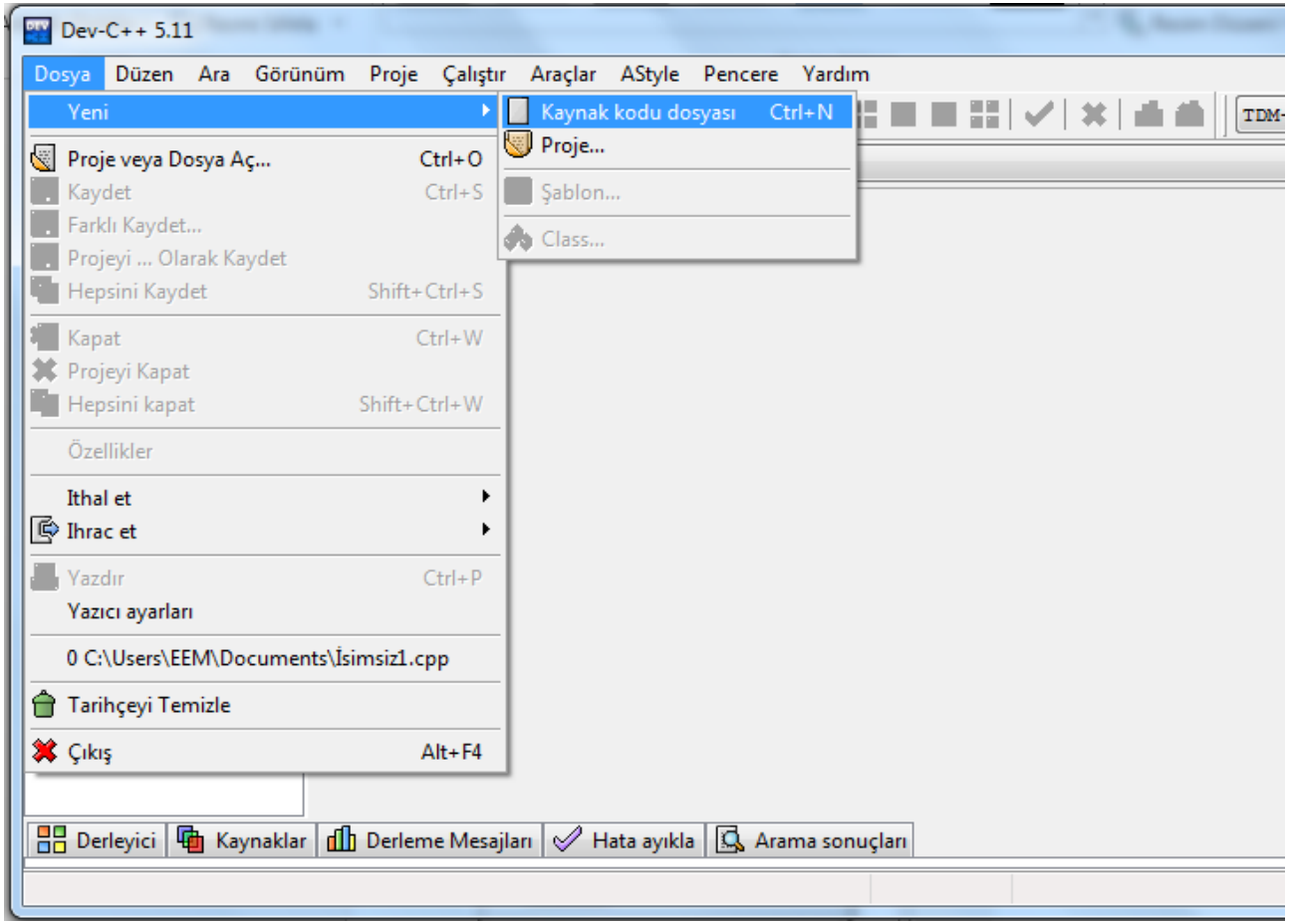
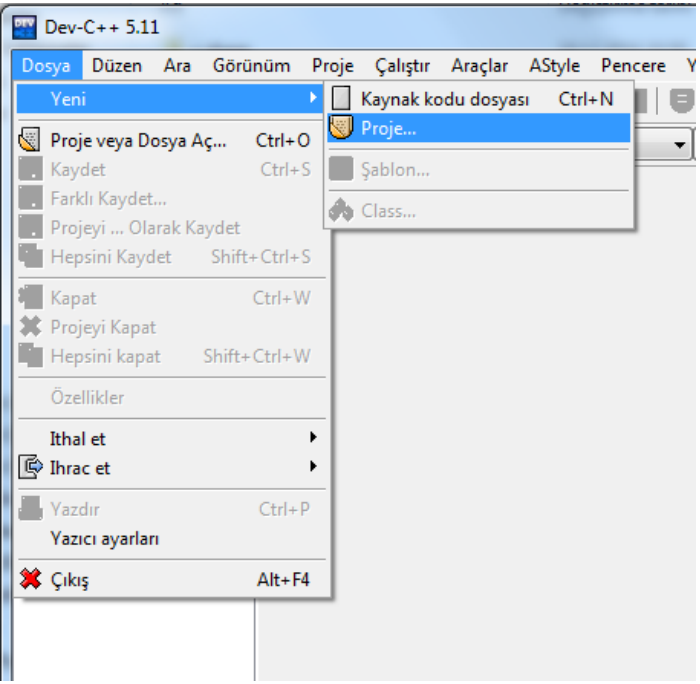
Dev C ++ Editörü

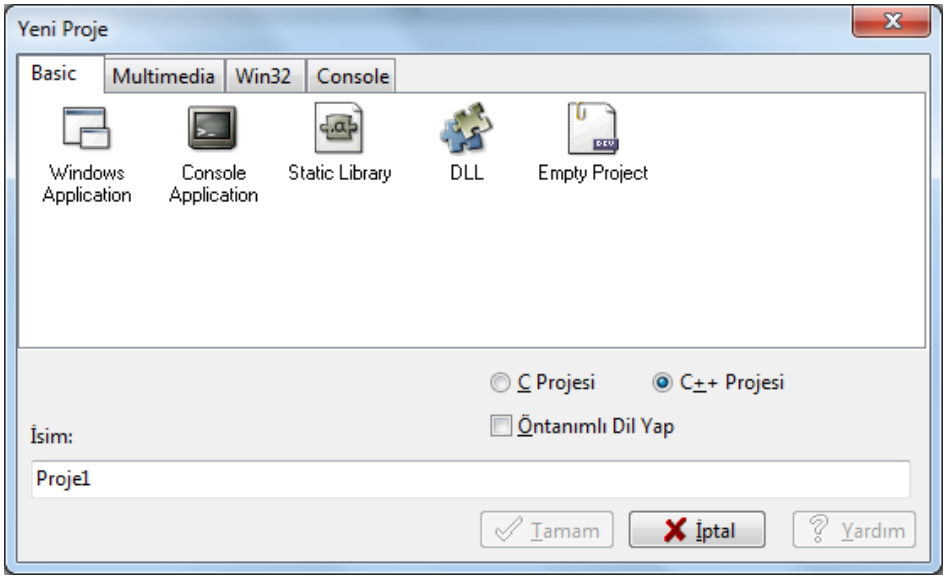
Dev C++ kaynak kodlu dosya ile yeni bir boş belge oluşturmak



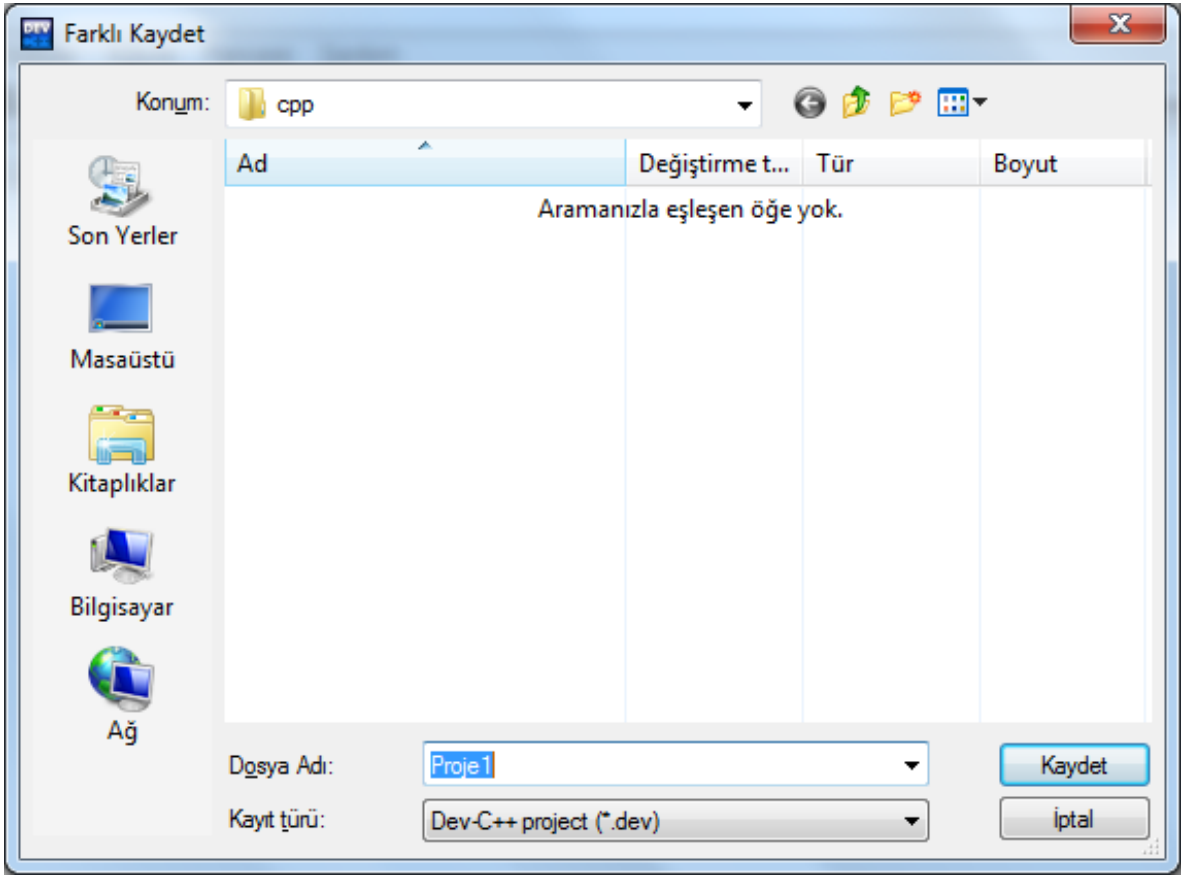
Dev C++ Konsol ile program oluşturma (Burası işlenmeyecek)

1. **Başlat → Programlar → Bloodshed Dev-C++ → Dev-C++** açın.
2. **Dev-C++** açılınca menüden **File→New Project** seçin.
3. Yeni Proje seçeneklerinden Console Application 'dan C Project ve isim kısmına da proje1 ismini yazıyoruz.

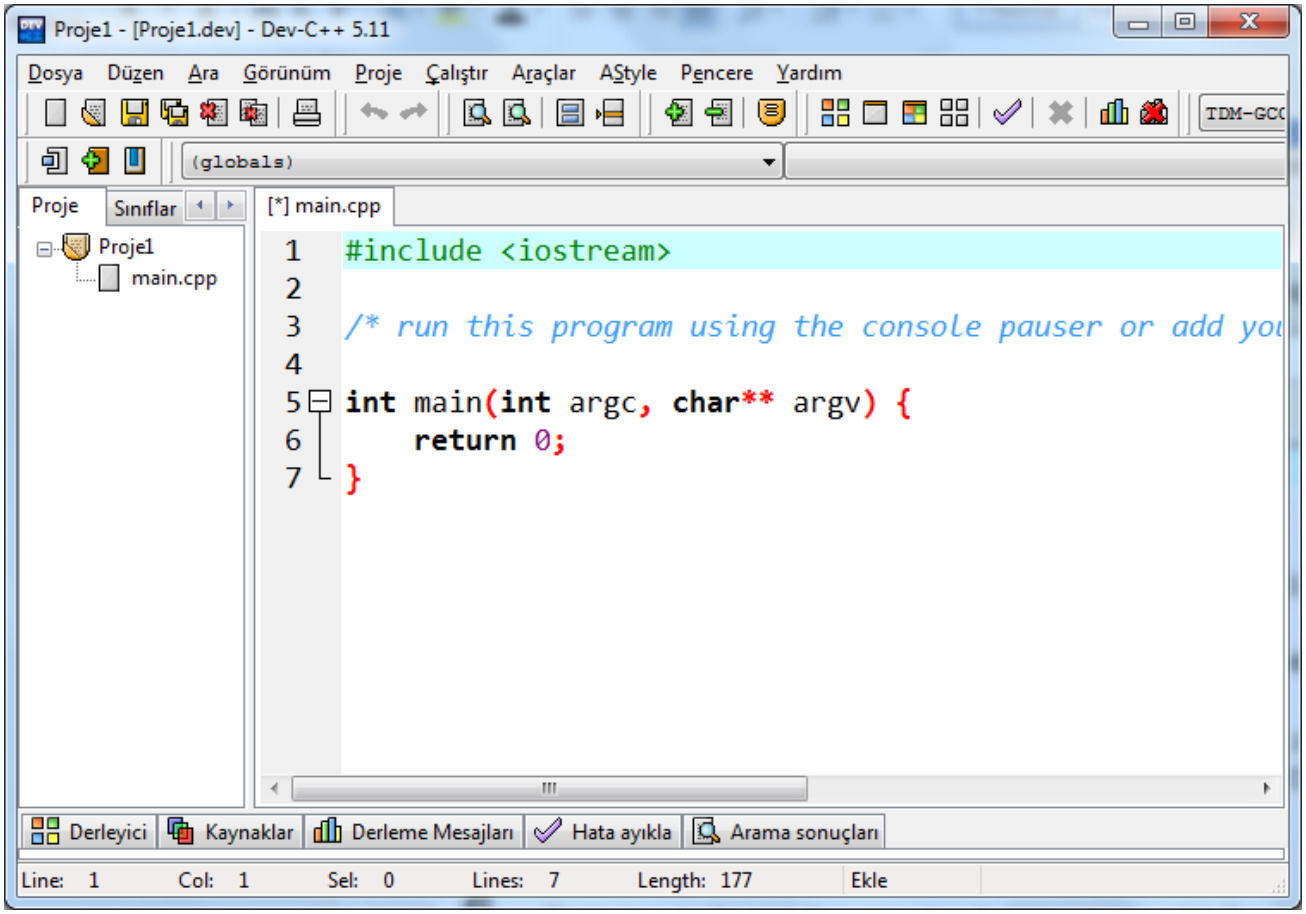




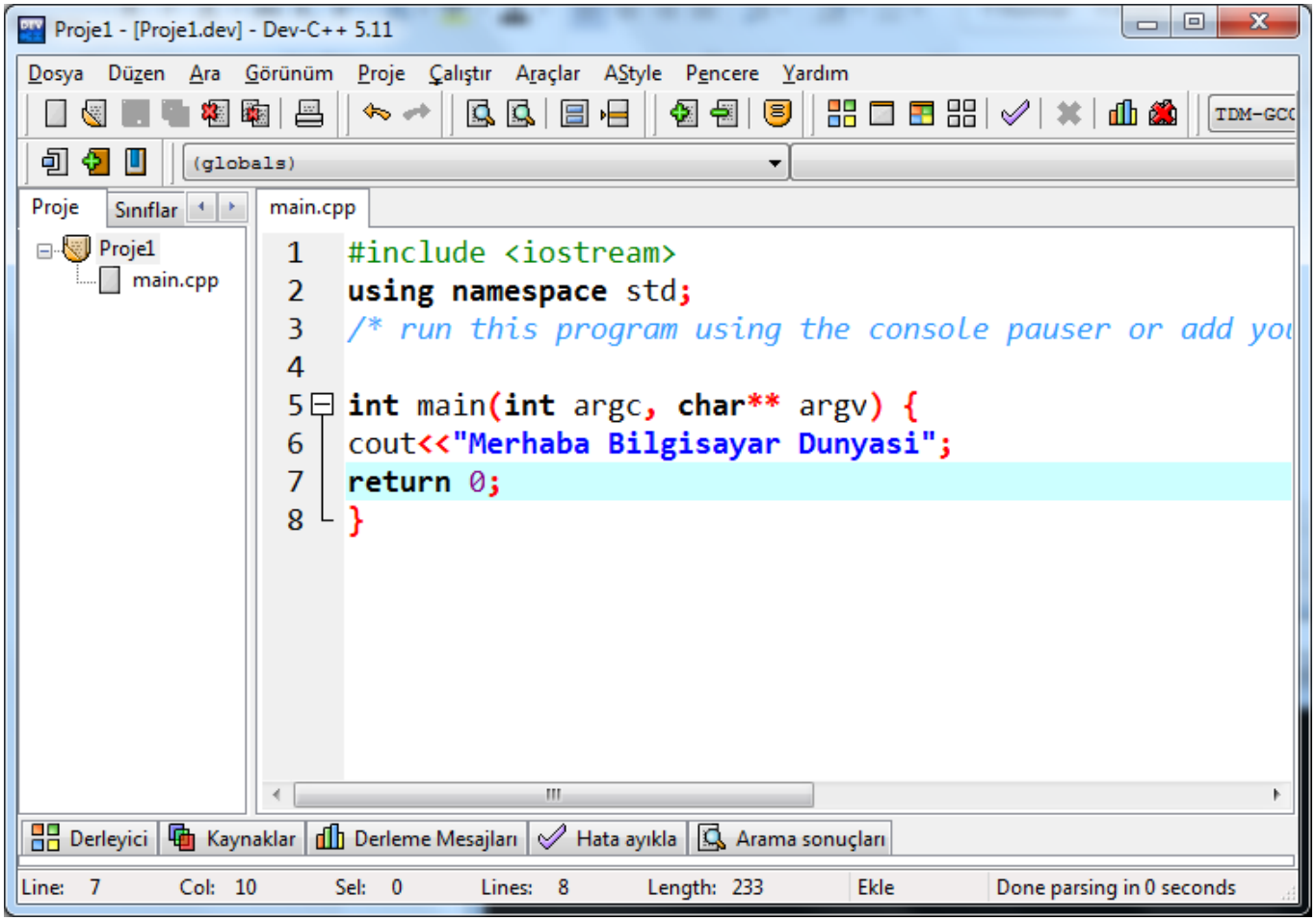
4. Projeyi kaydetmek için bir dizin seçin. Örnek olarak, **C:\cpp** adlı bir dizin oluşturup projemizi bu dizin altına kaydedebiliriz. Her projeyi **C:\cpp** altında ayrı bir dizine kaydetmeye dikkat etmemiz gerekir. Bütün projeler aynı dizine kaydedilirse proje dosyaları karışır.



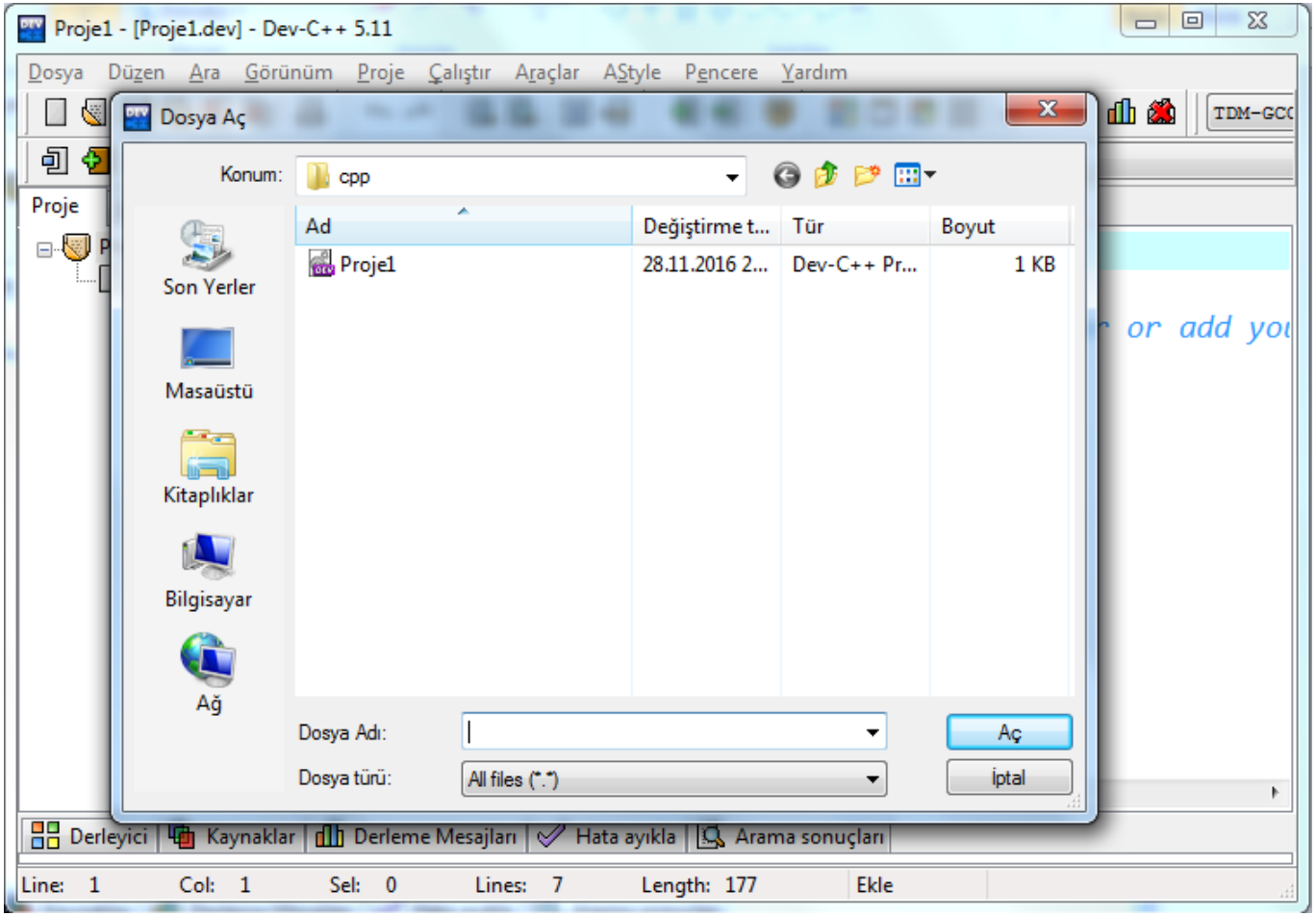
5. Projeyi oluşturunca **main.cpp** adlı bir ..**cpp** dosyası oluşacaktır. Bu dosya projenin ana dosyasıdır ve yazacağınız **C++** kodlarını bu dosyaya yazmanız gerekir. İsterseniz bu dosyanın adını değiştirebilirsiniz. **main.cpp** de projenin olduğu dizine kaydedilmelidir.



6. Projeyi ve **main.cpp** dosyasını kaydettikten sonra artık C++ kodları yazmaya başlayabilirsiniz.

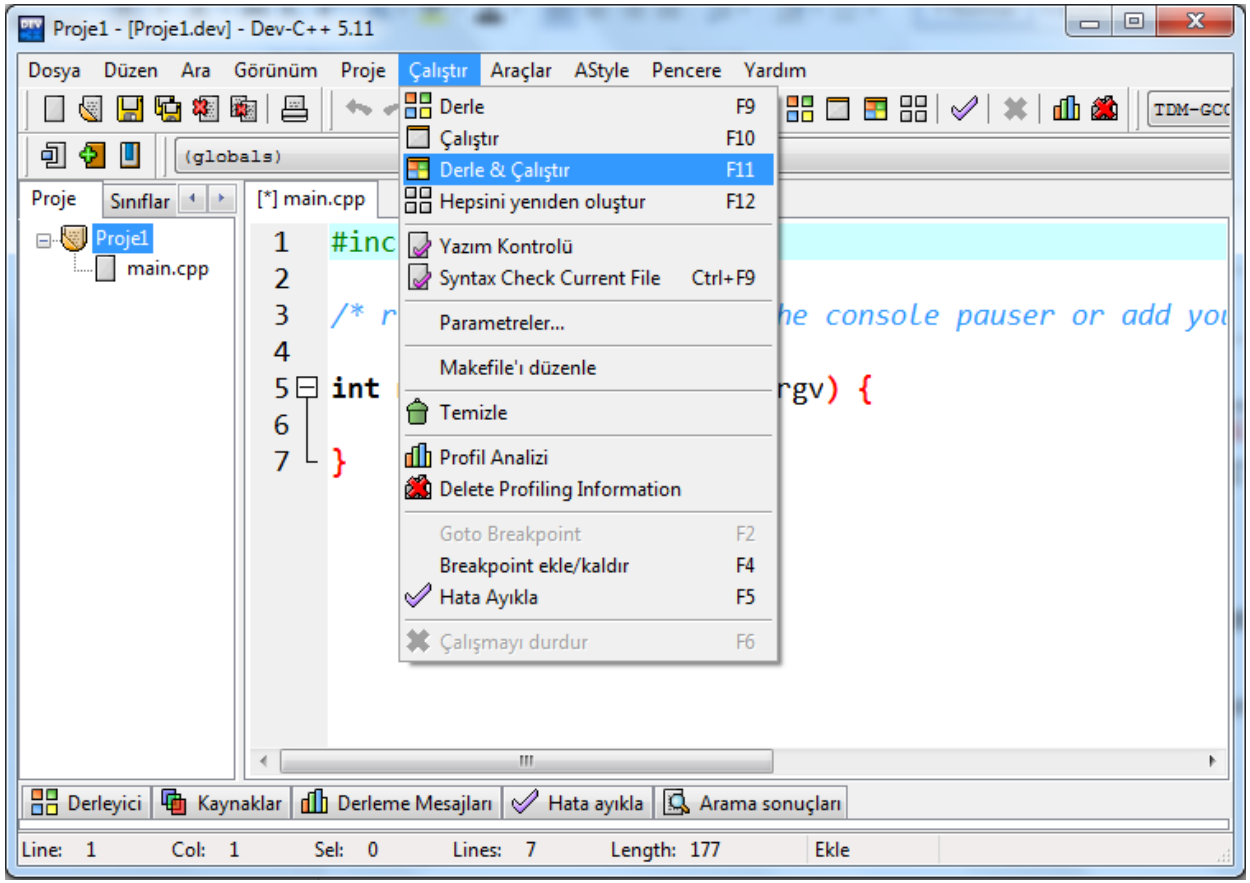


7. Kaydedilmiş bir projeyi açmak için menüde **File Open Project or File (Ctrl+O)** seçin.



8. Projeyi çalıştırmak için **Execute** **Run (Ctrl+F10)** veya yenden derlenmesi gerekiyorsa

Execute **Compile & Run (F11)** seçin.



İLK PROGRAM

C++ Temel Kuralları

Tüm C++ programları aşağıdaki kalıplaşmış satırları içerir.. (Bu üç kalıptan biri, bir değer döndürme durumuna göre tercih edilir.)

```
int main()
{
komut1;
komut2;
…….
return 0;
}
```

```
void main(void)
{
komut1;
komut2;
…….

}
```

```
main()
{
komut1;
komut2;
…….

}
```

Bir C Programının Genel Yapısı

Başlık (Header) Dosyaları (#include<….>)
Fonksiyon tanımlamaları
Global değişkenler

main ( )
{
Lokal değişkenler
Sabitler
Algoritma sıralı Komutlar
}

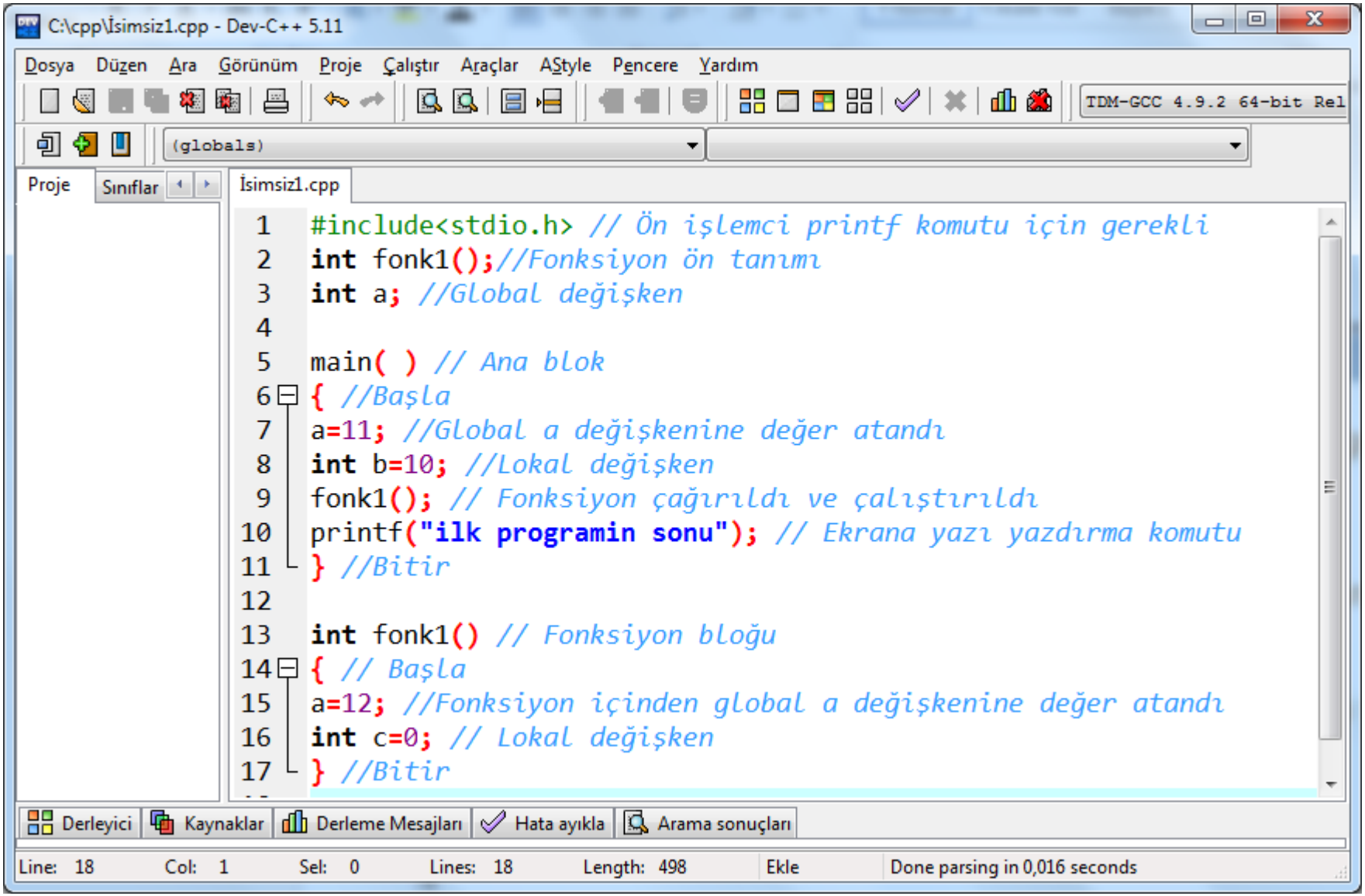
fonk ( )
{
}

Olması gereken mantığı tam bir örnek kod ile gösterirsek!

```
#include<stdio.h>
int fonk1//kullanılan her fonksiyon tanımlanmalıdır
int a; //kullanılan her değişken mutlaka tanımlanmalıdır

main( ) //Asıl program bloğu başlangıcı
{
a=11;
int b=10; //sadece main bloğunda tanımlama (yerel değişken)
fonk1(); //fonksiyonu çağırıyoruz
printf("ilk programin sonu");
} Asıl program bloğu bitişi

int fonk1() //fonksiyon başlangıcı
{
a=12;
int c=0;
} //fonksiyon bitişi
```


C:\cpp\İsimsiz1.cpp - Dev-C++ 5.11
Dosya Düzen Ara Görünüm Proje Çalıştır Araçlar AStyle Pencere Yardım
TDM-GCC 4.9.2 64-bit Rel
(globals)
Proje Sınıflar
İsimsiz1.cpp
1 #include<stdio.h> // Ön işlemci printf komutu için gerekli
2 int fonk1();//Fonksiyon ön tanımı
3 int a; //Global değişken
4
5 main( ) // Ana blok
6 { //Başla
7 a=11; //Global a değişkenine değer atandı
8 int b=10; //Lokal değişken
9 fonk1(); // Fonksiyon çağırıldı ve çalıştırıldı
10 printf("ilk programin sonu"); // Ekrana yazı yazdırma komutu
11 } //Bitir
12
13 int fonk1() // Fonksiyon bloğu
14 { // Başla
15 a=12; //Fonksiyon içinden global a değişkenine değer atandı
16 int c=0; // Lokal değişken
17 } //Bitir
Derleyici Kaynaklar Derleme Mesajları Hata ayıkla Arama sonuçları
Line: 18 Col: 1 Sel: 0 Lines: 18 Length: 498 Ekle Done parsing in 0,016 seconds

#include önişlemci komutu

#include, ilgili kaynak dosyanın derleme işlemine dahil edileceğini anlatan bir önişlemci komutudur. Bu komut ile önişlemci belirtilen dosyayı diskten okuyarak komutun yazılı olduğu yere yerleştirir. (metin editörlerindeki copy – paste işlemi gibi) Bu sayede kullanacağınız komutları tanımlamış olursunuz.

#include komutundaki dosya ismi iki biçimde belirtilebilir:
Açısal parantezlerle:

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

Çift tırnak içerisinde:

```
#include "stdio.h"
#include "deneme.c"
```

Dosya ismi eğer açısal parantezler içinde verilmişse, sözkonusu dosya önişlemci tarafından yalnızca önceden belirlenmiş bir dizin içerisinde aranır. Çalıştığımız derleyiciye ve sistemin kurulumuna bağlı olarak, önceden belirlenmiş bu dizin farklı olabilir.

Örneğin:
TC / INCLUDE
BORLAND / INCLUDE
C600 / INCLUDE gibi.

Benzer biçimde UNIX sistemleri için bu dizin, örneğin:
/USR/INCLUDE
biçiminde olabilir. Genellikle standart başlık dosyaları önişlemci tarafından belirlenen dizinde olduğundan, açısal parantezler ile kaynak koda dahil edilirler.

Dosya ismi iki tırnak içine yazıldığında önişlemci ilgili dosyayı önce çalışılan dizinde (current directory), burada bulamazsa bu kez de sistem ile belirlenen dizinde arayacaktır.

Örneğin:

C:\SAMPLE dizininde çalışıyor olalım.

#include "strfunc.h" komutu ile önilemci strfunc.h dosyasını önce C:\SAMPLE dizininde arar. Eğer burada bulamazsa bu kez sistem ile belirlenen dizinde arar. Programcıların kendilerinin oluşturdukları başlık dosyaları genellikle sisteme ait dizinde olmadıkları için iki tırnak içinde kaynak koda dahil edilirler.

#include ile koda dahil edilmek istenen dosya ismi dosya yolu da (path) de içerebilir:

#include <sysstat.h>

#include "c:headersmyheader.h"

....

gibi.

#include komutu ile yalnızca baslık dosyalarının koda dahil edilmesi gibi bir zorunluluk yoktur. Herhangi bir kaynak dosya da bu yolla kaynak koda dahil edilebilir.

```
//prog.cpp
#include "topla.c"
int main()     {
        toplam = a + b;
        printf("%d", toplam);
        return 0;     }

//topla.c
int a = 10;
int b = 20;
int toplam;
```

main()

C++ programlama dili nesneye dayalı bir dil olması dolayısıyla bazen yardımcı program parçalarını da bünyesinde kullanabilir. Eğer program bloğu main ile başlıyorsa devamındaki program bloğunun ana fonksiyon olduğunu anlatır.

C dili ile bilinilmesi gereken önemli noktalardan biriside C dili bütünüyle fonksiyonlardan oluşan bir dildir. En basit işlemleri yaptığımız kısım bile ana fonksiyon olarak adlandırılır, bilgisayar ilk bu fonksiyonu çalıştırmaya başlar, bu fonksiyonda eğer diğer fonksiyonlar çağırılmışsa onlara gider ve en son yine bu fonksiyonda program tamamlanır.

Duruma göre

{

Süslü parantez ana fonksiyonun yada programının başladığını göstermektedir.

Algoritma yaparken 1. Başla dediğimizde algılanan yapı taşıdır.

Bu ifade Pascal programlama dilinde Begin komutuyla aynı anlama gelmektedir.

printf

- Bu komut ekrana çıktı veren komuttur. Pascalda kullanılan writeln komutuyla aynı anlamdadır.

Değişkenler

Değişkene değer atama

- **Örnek:**

```
double x, y;
int i, k;
x = 2.8;
y = -1.4*x;
i = 9;
k = (i % 2) * (7 + 5*i);
```

Hesaplanmış değer soldaki değişkene atanır

Sağdaki işlemin sonucuhesaplanır

Değişkene değer atama

- Aynı anda tanımlama ve atama:

C++ da yeni bir değişken tanımlanırken aynı anda değer de atanabilir. (ilk değer atama)

Örnek;

```
double x = 1.0;
int i = 10, j = 20;
int k = i + j;
```

C++ ta değişkenleri kullanmadan önce tanımlamamız gerekir. Örneğin;

int alan**;**
float kare**;**

Aşağıdaki tabloda C ve C++ dillerine ait değişkenler özetlenerek verilmiştir.

Kullanım	Açıklama
char	Karakter
int	Tam sayı
short	Kısa tam sayı (Sayı tipi int)
Long	Uzun tam sayı (Sayı tipi int)
Unsiged char	İşaretsiz karakter
Unsigned	İşaretsiz tam sayı
Unsigned short	İşaretsiz kısa tam sayı (Sayı tipi int)
Unsigned long	İşaretsiz uzun tam sayı (Sayı tipi int)
Float	Reel sayı
Double	Çift reel sayı
Long double	Uzun çift reel sayı

Program içerisindeki konumlarına göre değişkenler

a) Lokal değişkenler.
Sadece tanımlandığı fonksiyonun içerisinde değer alabilen değişkenlerdir. Diğer fonksiyonlar içerisinde bu değişkenler kullanılamaz.

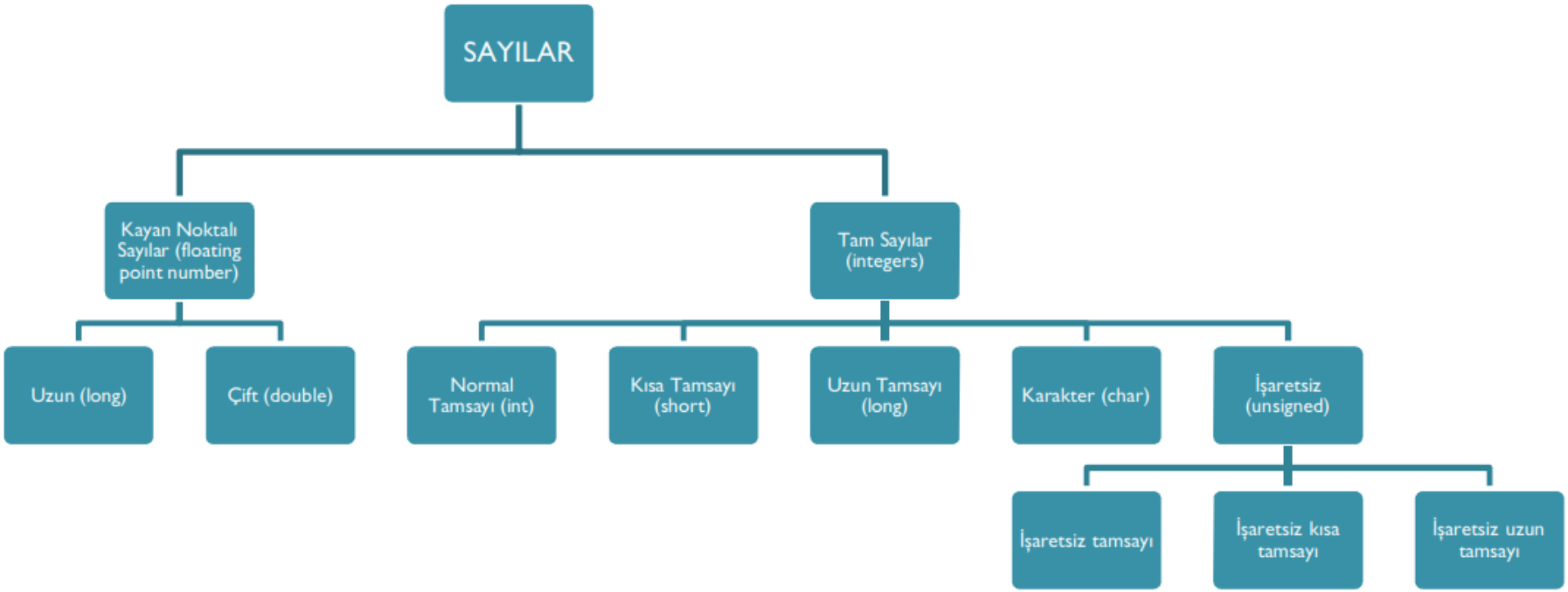
b) Global değişkenler.

Ana ve yardımcı fonksiyonların dışında tanımlanır. Bütün fonksiyonlarda bu değişkenler kullanılabilir.

C programında değişkenlerin genel prototipleri aşağıda verildiği gibidir.

değişken tipi değişken_1, değişken_2,...,değişken_n;

C'de kullanılan değişken tipleri aşağıda sıralanmıştır.


SAYILAR
Kayan Noktalı Sayılar (floating point number)
Tam Sayılar (integers)
Uzun (long)
Çift (double)
Normal Tamsayı (int)
Kısa Tamsayı (short)
Uzun Tamsayı (long)
Karakter (char)
İşaretsiz (unsigned)
İşaretsiz tamsayı
İşaretsiz kısa tamsayı
İşaretsiz uzun tamsayı

a) char tipi değişken (Küçük tamsayılar ve Karkterler)

-128 ile +127 arasındaki tam sayı işlemlerinin yapılabilmesinde kullanılır. Bellekte 1 baytlık alan kullanılır.

Örnek;
char x,y,z;
char sayi;
char rakam;

b) int tipi değişken (Tamsayılar)

Bu tip değişkenler -32768 ile +32767 arasındaki tamsayı değerleri için kullanılır. Bellekte 2 byte 16 bitlik alan kaplar.

Örnek;

```
int x,y,z;
int boy, sayı;
```

Gündelik hayatta kullandığımız gibi 10'luk tabandadırlar. C++ ta 8'lik ve 16'lık sayı tabanlarını da kullanabiliriz. Örneğin;

```
21 // Onluk taban
054 // 8'lik taban
0x5d // 16'lık taban
```

Aralarındaki farka dikkat ettiyseniz 10'luk tabandaki sayılar normal bir şekilde kullanılır iken 8'lik sayıların başında sıfır kullandık. 16'lık tabandaki sayıların kullanımında ise sıfır ile sayı arasında x karakterini görüyoruz.

c) float tipi değişken

Bu tip değişkenler ±3,4E-38 ile ± 3,4E+38 aralığındaki işlemlerde kullanılır. Bellekte 4 byte 32 bitlik alan kaplar.

Örnek;
float x,y,z;
float boy, sayı;

d) double tipi değişken

Bu tip değişkenler ±1.7E-308 ve ±1,7E+308 aralığındaki işlemlerde kullanılır. Bellekte 8 byte 64 bitlik alan kaplar.

Örnek;
double x,y,z,r;
double sonuç, toplam;

e) unsigned tip tamlayıcı

char ve int tipi tam sayı değişkenleri için kullanılır. Değişkenlerin sadece pozitif tanımlanması gereken durumlarda kullanılır.

unsigned char x,y,z; 0-255 arasında değer alabilir.
unsigned int x,y,z; 0-65535 arasında değer alabilir.

f) long tip tamlayıcı

int ve float ve double tipi değişkenlerinin aralıklarının artırılması için kullanılır.

long int x,yz;	4 bayt, -214 748 3648.....+214 748 3647
unsigned long int x,y,z;	4 bayt, 0.....+4294967295
long float x,yz;	8 bayt, ±1.7E-308 ..... ±1,7E+308
long double x,y,z;	10 bayt, ±3,4E-4932 .... ± 3,4E+4932

g) Sabitler

Program başlangıcında tanımlanırlar ve program içerisinde değiştirilemezler. Sabitlerin değerleri hiçbir zaman değişmez. Bunlar tam sayılar, küsüratlı sayılar, karakterler ve karakter dizileri manasına gelen stringler olabilir.

2 şekilde tanımlanabilir.

1) Define ile tanımlama

Başlık dosyalarından sonra tanımlanır. Program içerisinde değiştirilemez.
#define sabit_ismi sabitin alacağı değer yada ifade
#define PI 3.14

const

Sabit bildirimi, başlangıç değeri verilen değişken bildirimi gibi yapılır, ancak veri tipinin önüne *const* anahtar sözcüğü koyulmalıdır. Örneğin;

```
Const float pi= 3.142857 ;

Const double e= 2.718281 ;

Const int  EOF= -1 ;

Const char[ ]  =“Bir tuşa dokunun”;
```

gibi sabit bildirimleri gEçerli olup bunların içerikleri program boyunca değiştirilmez, yalnızca kullanılabilir.

Veri Tipi	Açıklama	Bellekte boyut (bayt)	Alt sınır	Üst sınır
char	Tek bir karakter veya küçük tamsayı	1	-128	127
unsigned char			0	255
short int	Kısa tamsayı için	2	-32,768	32,767
unsigned short int			0	65,535
int	Tamsayı için	4	-2,147,483,648	2,147,483,647
unsigned int			0	4,294,967,295
long int	Uzun tamsayı için	8	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
unsigned long int			0	18,446,744,073,709,551,615
float	Tek duyarlı gerçel sayı için (7 basamak)	4	-3.4e +/- 38	+3.4e +/- 38
double	Çift duyarlı gerçel sayı için (15 basamak)	8	-1.7e +/- 308	+1.7e +/- 308

Değişken Tanımlanırken şu kurallara dikkat edilmelidir.

Değişken adları bir harf ile başlamalıdır. (a-z, A-Z).

Değişken adı numara içerebilir. (0-9).

Değişken adında özel karakter kullanılmaz.

Değişken adında boşluk bırakılmaz.

Sadece İngilizce karakterler yer almalıdır. (ö, ü, ç, ş, ğ, ı olmamalı).

Değişkenler büyük ve küçük harf duyarlıdır. (sayi3 eşit değil Sayi3).

Değişkenin uzunluğu 32 karakteri geçmemelidir.

Değişken C/C++'a ayrılmış özel adlardan olmamalıdır.

Özel karakterler

’ ’ boşluk	, virgül	
! ünlem	**.** nokta	
; noktalı virgül	**:** iki nokta üst üste	
‘ tırnak	“ çift tırnak	
(sol parantez	**)** sağ parantez	
[sol köşeli parantez	**]** sağ köşeli parantez	
{ sol küme	**}** sağ küme	
| duvar	***/*** slaş	
**** ters slaş	~ tilda	
+ artı	# diyez	% yüzde
& ve	^ şapka	* yıldız
- Eksi	**=** eşit	**<** küçük
> büyük		

Özel Değişkenler (Ayrılmış Kelimeler)

break	**extern**	**sizeof**
case	**float**	**static**
char	**for**	**struct**
const	**goto**	**switch**
continue	**if**	**typedef**
default	**int**	**unsigned**
do	**long**	**void**
double	**return**	**while**
else	**short**	
enum	**signed**	

Geçerli Tanımlama:	**Hatalı Tanımlama:**
Sayi1	1ncisayi
Ogr_Numarasi	Ogr Numarasi
Adres	Genel#Toplam
OLCME	ölçme
Sayi1	1Sayi
a_switch	switch

Örnek 1

```
#include<stdio.h>
#include<conio.h>
#define PI 3.14
void main()
{
float r,alan,cevre;
r=1.5;
alan=PI*r*r;
cevre=2*PI*r;
printf("daire alanı = %5.2f\n",alan);
printf("daire çevresi=%5.2f", cevre);
getche();
}
```

```
#include<stdio.h>
#include<conio.h>
void fonk1();
int a,b,c;
void main()
{
clrscr();
a=10;
b=50;
fonk1();
printf(" a=%d b=%d c=%d",a,b,c);
}
void fonk1()
{
c=a*b;
}
```

printf (): Fonksiyonu

Sayısal ve alfanümerik değerleri ekrana (çıkış elemanı olarak tanımlı ise) göndermek için kullanılan formatlı çıkış fonksiyonudur. Bu fonksiyon stdio.h başlık dosyası altında tanımlıdır. Dolayısıyla fonksiyonunu kullanımı için stdio.h başlık dosyasının programa eklenmesi gerekir. Kullanım biçimi:

Genel yazım formatı;

printf(“format dizisi”,değer yada değişken listesi);

Örnek:

printf(“x değişkeninin değeri=%f= \n” , x);

printf(“Ahmet’in yaşı= %d \n” , 16);

printf(“a =10”); Ekrana a=10 ifadesini yazar. a değişkenin içeriği değişse bile ekrana a=10 yazılmaya devam edilir. a’nın program içerisinde aldığı değerin yazdırılması için format tanımlayıcı ile beraber kullanılması gerekir.

int k=12;
k değişkeni görüldüğü gibi int tanımlıdır. int değişkenler için format tanımlayıcı %d'dir.

Yani:
printf(" sonuc = %d", k);
%d olan yere k int sayısının sayısal değeri yazdırılacaktır.

Dönüşüm belirlemek için önce % karakteri ve ardından dönüşümün nasıl
olacağını belirten karakter verilir.

printf komut içinde tanımlayıcı kullanmak mecburidir tanımlayıcılar % işareti ile beraber kullanılır ve bu printf içinde

%[flags][width][.precision][length]specifier

Şeklinde kullanılır.

Bunlar %d, %f, %5.2l, %010d gibi görülebilir.

Değişken tipi tanımlama: SPECIFIER

specifier	Çıkış	Örnek
d or i	Tamsayı (int)	392
u	İşaretsiz tam sayı (0'dan başlar)	7235
o	İşaretsiz oktal	610
x	İşaretsiz heksadesimal	7 fa
X	İşaretsiz heksadesimal (Büyük karakterli)	7 FA
f	Ondalıklı sayı (float)	392.65
e	Gerçek sayıların bilimsel gösterimi (Küçük harf)	3.9265e+2
E	Gerçek sayıların bilimsel gösterimi (Büyük harf)	3.9265E+2
g	%e ve %f kullanılması gereken yerlerde kısa gösterim	392.65
G	%E ve %F kullanılması gereken yerlerde kısa gösterim	392.65
a	Heksadesimal ondalıklı sayı (Küçük harf)	-0xc.90fep-2
A	Heksadesimal ondalıklı sayı (Büyük harf)	-0XC.90FEP-2
c	Karakter	a
s	Karakter katarı (string)	sample
p	Gösterici adresi	b8000000
n	Bir şey yazdırmaz ancak girilen karakterin ne kadar uzakta olduğunun tespitinde kullanılabilir.	
%	% karakteri başka bir % karakteri ile devam ederse bu ekranda % işareti gösterilmesi içindir	%

```c
#include <stdio.h>
int main()
{
int val;
printf("blah %n blah\n", &val);
printf("val = %d\n", val);
return 0;
}
```

Ekran Çıktısı:

```
blah blah
val = 5
```

```c
int n;
printf("%s: %nFoo\n", "hello", &n);
printf("%*sBar\n", n, "");
```

Çıktı

```
hello: Foo
       Bar
```

Biçim tanımlayıcıları alt tanımlayıcılarda içerebilir ki bunlar: *flags*, *width*, *.precision* ve *modifiers* olarak adlandırılır

flags	description
-	Verilen bilgileri sola yaslı yazdırmak için kullanılır.
+	Sayısal işlemlerde yalnızca negatif sayıların işareti gösterilir. Bu bayrak sayesinde pozitif sayılarında işareti gösterilir.
(space)	Yazıldığı yere denk gelen kısma bir karakter boşluk karakteri ekler.
#	o, x or X tanımlayıcıları ile kullanılır. Bu sayede oktal decimal veya hexadecimal sayıların gerçek değerleri görülebilir.
0	Sol tarafa eklenen boşluk karakterleri yerine 0 karakteri eklemeye yarar

width	description
(number)	Minimum number of characters to be printed. If the value to be printed is shorter than this number, the result is padded with blank spaces. The value is not truncated even if the result is larger.
*	The *width* is not specified in the *format* string, but as an additional integer value argument preceding the argument that has to be formatted.

.precision	description
.number	Tam sayılar (d, i, o, u, x, X) ile: en az kaç dijit sayı yazılması gerektiğini gösterir. Eğer sayı dijitten kısa ise sağa yaslı ve ön tarafta boşluk dijitleri ile tamamlar. Ondalıklı sayılar (a, A, e, E, f and F)için: ondalıklı kısmın kaç basamak olması gerektiğini gösterir. g ve G tanımlayıcıları ile: Yazılabilecek maksimum dijit sayısını verir. Karakter katarı (s) ile: maksimum karakter katarı uzunluğunu verir. Eğer verilenden katar kısa ise öne boşluklar ekler.
.*	Bu bir biçimlendirme karakteri değildir. Tam sayı olarak girilen değeri ikinci sayıya boşluk olarak ekler.

```
printf ("Genişlik: %*d \n", 5, 10);
```

Çıktı→

Genişlik: 10

BİÇİMLİ YAZMA *precision kullanımı*

TAMSAYI DEĞİŞKENLER İÇİN

```
int x = 128 ;
printf("%7d",x) ;                    ….128
```

FLOAT DEĞİŞKENLER İÇİN

```
float x = 128.503 ;
printf("%7.2f",x) ;                  .128.50

float x = 85.47 ;
printf("%6.3f",x) ;                   85.470

printf("%06.1f",x) ;                  0085.5
```

STRING DEĞİŞKENLER İÇİN

```
s = "ABCDEF"
printf("%10s  ",s);                  . . . . A B C D E F
printf("%10.3s  ",s);                . . . . . . . A B C
```

Ekranda göremediğimiz veya gördüğümüz ama klavyenin üzerinde olmayan bazı karakter sabitleri de vardır. C++ ta bunları \ karakterinden sonra gelen bazı karakterler ile ifade ederiz. Bunlara Çıkış (Esc) karakterleri demekteyiz.

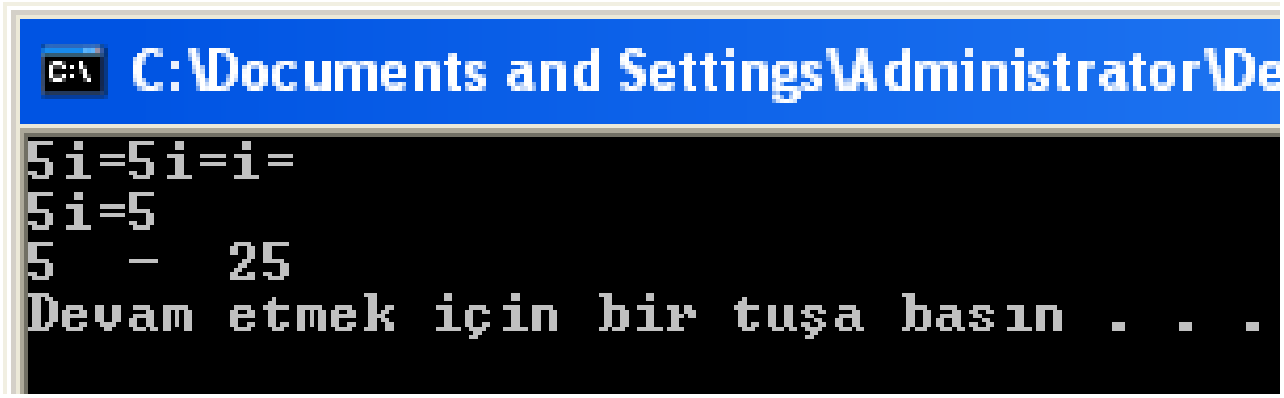
Karakter	Açıklama
\n	Yeni satır (Bir alt satıra iner)
\r	Aynı satır başı
\t	Tab karakteri kadar boşluk
\v	Dikey tab karakteri kadar boşluk
\b	İmleci geri götürür fakat değeri silmez
\a	Alarm
\’	Tek tırnak
\”	Çift tırnak
\?	Soru işareti
\\	Ters bölü

Örnekler:

Aşağıdaki program parçası çalıştırılmış olsa ekranda ne görülür?

```
int i = 5 ;
printf("%d",i);
printf("i=%d",i) ;
printf("i=") ;
printf("i=\n") ;
printf("%d",i) ;
printf("i=%d\n",i) ;
printf("%d  -  %d",i, 5*i);
system("pause");
```

ÇÖZÜM


C:\Documents and Settings\Administrator\De
5i=5i=i=
5i=5
5 - 25
Devam etmek için bir tuşa basın . . .

Not düşme (Açıklama satırı)

/* Açıklama Satırı: Program ile ilgili bilgiler yazılır. Bu program iki sayıyı toplar ekrana yazar 23 Şubat 2008 */

// Açıklama satırları yukarıda görüldüğü gibi /* açıklama*/ arasına yazılabilecegi gibi // iki slash kullanılarak da oluşturulabilir.

```c
#include <stdio.h>
#include<locale.h>
int main() {
setlocale(LC_ALL,"turkish");
printf ("Karakterler: %c %c \n", 'a', 65);
printf ("Tamsayılar: %d %ld\n", 1977, 650000L);
printf ("Boşluklarla uzunluk tanımlama: %10d \n", 1977);
printf ("Sıfırlarla uzunluk tanımlama: %010d \n", 1977);
printf ("Bazı değişik tanımlamalar: %d %x %o %#x %#o \n", 100,
100, 100, 100, 100);
printf ("Ondalıklı sayı tanımlama: %4.2f %+.0e %E \n", 3.1416,
3.1416, 3.1416);
printf ("Genişlik: %*d \n", 5, 10);
printf ("%s \n", "A string");
return 0 ;
}
```

örnek

```
#include<stdio.h>
#include<conio.h>
main()   {
char a;
float e;
int b;
long float f;
unsigned int c;
double g;
long int d;
a=10; e=12.2;
b=270; f=12.333;
c=50000; g=233.1111111;
d=2000000000;

printf("a'nin değeri=%d\n",a);
printf("b'nin değeri=%i\n",b);
printf("c'nin değeri=%u\n",c);
printf("d'nin değeri=%ld\n\n",d);
printf("e'nin değeri=%f\n",e);
printf("f'nin değeri=%lf\n",f);
printf("g'nin değeri=%lf\n",g);

}
```

Örnek

```
#include<stdio.h>
#include<conio.h>
#define carp(a,b) (a*b)
void main()
{
int x,y,z;
x=99;y=3;
z=carp(x,y);
printf("Çarpım=%d",z);
getche();
}
```

Örnek

```
#include<stdio.h>
#include<conio.h>
#define YAZ printf("z= %d\n",z)
void main()
{
int x,y,z;
x=15;
y=6;
z=x+y;YAZ;
z=x-y;YAZ;
z=x/y;YAZ;
z=x*y;YAZ;
z=x%y;YAZ;
}
```

Cout Komutu (C++)

Cout komutu ile tırnak işaretleri arasında yazılanlar ekrana bastırılır. Cout komutu ile << karakterleri birlikte kullanılır.

Örnek:

```
1. # include <iostream>
2. # include <string.h>
3. int main()
4. {
5.       cout << “ilk programım.”;
6.       return 0;
7. }
```

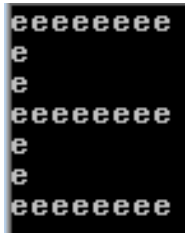
Ayrıca cout << komutu programda belirlenen ifadenin ekranda gösterilmesini sağlar.

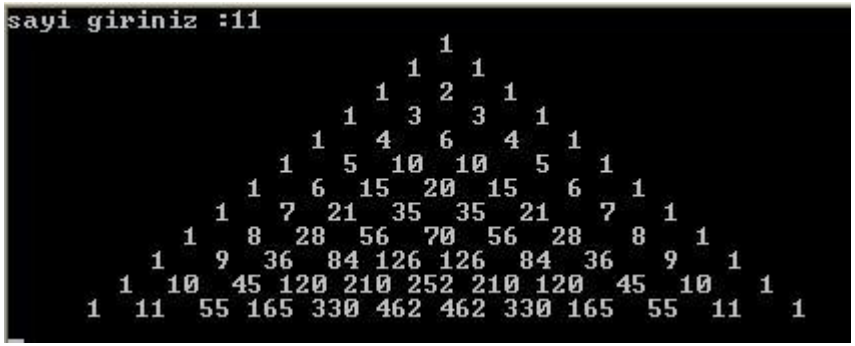
Örnek:

```
int a = 20;
cout << a;
```

ÖDEV 2

1. AŞAĞIDAKİ EKRAN GÖRÜNTÜLERİNİ AYNEN ELDE EDİNİZ





2. YUKARIDAKİ ÖRNEKTE GÖRÜLEN E HARFİNİ BENZER ALARAK İSMİNİZİ YAZDIRINIZ