BÖLÜM 1b: C++ PROGRAMLAMANIN YAPISI

C++, hard diskte TC, BIN, INCLUDE, LIB… gibi alt dizinlere yüklenir.

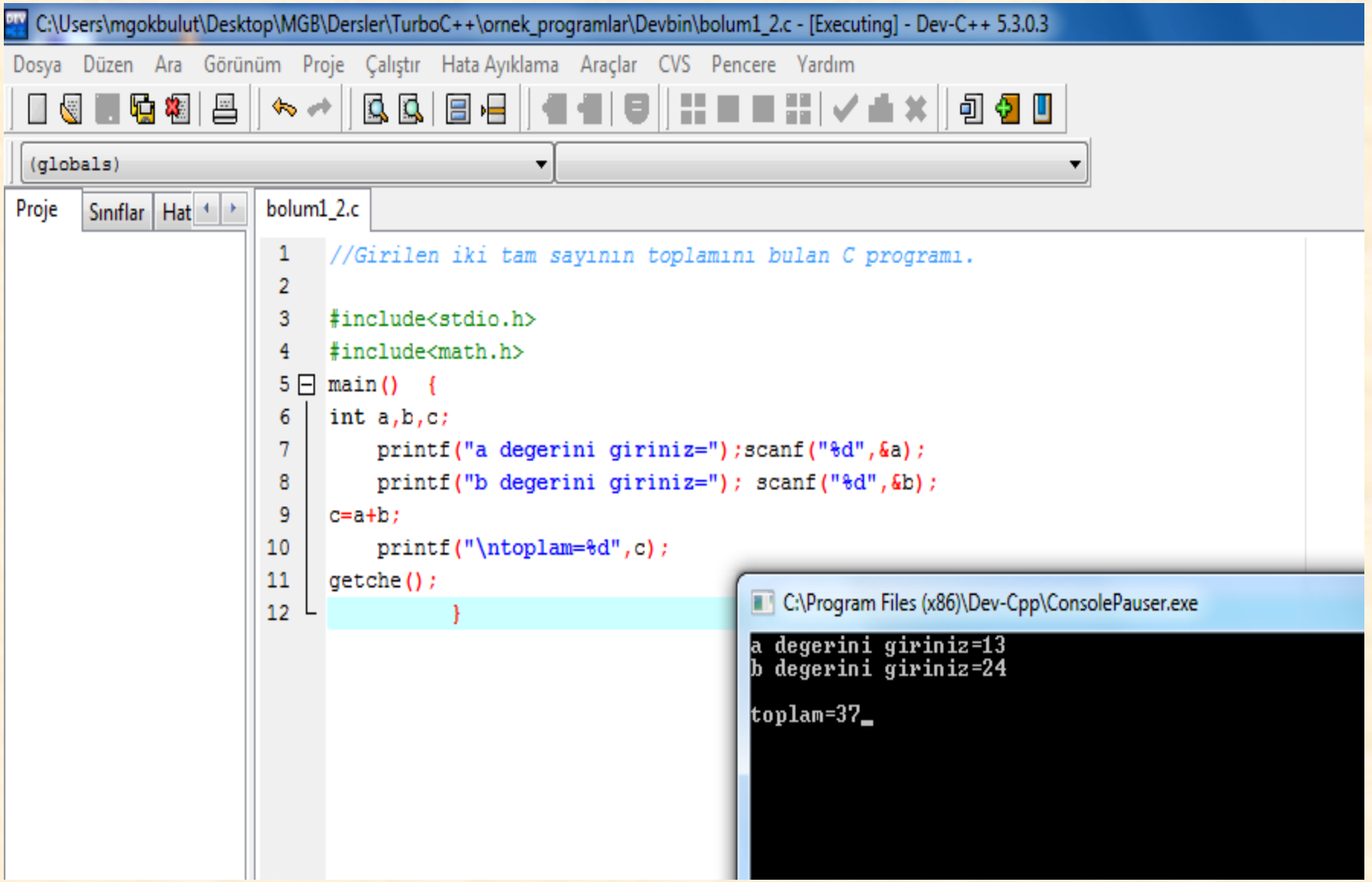
TC programı çalıştırıldığında C++ çalışma ortamı açılır.

C++ çalışma ortamında istenirse yeni bir program sayfası açılarak program yazılır istenirse önceden yazılan bir program yüklenerek çalıştırılır.

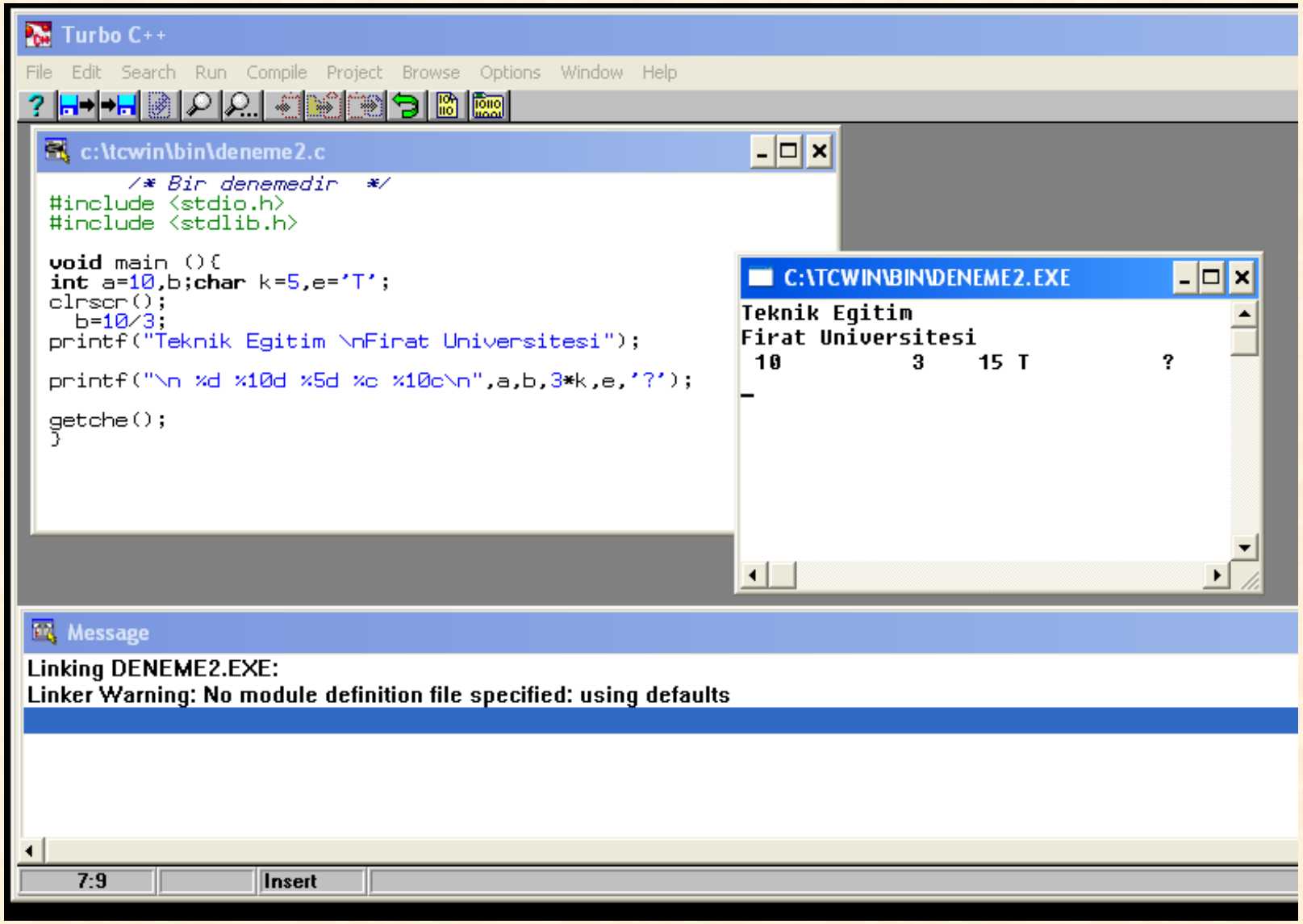
Menü seçenekleri yardımıyla açma, kaydetme yazılan programı derleme ve çalıştırma vs gerekli işlemler yapılır.

Aşağıda WINDOWS tabanlı C++ ve ayrıca developer C++ programında yazılmış örnek programların yapısı ve C++ çalışma ortamları sayfaları görülmektedir.

Developer C++


C:\Users\mgokbulut\Desktop\MGB\Dersler\TurboC++\ornek_programlar\Devbin\bolum1_2.c - [Executing] - Dev-C++ 5.3.0.3
Dosya Düzen Ara Görünüm Proje Çalıştır Hata Ayıklama Araçlar CVS Pencere Yardım
(globals)
Proje Sınıflar Hat
bolum1_2.c
1 //Girilen iki tam sayının toplamını bulan C programı.
2
3 #include<stdio.h>
4 #include<math.h>
5 main() {
6 int a,b,c;
7 printf("a degerini giriniz=");scanf("%d",&a);
8 printf("b degerini giriniz="); scanf("%d",&b);
9 c=a+b;
10 printf("\ntoplam=%d",c);
11 getche();
12 }
C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe
a degerini giriniz=13
b degerini giriniz=24
toplam=37

WINDOWS Tabanlı C++



C++ Programın Yapısı

```
        /*  Açıklama satırları*/
# include < stdio.h>
# include <conio.h>                          Ön bildiriler
# define a 5
# define yaz printf
int i ;                                      Global değişken tanımlamaları
float b=12.5;

        deneme(){                             Alt fonksiyon başlangıcı
        int firat=10;                         Alt fonksiyon algoritması
        ---------                             ---------------
        }                                     Alt fonksiyon sonu

main () {                                     Ana fonksiyon başlangıcı
        char y=10;                            Lokal değişken tanımlamaları
        printf (“firat”) ;                    Ana fonksiyon algoritması
        yaz (“üniversitesi”) ;                ----------
        …….                                   ……
 }                                            Ana fonksiyon sonu
```

C’ de Açıklama Satırları

Programa açıklama satırı eklemek için ***//*** **veya** ***/**** işaretleri kullanılır.

// *Tüm satır açıklama satırı olarak kullanılır.*

/* *Bu işaretten sonra programın tüm satırları açıklama satırı olarak kullanılır.*

***/*.....**

Aralıktaki satırlar açıklama satırları olarak kullanılır.

*/

Derleyici Ön Bildirileri

Kullanıcı tarafından derleyiciye bildirilen ve programın derlenmesinden önce değerlendirilen komut niteliğindeki ifadelerdir.

Ön bildiriler (#) diyez işaretiyle başlar.

Ön bildiriler yardımıyla karşılığı text olan, sabit veya özel isimler tanımlanabilir, mevcut programa bir başka program dosyası ilişkilendirilebilir ve programın koşullu olarak çalışması sağlanabilir.

define Ön Bildirisi : *# define makroadı karakter dizisi*

define ön bildirisi kullanıldığında, program içerisinde makroadı ifadesinin görüldüğü her yerde karakter dizisi ifadesiyle gösterilen ifade yazılmış gibi hareket eder.

Örn:

```
# define  PI       3.1452
# define  yaz      printf
# define  basla    main () {
# define  f (x, y)   x*x-2*y-x/y
```

a = f(2,3) ?

undef Ön Bildirisi :

Define ile tanımlanan sabit ya da makro işlemlerini tanımsız hale getirerek programın başka bir yerinde aynı isimlerle tekrar tanımlanmasına imkanı verir. Dolayısıyla define ve undef bildirileri, main fonksiyonu içerisinde de kullanılabilir.

Örn:

```
# define f (X)   X*X – 2
        …….
# undef  f
# define f(X)   X*X*X
```

include Ön Bildirisi :

Mevcut program içerisine bir başka kaynak program dosyasını dahil ederek her iki programın birlikte derlenmesini sağlar.

 # include < dosya ismi >
kaynak dosya **include** dizini içerisinde aranır.

 # include “dosya ismi”
kaynak dosya aktif dizin içerisinde aranır.

Örn:
include < stdio.h>
include “ c:\firat\deneme.c”

C derleyicisinin doğrudan tanıdığı komutlar oldukça azdır. Ancak çok sayıda kütüphane fonksiyonu tanımlandığından ve bu fonksiyonlar belirli header (*.h) dosyaları içerisinden derleyiciye tanıtıldığından bir program yazarken kullanılacak fonksiyonlarla ilgili header dosyasının programa dahil edilmesi gerekir. **Bazı *.h dosyaları**

conio.h : Ekran, klavye ve renk ayarlarıyla ilgili işlem yapan fonksiyonların tipini barındıran kütüphanedir.

graphics.h : Grafik Çizimleri üzerinde işlem yapan fonksiyonların tiplerini barındıran kütüphanedir.

math.h : Matematiksel işlemlerde gerekli olan fonksiyonları barındıran kütüphanedir.

stdio.h : Standart giriş/çıkış kütüphanesidir. Genelde gerekli olan bütün fonksiyonlara referans eder.

stlib.h : Standart fonksiyonların tiplerini tanımlayan kütüphanedir. Dönüşüm, sıralama, atama vb.

C++ Dilinde Operatörler

Operatörler, bilgisayara çeşitli matematiksel ya da mantıksal işlemleri yapmasını bildiren sembollerdir.

Matematiksel Operatörler :

+Toplama - Çıkarma * Çarpma / Bölme = atama

++ Arttırma - - Eksiltme % Modüler bölme (Bölme işleminde kalan)

Örn:

X = 10 % 3 ise X = 1, X = 10 % 5 ise X = 0 ve X=1%2 ise X=1 (ilk değer)

X = X+1 , X+ = 1, ++ X , X++ X ' in değeri 1 artar .

X = X –1 , X - = 1 , -- X, X-- X ' in değeri 1 azalır.

İşlemleri matematiksel yerine (X = X+1 gibi) operatörlerle yapmak işlemi hızlandırır.

X = 1 ; y = X ++ Önce atama sonra artırma yapar X = 2 , y = 1

X = 1 ; y = X -- ise X = 0, y = 1

X = 1 ; y = ++ X Önce artırma sonra atama yapar X = 2 , y = 2

X =1 ; y = -- X X = 0 , y = 0

Öncelik Sırası : ++ , -- , * , / , % , + , -

Karşılaştırma Operatörleri :

= =	Eşittir	> =	Büyük Eşit
>	Büyük	<=	Küçük eşit
<	Küçük	!=	Farklı (eşit değil)

Karşılaştırma işlemi doğru ise 1, yanlış ise 0 üretirler. Ör: a=1<2 ise a=0 olur.

Mantıksal Operatörler :

&&	and (ve)
||	or (veya)
!	not

Örn: x=(a > = 5 && b= = 3) doğru ise **x = 1 ;** yanlış ise **x = 0 ;**

Bit Operatörleri :

Sayıların bit değerlerini alarak bit' ler üzerinde işlem yapan operatörlerdir.

&	and	^	xor
|	or	<<	sola kaydırma
~	not	>>	sağa kaydırma

C++ Dilinde Veri Tipleri ve Değişkenler

- Programlama dillerinde işlemler; **tamsayı, gerçel (ondalıklı) sayılar ve karakter yada stringler** (karakter zincirleri) gibi çeşitli veri tipleri üzerinde yapılır.
- Ayrıca, bir programda işlenecek verilerin büyüklüğüne ve duyarlılığına bağlı olarak bellekte uygun büyüklükte (byte olarak) saklamak da önemlidir.
-
- Diğer taraftan bilgisayarlar belleğinde bilgi saklamak amacıyla ayrılan yerlere veri-değer aktarmak ya da bellekteki mevcut veriyi-değeri kullanmak için belirlenen sembollere (isimlere) değişken denir. **C dilinde bir değişken kullanılmadan önce mutlaka tanımlanmalıdır.**

- Buna göre de C++ dilinde veriler aşağıdaki tiplerle tanımlanmıştır.

Veri Türü	Veri Tipi	Bellek (byte)	Tanım aralığı
Tam sayı	char ya da signed char	1	-128 ile +127
	unsigned char	1	0 ile 255 (2^8)-1
	int ya da signed int/signed short int	2	-32768 ile 32767
	unsigned int ya da unsigned short int	2	0 ile 65535 (2^16)-1
	long int ya da signed long int	4	-(2^31)+1 ile (2^31)-1
	long int ya da unsigned long int	4	0 ile (2^32)-1
	long long int	8	-(2^63)+1 ile (2^63)-1
	unsigned long long int	8	0 ile (2^64)-1
Ondalık sayılar	float	4	1.7 10^-38 ile ±1.7 10^38 yada
	long float yada double	8	1.7 10^-308 ile ± 1.7 10^308
	long double	10	1.7 10^-4933 ile 1.7 10^4933
karakter	char	1	

Değişken tanımlama

Bilgisayarlar belleğinde bilgi saklamak amacıyla ayrılan yerlere veri-değer aktarmak ya da mevcut veriyi-değeri kullanmak için belirlenen sembollere (isimlere) değişken denir. **C dilinde bir değişken kullanılmadan önce mutlaka tanımlanmalıdır.**

C de Değişken Tanımlama ;

[işaret] [uzunluk] Veri Tipi Değişken adı [= ilk değer]

Satırı ile yapılır ve ilk değer atanmayabilir. Değişken adlarında Türkçe karakterler kullanılmamalı, 32 karakteri geçmemeli ve C de kullanılan fonksiyon ya da komutlar kullanılmamalıdır.

Veri Tipleri ise önceki sayfada tanımlanmıştır.

Örnekler:

char a,b = 20 , c= ‘F’ , d= ‘8’ , adı [] = “fatih” , s [10] ;
unsigned char x=300 ??

Örnek:

```
int a,b = 100, c,d,x ;
unsigned lont int m,n=50;
a=b/3          => a=33
c=a/2          => c= 16
```

float: Tek hassasiyetli (7 digit ondalık) reel sayı içerikli değişkenleri tanımlar.

Örnek:

```
float a,b = 100 ;
a=b/3                  => a=33.3333333
```

double: Çift hassasiyetli (15 digit ondalık) reel sayı içerikli değişkenleri tanımlar.

Örnek:

```
double a,b = 100 ;
a=b/3                  => a=33.33…..33333
```

İşaret Bildirileri (signed, unsigned) kullanılabilir aralarındaki fark tabloda verilmiştir.

Örn: unsigned int a,b ; a ve b = 0 – 65535
unsigned char x ; x = 0 – 255

Uzunluk Bildirileri (short , long) kullanılabilir aralarındaki fark tabloda verilmiştir.

long int a; int = 2 bayt olduğundan
long int= 4 bayt olur ve dolayısıyla sayı değeri: 2^32

sizeof : Çeşitli veri tiplerinin bellek alan uzunluklarını verir.
int a,b ; float x;
a= sizeof(b) ; b= sizeof (double) ; a= sizeof (x) ;

Tip dönüştürme Operatörü:

Tanımlanan bir değişkenin yada ifadenin tipini başka bir tipe çevirir.
(float) 10/3 => 3.333 (10.0/3 yani ilk elemanı etkiler)
(float) (10/3) => 3.000 (sonuç etkilenir)

C' de Temel Bazı Fonksiyonlar – Komutlar

printf : standart çıkış birimine (ekrana) bilgi yazmayı sağlar. Kullanımı:

```
printf (" Açıklama" ) ;
printf (" Değişken formatları" , değişkenler ) ;
```

Örn:

```
printf ( " firat " ) ;
printf ( "\ n Elektronik") ;
printf (" sayı = % d " ,25) ;
printf ( " \n sonuç = %d % c ", a ,b ) ;
```

Değişkenleri Yazma ve Okutma Formatları

Değişkenler ekrana yazdırılırken ya da bir değişkene değer verilirken aşağıdaki format karakterleri kullanılır.

% d – işaretli tamsayılar (int ve char)
% u – işaretsiz tamsayılar
% o – işaretsiz tamsayıların 8 tabanlı
% x– işaretsiz tamsayıların 16 tabanlı
% ld – long tipi işaretli tamsayılar
% f – float tipi sayıların
% lf – long float ve double sayıların
% Lf – reel sayılar long-double

% c – bir tek karakteri yazdırma – okutma
% s – karakter zincirini (string)
% e – reel sayıları üstel formda yazdırma ve okutmaya yarar.

scanf() : Standart giriş biriminden bilgi okutma komutudur .

scanf (“Değişken formatları“ , değişkenler) ;

Örn: scanf (“ % d , % c , % s “ , & x, & y , adı) ;

NOT: stringlerde adres işareti gerekmez.

putchar : Bir karakter değişkeni ekrana yazar

putchar (x) ; == printf (“%c” ,x) ;

getchar: Standart giriş biriminden bir karakterlik bilgi girişi bekler.

```
char x,y ;
x=getchar () ;    ==    scanf (“\n %c “ , & y ) ;
```

<u>getche ya da getch:</u> Klavyeden bir tuş girişi bekler ve girilen tuşu ekrana yazar.

getche()

Özel Karakterler

\ n – Bir sonraki satır başını verir
\ a – zil sesi
\ t – tab verme karakteri
\ v – düşey tab verme (bir sonraki satırın aynı sütunu)
\ r – aynı satırın satır başı
\” – ” özel karakter.

C dilinde küçük harflerle büyük harfler birbirinden farklı olarak değerlendirilmektedir. Bu nedenle program yazılırken büyük harf küçük harf ayrımına dikkat etmek gerekir.

Örnek 1.1

Çıktı ?

```
# include <stdio.h>
# include <conio.h>
# define a 5
# define yaz printf

main () {
        int i=10,x ;
        float b=12.5;
        char y='A';
x=(a>=5 && i==3);
printf ("firat") ;  yaz ("  Universitesi") ;

printf("\n \n %d %f \n %c  %d",a,b/2,++y,x);
        getch();
         }
```

Örnek 1.2

Çıktı ?

```
/* Bit operatörleri*/
#include<stdio.h>

int a=2,b=3,c,d,e,f,g,h,k;

main(){
c=a&b; d= a|b;
e= ~a;  f= a^b;
g= a<<1;
h=a<=b;
k=a==2||b<3;
a-=++a-b++;       //a=a-(++a-b++)
printf("%d  %d  %d  %d   %d   %d
%d",c,d,e,f,g,h,k);
}
```

NOT: İkili tabanda Negatif sayılar, sayının değili alınır ve 1 ile toplanarak bulunur.

Örnek 1.3

*//*Girilen iki tam sayının toplamını bulan C programı.

Çıktı ?

```
#include<stdio.h>
#include<conio.h>
main()  {
int a,b,c;
        printf(“a değerini giriniz=”);scanf(“%d”,&a);
        printf(“b değerini giriniz=”);scanf(“%d”,&b)
c=a+b;
        printf(“\ntoplam=%d”,c);
getche();
          }
```

Örnek 1.4

Çıktı ?

```
/* Değişkenleri tanımlama ve yazdırma programı*/
#include<stdio.h>
#include<conio.h>

int x=10/3;
float y=-10.0/3,t;
double w=10.0/3;
long z=10/3;              // long int z=10/3

main() {

t=x+y;
        printf("%+d  %f %2.8f  %ld\n\n%lf  %2.18lf
%f\n",x,y,y,z,w,w,t);
getch();
}
```

Örnek 1.5

```
/* Değişkenleri tanımlama ve yazdırma programı*/
# include <stdio.h>
# include <conio.h>

main() {
int a=5,b;
char x,y='A',adi[10]="Firat",soyadi[15];
float n=103.5;

printf("b yi giriniz:");scanf("%d",&b);
printf("x i ve soyadi giriniz:");scanf("%d %s",&x,soyadi);

printf("\n%d,  %+3.4f,  %-3d",b/a,n,a);
printf("\n%d,  %2c,  %s,  %d",x,y,soyadi,y);
getche();
}
```

Çıktı ?

Global ve Yöresel Değişkenler

C++ de bir değişkenin kullanılmadan önce mutlaka tanımlanması gerektiğine değinmiştik. Değişken bildirimi temel olarak iki farklı yerde yapılabilir. Bir fonksiyon içinde bildirilen değişkenlere LOCAL (yöresel) değişkenler , bütün fonksiyonların dışında bildirilen değişkenlere GLOBAL değişkenler adı verilir.

```
# include < stdio.h>
int i;
main () {  int j;
              ____
              ____   }

altfonk () {  int k ;
                    ____   }
```

Burada **i** değişkeni global olup hem main, hem de altfonk fonksiyonunun içinde kullanılabilir. **j** ve **k** ise yöreseldir. Sadece kendi fonksiyonları içerisinde kullanılabilir.

Örnek 1.6

Çıktı ?

```
//Global ve lokal değişkenler
# include <stdio.h>
# include <conio.h>

deneme () {     int a = 10 ;
printf (“ \na = %d “ , a) ; }

main() {            int a=5 ;
                    printf (“ \na=%d “ a) ;
                    deneme () ;
          { int a= 15 ;
          printf (“\na=%d “ ,a) ; }

                    printf (“\na=%d “,a) ;
                    getche () ;
                   }
```

Bazı Hazır (Kütüphane) Fonksiyonları

abort: Anormal olarak ve bir uyarı mesajı da vererek programı kırar (stdlib.h) .

```
abort();
```

abs: Bir doğal sayının mutlak değerini verir (math.h,stdlib.h) .

```
x=abs(n);
```

acos: Arccosinüsü verir (math.h) .

```
y=acos(0.5);
```

asin: Arcsinüsü verir (math.h) .

```
y=asin(0.5);
```

atan: Arctanjantını verir (math.h) .

```
y=atan(2);
```

Trigonometrik fonksiyonlara Radyan olarak giriş verilmelidir.

sin: Sinüsü verir (math.h) .

```
y=sin(30);
```

cos: Cosinüsü verir (math.h) .

```
y=cos(30);
```

tan: Tanjantını verir (math.h) .

```
y=tan(30);
```

atof: Sayı içerikli bir stringi float sayıya çevirir (stdlib.h) .

```
char s[]=”1234.567”;
x=atof(s);
```

Bazı Hazır (Kütüphane) Fonksiyonları

atoi: Sayı içerikli bir stringi tamsayıya çevirir (stdlib.h) .

```
char s[]=”125”;
x=atoi(s);
```

atol: Bir stringi long sayıya çevirir (stdlib.h) .

```
char s[]=”123456789123.....”;
x=atol(s);
```

clock: Mikroişlemci zamanını belirler (time.h) .

clock_t ,t1,t2 ile önce zaman başlatılmalı ve zaman CLK_TCK’ya bölünerek saniyeye çevrilebilir.

```
main(){
clock_t  t1,t2;
t1=clock();
delay(2000);
t2=clock();
printf(“%f”,(t1-t2)/CLK_TCK);  }
```

delay: Programın işleyişini belli bir süre(milisaniye) bekletir (dos.h) .

sound: Belirlenen frekansta PC’nin mikrofonunu çalıştırır (dos.h) .

```
sound(10);
delay(5000);
nosound();
```

Bazı Hazır (Kütüphane) Fonksiyonları

time: Günün saatini belirler (time.h)(ocak 1,1970den başlayarak geçen zamanı saniye olarak verir) .

```
time_t  x;
x=time(NULL);
printf("%1d",x);
```

difftime: İki zaman arasındaki farkı belirler (time.h) .

```
time_t  t1,t2;
t1=time(NULL);
delay(2000);
t2=time(NULL);
t=difftime(t2,t1);
```

exit: Programın işleyişini durdurur (stdlib.h) .

```
exit(0);
```

exp: e tabanına göre üst alır (math.h) .

```
y=exp(x);
```

gotoxy: Kursörü verilen koordinatlara götürür (conio.h) .

```
gotoxy(35,20);
printf(".....");
```

log: Doğal logaritmayı verir (ln(x),math.h) .

```
y=log(x);
```

Bazı Hazır (Kütüphane) Fonksiyonları

log10: 10 tabanına göre logaritmayı hesaplar .

y=log10(x);

pow: Üst hesaplar (math.h).

pow(x,y); ise x üzeri y

pow10: 10 tabanına göre üst hesaplar (math.h) .

pow10(x); ise 10 üzeri x

randomize: Rastgele sayı üretim generatörünü başlatır (stdlib.h) .

random: Rastgele sayı generatörü (stdlib.h) .

```
randomize();
y=random(100);  (0-99 arasında bir sayı üretir)
```

sqrt: Karekök alır (math.h) .

y=sqrt(x);