C PROGRAMLAMA DİLİ

C programlama dili 1970'lerde unix işletim sistemi ile kullanılmak için tasarlanmıştır. Işletim sistemleri derleyiciler gibi sistem programlarının yazılımında yoğun olarak C programlama dili kullanılır. Bir çok dilde C programlama dilinden esinlenerek yazılmıştır. (Java, PHP)

BAZI ÖZELLİKLER

- Güçlü ve esnek bir dildir.
- C ile işletim sistemi yada derleyici yazılabilir.
- Kelime işlemciler oluşturulabilir. Word Exel gibi
- Grafik çizilebilir.
- Yazılım geliştirme ortamına sahiptir.
- Özel komut ve veri tipi tanımlanmasına izin verir.
- Taşınabilir bir dildir.
- Gelişimini tamamlamış ve standardı oluşmuş bir dildir.

Her komut satırı noktalı virgül ile biter.

Çift slash kullanılarak (//) aktif komut satırı inaktif hale getirilir. Bu sayede programa açıklama satırı da eklenebilir.

(/**/) işateti kullanılarak uzun bir komut satırını yada açıklamayı genel kapsamlı olarak inaktif hale getirebiliriz.

C DİLİNDE

İnclude:

Include<stdio.h> c dilinde bu şekilde yazılır. Bu include değimi programa eklenecek olan başlığı belirtir. Programa eklenecek olan başlık dosyasını (HeaderFile) ifade eder. Burada verilen stdio.h başlık dosyasının derleme işlemine dahil edileceğini belirtir.

#include<math.h> bu kütüphane dosyası tüm matematiksel fonksiyonları ifade eder.

MAİN():

Main özel bir fonksiyondur. Ana program bu dosyada saklanıyor anlamındadır. Programın yürütülmesine bu fonksiyondan başlanır. Dolayısıyla her C programında bir tane main adlı fonksiyon bulunmaktadır. Main içerisine kod yazmak için ({}) işaratleri kullanılır ve içerisine tüm kodlar yazılır.

Başlık Dosyaları (HeaderFiles):

C dilinde bir program yazılırken başlık dosyası (HeaderFile) olarak adlandırılan bir takım dosyalar #include ön işlemcisi kullanılarak program içerisine dahil edilir. C kütüphanesinde bulunan birçok fonksiyon başlık dosyaları içindeki bazı bildirimleri kullanır. Bu tür dosyaların uzantısı .h'dır.

ÖRN: ctype.h, assert.h, math.h, time.h, stdio.h, stdlib.h, float.h, string.h gibi kütüphaneler kullanılır.

Bir çok C derleyicisinde yukarıdakilere ek olarak tanımlanmış başlık dosyaları da vardır. Stdio.h standart giriş çıkış kütüphane fonksiyonları için bazı bildirimleri barındıran bir dosyadır.

C++ Dilinde

#include<iostreaw> bu deyimle C++'ın giriş ve çıkış sistemini destekleyen iostreaw dosyasına denk gelir.

Main fonksiyonu C deki ile aynı görevi görür.

C++ da başlık dosyası kütüphane dosyası (HeaderFile) tanımlanırken C den farklı olarak şu şekilde yazılmaktadır <cmath>. C++ .h uzantısı bulunmamaktadır.

C++ daki bir diğer yeni özellik ise namespace lerdir. Bunlar kütüphane fonksiyonları ve diğer elemanların tanımlanması için ayrılmış alanlardır. Namespace kullanmamızın amacı ad çakışmalarını engellemek için tanımlayıcı adlarını uygun şekilde yerleştirmektir. Programımıza yeni bir başlık eklediğimizde bu başlığın içeriği std namespace in içine yerleştirilir ve burada saklanır. Bu iş için #using namespace std; deyimi kullanılabilir. Bu deyim derlendikten sonra eski veya yeni başlıklarla çalışıyor olmamızın bir önemi kalmaz.

İnclude ile başlayan deyimden sonra noktalı virgül gelmez.

NOT: C dosyalarının program uzantıları .cpp dir.

```
/*C ile ilk örnek*/

#include<stdio.h>

Int main  /*program main foksiyonu ile başlıyor*/

{
```

```
Printf(“Merhaba Dünya...\n”);
Return 0; /*programın başarıyla sonlandırıldığını gösterir.*/
}
----------
//C++ deneme programı
#include<iosstreaw>
Using namespace std;
Int main()
{
Cout<<”Merhaba....” << “\n”;
Return 0;
}
```

C++ Ön İşlemcisi

C++ programlarının kendi derleyicileri ile olan ilişkisi C++ ön işlemcisi yardımıyla sağlanır. Çeşitli emirlerden oluşan ön işlemciler C++ derleyicisinin kaynak kodunu denetlemekte kullanılır. Ön işlemci emirleri program içerisinde # işareti ile başlar. C++ ın en çok kullanılan ön işlemci emirleri #include ve #define dır. Ön işlemci emirleri ; işareti ile sonlandırılmaz.

İnclude Emri: bu emir kaynak dosyanın (Başlık dosyaları yani HeaderFiles) programa dahil edilmesini sağlar.

Define Emri: bu emir sembolik değişmezlerin tanımlanmasını sağlar. Söz konusu ön işlemci bir isim ve bir değer ile birlikte tanımlanır. Kullanılan ismin büyük harfle yazılması bir zorunluluk olmamakla beraber genel bir alışkanlık olarak kabul edilmektedir.

```
#include<iostream>
#define Son 50
Using namespace std;
Int main()
{
Cout<<”Sayı..”<<Son<<”\n”;
```

```
Return 0;
}
```

Program çıktısı: Sayı..:50

C kodlarının temel özellikleri

- Yazılımda kullanılacak olan her fonksiyon için ilgili başlık dosyası programın başına ilave edilmelidir. (C++ ile ortak özellik)
- Her C programı main fonksiyonu içermelidir. (C++ ile ortak özelliktir)
- Program içerisinde kullanılacak olan değişken ve sabitler mutlaka tanımlanmalıdır satırın sonuna noktalı virgül işareti gelir. (C++ Ortak özellik)
- Her bloğun ve fonksiyonun başlangıcı ve bitişi süslü parantezler ile yapılır.
- Bir satırı inaktif yorum satırı haline getirmek için C++'da //, C'de ise /* kullanılır.

C VERİ TİPLERİ

Veri tipi program içinde kullanılacak değişken, sabit, fonksiyon isimleri gibi tanımlayıcıların tipini yani bellekte ayrılacak bölgenin büyüklüğünü belirlemek için kullanılır.

C'de 4 tane temel veri tipi bulunmaktadır.

- Char
- Int
- Float
- double

Fakat bazı özel niteleyiciler vardır ki bunlar yukarıdaki temel tiplerin önüne gelerek onların türevlerini oluştururlar.

- Short
- Long
- Unsignet

Bu niteleyiciler sayesinde değişkenin bellekte kaplayacağı alan isteğe göre değiştirilebilir. Short kısa, uzun long ve normal int tam sayı arasında yalnızca uzunluk farkı vardır.

İşaretsiz (Unsignet) ön eki kullanıldığı takdirde veri tipi ile saklanacak değerin sıfır ve sıfırdan büyük olması sağlanır.

C Dilinde

Int x;

Short s; ----> short int s; anlamına gelir.

Long k; ------> long int k; anlamına gelir.

Veri Tipi:

- Char ----------->Karakter veya küçük tamsayı
- Unsigned char ------> ----------->Karakter veya küçük tamsayı
-
- Short int
- Unsigned short int
-
- Int----->Normal
- Insigned int
- Long int
-
- Float ------->Ondalıklı SAyı
- Double

C++ Da Veri tipleri

C++ da değişkenin veri tipleri ve türleri C ile hemen hemen aynıdır. Tam sayıları tanımlamak için yine int, short ve long veri tipi kullanılır. Ondalık sayıları tanımlamak için double, long double, float. Karakter veya karakterleri tanımlamak için ise char kullanılır.

NOT: Hem C hem de C++ küçük büyük harfe duyarlıdır.

Int yas;

Float fiyat;

Char harf;

Sizeof aperatörü: Veri tiplerinin değişkenlerin ve dizilerin bellekte kapladığı alan sizeof operatörü ile öğrenilebilir. Bunun genel yazılımıda sizeof(nesne) şeklindedir.

DEĞİŞKENLER

C veya C++'ta küçük büyük harfe duyarlılık vardır örneğin:

char harf

char Harf

char HARF

Bunların üçü de ayrı birer değişkendir.

- C ve C++'ta değişken tanımlamada alt çizgi _ karakteri haricinde hiçbir karakter veya özel sembol kullanılamaz.
- C ve C++'ta Değişken rakam ile başlamaz fakat alt çizgi ve harf ile başlayabilir.
- C ve C++'ta Değişken adı tanımlanmasında boşluk karakteri kullanılmaz.
- C ve C++'ta Değişken ismi verirken programlama diline ait komut veya deyim ismi kullanılmaz.

1) Yerel Değişkenler

Program içinde birden fazla fonksiyon varsa sadece tanımlandığı fonksiyonda geçerli olabilecek değişken türüdür. Bu değişkenler fonksiyon sınırlarını belirten {} işaretleri içinde yer alırlar.

2)Küresel Değişkenler (Global)

Program içerisindeki tüm fonksiyonlarda geçerli olabilecek değişken türüdür. Bu tür değişkenler fonksiyon sınırlarını belirten {} içerisinde yer almalıdır.

3)Statik Değişkenler

Bir fonksiyonun içinde yerel olarak tanımlanmış bir değişkenin program çalıştığı sürece içinde bulunduğu fonkisyonun tekrar tekrar çağırılması durumunda sabit kalması ve değişmemesi istendiğinde o değişken statik değişken olarak tanımlanmaktadır.

SABİTLER

Sabitler programın başından sonunda kadar değeri değişmeyen program bileşenleridir.

C++ dilinde şu tip sabitler kullanılır:

- Tam sayı sabitler
- Ondalıklı sabitler
- Karakter sabitler
- Katar sabitler

TAM SAYI SABİTLER

Tam sayı sabitler int, short, long olmak üzere üç türlüdür.

Bir sabitin hangi türe ait olduğunu belirtmek için o sabitin sonuna türünü belirtecek karakter eklenir. Eğer sayısal bir ifadenin sonunda herhangi bir karakter yoksa o ifadenin türü int olarak kabul edilir.

Bu ifadeyi long türü olarak belirtmek için sonuna “l veya L” karakteri eklememiz gerekmektedir. Bu şekilde artık ifade int türüne değil long türüne ait olur

Short türü için bir ifadenin değeri hesaplanırken short türüne ait olsada int gibi işlem görürler. Aslında short türünde bir sabit yoktur olarak da kabul edebiliriz çünkü C++ tarafından int türü olarak kabul edilmektedir.

ONDALIKLI SABİTLER

Bu sabitler float (kayan ondalıklı), double(Çift ondalıklı), long double(uzun ondalıklı) olmak üzere üç türlüdür.

Eğer ondalıklı bir sabitin sonunda herhangi bir karakter yoksa o ifadenin türü double olarak kabul edilir. Ama bu ifadeyi 276.372 float türü olarak kabul etmek için “f veya F” karakteri eklememiz gerekir.

Pek kullanılmamakla birlikte ifadenin sonuna “l veya L” eklenirse long double olarak kabul edilebilir.

KARAKTER SABİTLER

Karakter sabitleri kullanırken karakterlerin sayısal karşılıklarını kullanabildiğimiz gibi doğrudan kendilerini de kullanabiliriz. Yanlız karakter sabitler tek tırnak içerisinde belirtmek zorundayız.

Char karakter1=’g’;

Char karakter2=103;

Yukarıdaki kodların ikiside aynıdır. Oradaki 103 sayısı g’nin aski kod karşılığıdır.

KATAR SABİTLER

String(Karakter) sabitler, ardışık olarak sıralanmış karakter sabiti dizilerinden oluşmaktadır. C++ da veya C de çift tırnak içine alınmış her ifade string türü ifadedir.

KONTROL KARAKTERLERİ ve TİP KARAKTERLERİ

Değişkenlerin ve sabitlerin nasıl yazılacağını belirtmek veya imlecin alt satıra geçirilmesi gibi bazı işlemlerin gerçekleştirilmesi için kullanılır.

C DİLİNDE

printf("\t Selam....\n");

Karakter ve Anlamları

- \a: Ses Üretimi(alert)
- \b: İmleci Bir sola kaydır (backspace)
- \f: Sayfa Atla (Bir sonraki sayfanın başına geç)
- \n: Bir Alt satıra geç
- \r: Satır başı yap
- \t: yatay tab(horizontal tab)
- \v: Dikey tab (vertical Tab)
- \": çift tırnak işareti yap
- \': tek tırnak işareti yap
- %%: % karakterini ekrana yaz

NOT: Bu karakterlerin tamamı (control karakterleri) C++'da da kullanılır aynıdır.

C++ da ekstra olarak

\0:null demektir anlamı ise Karakter dizilerini string haline getirmek.

TİP KARAKTERLERİ

% işareti ile başlar ve bir veya iki karakterden oluşur. Örneğin %d gibi. Ekrana yazdırılmak istenen değişkenin tipi % işretinden sonra belirtilir.

C DİLİNDE

printf("%d", x+y)

Tip Karakterleri	Anlamı	Yazdırılacak Veri Tipi
%c	Tek Bir Karakter	Char
%s	Karakter Dizisi(String)	Char
%d	İşaretli Ondalık Tam Sayı	İnt, Short
%ld	Uzun İşaretli ondalık tam sayı	Long
%u	İşaretsiz ondalıklı tam sayı	Unsigned int, unsignet short
%lu	İşaretsiz uzun tam sayı	Unsigned long
%f	Gerçel Sayı	float
%lt	Çift Duyarlı gerçel sayı	double

Program:

printf("Atom numarası=%d",13)

printf("Yoğunluk=%f",2.7)

Çıktı:

Atom Numarası=13

Yoğunluk=2.7

TÜR DÖNÜŞÜMÜ

C++'da Örn Prog:

```
#include<iostream>
using mainspace std;
int main ()
{
int sayi=9;
float a,b,c;
a=Sayi/4; Tam sayıya bölündü virgül yok sayıldı
b=sayi/4.0; Virgüllü sayıya bölündü sonuç virgüllü çıktı
c=float(Sayi)/4; tür dönüşümü yapıldı sonuç virgüllü çıktı
```

}

Programlarımızda değişkenler veya sabitler birbirinden farklı türde olabilir. Eğer böyle bir durum söz konusu ise matematiksel işlemlerimizde hesap sonucumuzun hangi türde olacağı önemlidir. bu nedenle bir hata ile karşılaşmamak için tür dönüşümü yapılmalıdır.

TEMEL GİRİŞ VE ÇIKIŞ FONKSİYONLARI

C DİLİNDE

Temel giriş çıkış fonksiyonları bütün programlama dillerinde mevcuttur. Bu tür fonksiyonlar kullanıcıya ekrana veya yazıcıya bilgi yazdırmasına ve bilgisayara klavyeden veri girişi yapmasına izin verir. Bu fonksiyonlar kullanılırken C'de <stdio.h> başlık dosyası dahil edilmelidir.

printf() Fonksiyonu: bu fonksiyon değişkenlerin tuttuğu değerleri onların adreslerini veya bir mesajı ekrana belli bir formatta düzenle yani ekrana yazdırmak için kullanılan fonksiyondur.

scanf() Fonksiyonu: bir çok programda ekrana verilerin bastırılması, klavyeden girilmesi veri okunması için kullanılan komuttur.

Örn: scanf("%d",&x);

Bu örnekte klavyeden bir x tamsayısı okumak için kullanılan örnektir. Burada & işareti adres operatörüdür.

C++ DİLİNDE

C++ da ise standart giriş-çıkış fonksiyonlarını kullanmak ve giriş çıkış nesnesine yönlendirme yapmak için "<<" işleci kullanılır. Giriş çıkış fonksiyonları için iostream dosyası tanımlanır. Programda verileri mesaj ifadelerinin değişkenleri ve değerlerini ekrana yazdırmak için cout deyimi kullanılır. C++ ta giriş fonksiyonu olarak yani klavyeden veri girmek için kullanılan komut ise cin dir.

C DİLİNDE

KOD:

```
#include<stduio.h>
int main ()
{
printf("Merhaba...."\n");
return 0;
}
```

ÇIKTI:

Merhaba

C++ DİLİNDE

KOD:

```
#include<iostream>
using namespace std;
int main()
{
cout<<"Merhaba..."<<"\n";
cout<<"Merhaba Bil Prog";
return 0;
}
```

Kod:

```
#include<iostream>
using namespace std;
int main()
{
int sayi1=10;
```

```
int sayi2=15;
int toplam;
toplam=sayi1+sayi2;
cout<<"Sonuç..."<<toplam<<"\n";
return 0;
}
```

Kod:

```
#include<iostream>
using namespace std;
int main()
{
int sayı;
cout<<"Sayıyı Giriniz..";
cin>>sayi;
cout<<"Sayı..."<<sayi<<"\n";
return 0;
}
```

UYGULAMALAR

Ekrana Yadırma	

C++ Dili
 	
#include <iostream>
using namespace std;
int main ()
{
cout <<"Merhaba.."<<"/n"
cout <<"BilProg1!";
 	
return 0;
}
	C Dili
	
	#include <stdio.h>
	main ()
	{
	printf("Merhaba...:\n");
	printf("BilProg...:");
	return 0;
	}

BASİT HESAP MAKİNESİ	
C++ Dilinde	C Dilinde

```
#include <iostream>
using namespace std;
int main()
{
int sayi1;
int sayi2;
int t,f,b,c;

cout<<"1.Sayiyi Giriniz...";
cin>> sayi1
cout<<"2.Sayiyi Gir:";
cin>>sayi2;

t=sayi1+sayi2;
f=sayi1-sayi2;
b=sayi1/sayi2;
c=sayi1*sayi2;

cout<<"Toplam Degeri:"<<t<<"\n";
cout<<"Fark Degeri:"<<f<<"\n";
cout<<"Carpim Degeri:"<<c<<"\n";
cout<<"Bolum Degeri:"<<b<<"\n";

return 0;
}
```

```
#include <stdio.h>
main()
{
        int t;
        int g;
        int h;
        int b,f,j;

        printf("Bir Gercel Sayi Giriniz:");
        scanf("%d",&g);
        printf("Bir Gercel Sayi Giriniz:");
        scanf("%d",&t);

        h=g*t;
        b=g/t;
        f=g-t;
        j=g+t;

        printf("toplam=%d\n",j);
        printf("fark=%d\n",f);
        printf("carpim=%d\n",h);
        printf("bolum=%d\n",b);
        return 0;
}
```

Getchar() fonksiyonu: Bu fonksiyon ile standart girişten bir karakter okunur. Programı istenen yerde durdurup bir karakter girinceye kadar bekletir. Yani bir tuşa basıncaya kadar bekleme sağlar.

Puts() Fonksiyonu(C Dilinde): ekrana yazdırılacak ifade bir karakter topluluğu ise printf fonksiyonuna alternatif olarak puts fonksiyonu kullanılabilir. Puts fonksiyonu bu karakter topluluğunu yazdıktan sonra imleci bir alt satıra geçirir.

Printf(“Bu bir Deneme..\n”); = puts(“Bu bir Deneme..”)

Gets() Fonksiyonu: klavyeden bri karakter topluluğu oluşmak için kullanılır. Okuma işlemi yeni satır karakteri ile (\n) karşılaşılıncaya kadar sürer

Scanf(“%s”,str); = gets(str);

Endl: programda yeni bir satırdan itibaren devam edileceğini gösteren (program satırı sonu) deyimdir.

Aritmetik Operatörler

Değişken veya sabitler üzerinde aritmetik işlemleri gerçekleştiren operatörlerdir.

Operatör	açıklama	İşlem	Anlamı
+	Toplama	X+y	toplama
-	Çıkarma	X-y	Çıkartma
*	Çarpma	X*y	Çarpma
/	Bölme	x/y	Bölüm
%	Artık Bölme	X%y	x/y den kalan sayı

Atama Operatörleri

Bu operatörler bir değişkene bir sabit veya bir aritmetik ifade atamak (eşitlemek için) kullanılır.

Bileşik Atamalar: Bazı ifadelerde işlem operatörü ile atama operatörü birlikte kullanılarak ifadeler daha kısa yazılabilir. Bu atamalar bileşik atamalardır.

Örnek:

C Dilinde:

```
#include <stdio.h>

#include <stdlib.h>

int main(int argc, char *argv[])

{

        int a;

        int b;

        int toplam;

        a=25;

        b=13;

        toplam=a+b;

        printf("a sayisi %d ve b sayisi %d, Toplami %d \n",a,b,toplam);

        return 0;

}
```

Aritmetik Operatörler

Operatör	açıklama	İşlem	Anlamı
+	Toplama	X+y	toplama
-	Çıkarma	X-y	Çıkartma
*	Çarpma	X*y	Çarpma
/	Bölme	x/y	Bölüm
%	Artık Bölme	X%y	x/y den kalan sayı

Atama Operatörleri

Bu operatörler bir değişkene, bir sabit veya artimetik ifade atamak (eşitlemek) için kullanılır.

Bileşik Atama

Bazı ifadelerde işlem operatörü atama operatörü ile birlikte kullanılarak ifadeler daha kısa yazılabilir.

Operatör	Açıklama	Örnek	Anlam
=	Atama	X=7;	X=7
+=	Ekleyerek atama	X+=3	X=x+3
-=	Eksilterek atama	X-=5	X=x-5
*=	Çarparak Atama	X*=4	X=x*4
/=	Bölerek Atama	x/=2	X=x/2
%=	Bölüp kalanını atama	X%=9	X=x%9
++	Bir arttırma	X++ veya ++x	X=x+1
--	Bir eksiltme	X- – veya - -X	X=x-1

Arttırma ve Azaltma İşleçleri

(++) İşleci Yanındaki değişkenin değerini 1 arttırır. (- -) işleci ise 1 azaltır. Söz konusu işleçlerin değişkenin solunda veya sağında yer alması durumunda anlamı değişir.

1) Arttırma işlecini değişkenin solunda kullanırsak a=++b bu durumda arttırma işleci önce b nin değerini 1 arttırır ve sonra a değişkenine atar. Bu durumda a ve b değişkenlerinin değeri aynı olur.
2) Eğer arttırma işlecini değişkenin sağında kullanırsak yani a=b++ bu durumda arttırma işleci bu durumda önce b nin değerini a ya atar sonra b nin değerini 1 arttırır. Bu durumda b nin değeri a nın değerinden bir fazla olur.

C++ ÖRN:

```
int a,b;
cout<<"Birinci sayiyi giriniz: ";
cin>>a;
cout<<"ikinci Sayiyi Giriniz: ";
cin>>b;
--a;
--b;
cout<<"sonuc="<<a*b<<endl;
```

Karşılaştırma işleçleri

İki sayısal değeri veya karakteri karşılaştırmak amacıyla kullanılan işleçlerdir. Karşılaştırma yapmak amacıyla ise if deyimi kullanılır. Eğer karşılaştırmanın sonucu doğru ise bu deyimin ardından gelen satır işlem görür. Doğru değil ise else deyiminden sonra gelen satır işlem görür.

İşlem	Anlam
>	Büyüktür
<	Küçüktür
>=	Büyük yada Eşit
<=	Küçük yada eşit
==	Eşitmi
!=	Eşit Değilse

Mantıksal İşleçler

Iki veya daha fazla sayıdaki koşulu birlikse sınanması amacıyla kullanılır bu durumda iki veya daha fazla koşul doğruluk değerleri göz önüne alınarak birlikte değerlendirilir.

İşleç	Anlamı	Açıklama
&&	ve	Verilen ifadelerin tümünün koşulu sağlaması gerekmektedir.
||	veya	Verilen ifadelerin en az biri koşulu sağlamalıdır.
!	Değil	Kendisinden sonra gelen ifadenin zıttını alır.

Örnek C++ proje:

```
#include <iostream>

using namespace std;

int main(int argc, char** argv)
{
	cout<<"merhaba.."<<"\n";
	getchar();
	cout<<"bilprog1!";

	return 0;
}
```

```
#include <iostream>

using namespace std;

#define PI 3.14159
#define YENISATIR '\n'

int main(int argc, char** argv)
{
```

```
	double r=5.0;
	double cember;

	cember=2*PI*r;
	cout<<cember;

	return 0;
}
```

```
#include <iostream>

using namespace std;

int main(int argc, char** argv)
{
	int sayac;
	int a;

	sayac=5; a=0;
	a=sayac++;
	cout<<"a= "<<a++<<"\n";
	cout<<"sayac="<<sayac<<"\n";
	cout<<"--------"<<"\n";
	sayac=5; a=0;
	a=++sayac;
	cout<<"a= "<<++a<<"\n";
	cout<<"sayac= "<<sayac<<"\n";

	return 0;
}
```

Örnek C Proje:

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

int main(int argc, char *argv[])
{
	int t, s, h;

	printf("Bir Gercel sayi giriniz:");
	scanf("%d",&s);
	printf("Bir gercel sayi giriniz:");
	scanf("%d",&t);

	h=s*t;
	printf("Sonuc=%d\n",h);

	return 0;
}
```

Karşılaştırma İşlemleri

C programlamanın pek çok yerinde iki ifadenin karşılaştırılması işlemine başvurmak gerekecektir karşılaştırma işlemlerinde karşılaştırma işleçleri kullanılabileceği gibi mantıksal işleçlerde kullanılabilir.

Sayı>5 (Sayı değişkeninin değeri 5'den büyük)

Değer!=2 (değer 2'ye eşit değilse)

Sayac==5 (Sayac değişkeninin değeri 5'e eşit)

S<= (S 6'dan küçük yada eşitse)

İf Yapısı

Karşılaştırma işlemi sonucunda bir eylemin yapılması söz konusu ise diğer bir ifade ile belirli deyimlerin çalıştırılması gerekiyorsa if deyimine başvurulur. If deyimi şu şekilde tanımlanır:

If ifade

deyim

Bu tanıma göre ifade içerisinde bulunan koşulun doğru olması durumunda if içinde belirtilen deyim çalışır. Aksi halde işlem görmez.

Karşılaştırma işlemi sonucunda birden fazla deyimin çalışıtırılması söz konusu ise bu durumda ({ }) simgelerini kullanmamız gerekir.

```
If(ifade)
{
Deyim1;
Deyim2;
}
```

İf-Else yapısı

Bir koşulun gerçekleşmemesi durumunda yerine getirilecek eylemleri belirlemek için else deyimi kullanılır.

```
If ifade
Deyim1;
Else
Deyim2;
```

Bu tanıma göre ifade içinde belirtilen koşulun doğru olması halinde deyim 1 çalışır. Aksi halde ise (ifade doğru değilse) deyim2 işlem görür.

Program içinde else sözcüğünün kullanılması durumunda deyim gruplarına yer verilebilir. Deyim grupları süslü parantez ({}) işaretleri arasında tanımlanmalıdır.

If ifade

{

Deyim1;

Deyim2;

}

Else

{

Deyim1;

Deyim2;

}

C++ İf Örnekler

```
#include <iostream>

using namespace std;

int sayi=5;
int main(int argc, char** argv)
{
	if(sayi<10)
	{
		cout<<"Kosul Dogrudur.."<<"\n";
	}

	return 0;
}
```

```
#include <iostream>

using namespace std;

int sayi=5;
int main(int argc, char** argv)
{
	if(sayi<10)
	{
		cout<<"Kosul Dogrudur..";
		cout<<"bildiniz.."<<"\n";
	}

	return 0;
}
```

```
#include <iostream>

using namespace std;


int main(int argc, char** argv)
{
	int sayi1, sayi2;

	cout<<"Birinci Sayiyi giriniz..";
	cin>>sayi1;
	cout<<"Ikinci Sayiyi Giriniz..";
	cin>>sayi2;

	if(sayi1>0 && sayi2>0)
	{
		cout<<"Her Iki Sayi Pozitiftir..";
	}
	else
	{
		cout<<"Sayilardan en az birisi negatiftir..";
	}

	return 0;
}
```

```
#include <iostream>

using namespace std;


int main(int argc, char** argv)
{
	int sayi;

	cout<<"bir sayi giriniz...";
	cin>>sayi;

	if(sayi>0)
	{
		cout<<"Pozitif Sayi Girdiniz.."<<" Sayiniz="<<sayi;
	}
	else
	{
		cout<<"Negatif Bir Sayi Girdiniz.."<<"Sayiniz="<<sayi;
	}

	return 0;
}
```

C Örnekler

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

int main(int argc, char *argv[])
{
	int sayi;
	printf("Bir sayi giriniz: ");
	scanf("%d",&sayi);

	if(sayi % 2==0)
	{
		printf("Sayi cifttir...");
	}
	else
	{
		printf("Sayi Tektir...");
	}

	return 0;
}
```

C++ Örnek

```
#include <iostream>

using namespace std;

int main(int argc, char** argv)
{
	int sayi;
	cout<<"Bir Sayi giriniz:";
	cin>>sayi;
	if(sayi<0)
	{
		cout<<"Negatif bir sayi girdiniz.."<<sayi<<"\n";
	}
	else
	{
		cout<<"Pozitif bir sayi girdiniz.."<<sayi<<"\n";
	}

	return 0;
}
```

```
#include <iostream>

using namespace std;

int main(int argc, char** argv)
{
	int sayi1,sayi2;
	cout<<"Bir Sayi giriniz:";
	cin>>sayi1;
	cout<<"Ikinci Sayiyi Giriniz:";
	cin>>sayi2;
	if(sayi1>0 && sayi2>0)
	{
		cout<<"Her iki sayi da pozitiftir";
	}
	else
```

```
	{
		cout<<"Sayilardan en az biri negatiftir.";
	}
	return 0;
}
```

```
#include <iostream>

using namespace std;

int main(int argc, char** argv)
{
	int sayi1, sayi2;
	cout<<"Birinci sayiyi giriniz:";
	cin>>sayi1;
	cout<<"Ikinci Sayiyi Giriniz:";
	cin>>sayi2;
	if(sayi1>sayi2)
	{
		cout<<"Birinci sayi daha buyuktur."<<endl;
	}
	else
	{
		cout<<"Ikinci sayi daha buyuktur."<<endl;
	}
	return 0;
}
```

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

const double yuzalti=0.5;
const double yuzustu=0.2;
int kilometre;
double ucretyuzalti, ucretyuzustu, toplamucret;
int main(int argc, char** argv)
{
	cout<<"Gidilen KM'yi Giriniz:";
	cin>>kilometre;
	if(kilometre<=100)
	{
		ucretyuzalti=kilometre*yuzalti;
		ucretyuzustu=0;
	}
	else
	{
		ucretyuzalti=100*yuzalti;
		ucretyuzustu=(kilometre-100)*yuzustu;
	}
	toplamucret=ucretyuzalti+ucretyuzustu;
	cout<<"\nToplam Ucret="<<toplamucret<<"TL"<<"\n";
	return 0;
}
```

C++ Örnek

```
#include <iostream>

using namespace std;

int basari;
char harfnot;

int main(int argc, char** argv)
{
	cout<<"Basari Notunu giriniz:";
	cin>>basari;

	if(basari>=90)
	{
		harfnot='A';
	}
	else if(basari>=80)
	{
		harfnot='B';
	}
	else if(basari>=70)
	{
		harfnot='C';
	}
	else if(basari>=60)
	{
		harfnot='D';
	}
	else
	{
		harfnot='F';
	}
	cout<<"Ogrencinin Harf Notu:"<<harfnot<<endl;
	return 0;
}
```

```
#include <iostream>

using namespace std;

char adi[22];
char s_adi[22];
float saat;
float top_satis, prim, ucret;

int main(int argc, char** argv)
{
	cout<<"**** Ucret Hesabi **** \n\n";
	cout<<"Satis elemaninin adini giriniz:";
	cin>>adi;
	cout<<"Satis elemaninin soyadini giriniz:";
	cin>>s_adi;
	cout<<"Calistigi toplam saati giriniz:";
	cin>>saat;

	ucret=50*(float)saat;
	cout<<"Sattigi urunler toplamini giriniz:";
	cin>>top_satis;

	if(top_satis>5000)
	{
		prim=1000;
```

```
	}
	else
	{
		prim=0;
	}

	cout<<"\n**** Odenecek Ucret **** \n";
	cout<<"\nSatis Elemani: "<<adi<<" "<<s_adi<<"\n";
	cout<<"\nMaas Bordrosu:\n";
	cout<<"----------\n";
	cout<<"Ucret= "<<ucret<<"ve Alacagi Prim="<<prim<<"\n";
	cout<<"Toplam= "<<ucret+prim<<"TL Odenecektir."<<endl;

	return 0;
}
```

C Örnek

```
#include <stdio.h>
#include<math.h>

int main(int argc, char *argv[])
{
	float a,b,c,delta,x1,x2,x,kok_delta;
	printf("a,b,c degerlerini girin:\n");
	scanf("%f %f %f",&a,&b,&c);
	delta=b*b-4.0*a*c;
	if(delta>0.0)
	{
	x1=(-b+sqrt(delta))/(2.0*a);
	x2=(-b-sqrt(delta))/(2.0*a);
	printf("\nReel Kokler:");
	printf("\nx1=%f",x1);
	printf("\nx2=%f",x2);
	}
	else if(delta<0.0)
	{
		kok_delta=(sqrt(-delta))/(2.0*a);
		x=-0.5*b/a;
		printf("\nx1=%.2f+(%.2f)i",x,kok_delta);
		printf("\nx2=%.2f-(%.2f)i",x,kok_delta);
	}
	else
	{
		x=-0.5*b/a;
		printf("\nKokler Esit");
		printf("\nx1=x2=%f",x);
	}

	return 0;
}
```