldığında oluşan ekran çıktısı

```
a = 15
```

Bölüm Özeti

Problemlerin çözümünde koşula bağlı karar verilmesi gereken durumun ikiden fazla olduğu durumda kümelenmiş **if…else** yapılarının nasıl kullanılacağını ve bu problemlere ait algoritmaların akış şemalarının nasıl çizileceğini öğrendik.

Problemin çözümü için büyük bir karar ağacı olması ve bütün kararların aynı değişkenin değeri bağlı olması durumunda, daha çok **switch** deyiminin kullanılması gerektiğini öğrendik. switch deyiminin akış şemasının çizimi ve C++ kodlarının yazılmasını örnekler çözerek pekiştirdik.

Genellikle birçok kaynakta karar yapılarının en son anlatılan konusu koşul operatörü konusudur. Biz de aynı geleneği bozmayarak kümelenmiş if else yapılarının sonunda koşul operatörünü anlattık ve konuyla ilgili bir örnek çözdük.

H. Burak Tungut, Algoritma ve Programlama Mantığı, KODLAB Yayın Dağıtım Yazılım ve Eğitim Hizmetleri San. ve Tic. Ltd. Şti, 2019.

Duygu Arbatlı Yağcı, Nesne Yönelimli C++ Programlama Kılavuzu, Alfa Basım Yayım Dağıtım Ltd. Şti, 2016.

G. Murat Taşbaşı, C programlama, Altaş yayıncılık ve Elektronik Tic. Ltd. Şti. 2005.

Muhammed Mastar, Süha Eriş, C++, KODLAB Yayın Dağıtım Yazılım ve Eğitim Hizmetleri San. ve Tic. Ltd. Şti, 2012.

Sabahat Karaman, Pratik C++ Programlama, Pusula Yayınevi,2004