**Yorum Satırları**

Bölü-yıldız (/\*) ile ondan sonraki ilk yıldız-bölü (\*/) arasında bulunan herhangi bir  
karakter dizisi bir program açıklamasıdır ve derleyici açısından bir boşluğa eşdeğer  
olarak kabul edilir.   
/\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*\  
\* \*  
\* Buyuk bir aciklama obegi... \*  
\* \*  
\\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*/  
Daha kısa bir yorum için (//) şeklinde yorum satırları kullanılabilir.

**Anahtar Sözcükler**

C dilinde 32 adet anahtar sözcük vardır; hepsi küçük harfle yazılır. Anahtar sözcükler tanıtıcı sözcük olarak kullanılamazlar; kendilerine özgü kullanım alanları vardır.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **veri tipi** | | **bellek sınıfı** | | |  | **deyim** | | **işleç** | |
|  |  |  |  |  |  |  |  |  |  |
| **char** | **auto** | **break** | **sizeof** | |
| **const** | | **extern** | | | **case** | | |  |  |
| **double** | | **register** | | | **continue** | | |  |  |
| **enum** | |  | **static** | | **default** | | |  |  |
| **float** | | **typedef** | | | **do** | | |  |  |
| **int** | |  |  |  |  | **else** | |  |  |
| **long** | |  |  |  |  | **for** | |  |  |
| **short** | |  |  |  |  | **goto** | |  |  |
| **signed** | |  |  |  | **if** | | |  |  |
| **struct** | |  |  |  | **return** | | |  |  |
| **union** | |  |  |  |  | **switch** | |  |  |
| **unsigned** | |  |  |  | **while** | | |  |  |
| **void** | |  |  |  |  |  |  |  |  |
| **volatile** | |  |  |  |  |  |  |  |  |

**Temel Veri Tipleri Ve Tanımlar**

Bir C programında, kullanılmadan önce, tüm değişken ve fonksiyonların tanımı veya  
bildirimi yapılmalıdır. Temel veri tiplerinin, yani tamsayı, kayan noktalı sayılar ve  
karakterlerin bildiriminde kullanılan anahtar sözcükler şunlardır:

| **DATA TYPE** | **MEMORY (BYTES)** | **RANGE** | **FORMAT SPECIFIER** |
| --- | --- | --- | --- |
| short int | 2 | -32,768 to 32,767 | %hd |
| unsigned short int | 2 | 0 to 65,535 | %hu |
| unsigned int | 4 | 0 to 4,294,967,295 | %u |
| int | 4 | -2,147,483,648 to 2,147,483,647 | %d |
| long int | 8 | -2,147,483,648 to 2,147,483,647 | %ld |
| unsigned long int | 8 | 0 to 4,294,967,295 | %lu |
| long long int | 8 | -(2^63) to (2^63)-1 | %lld |
| unsigned long long int | 8 | 0 to 18,446,744,073,709,551,615 | %llu |
| signed char | 1 | -128 to 127 | %c |
| unsigned char | 1 | 0 to 255 | %c |
| float | 4 |  | %f |
| double | 8 |  | %lf |
| long double | 16 |  | %Lf |

**Include Emri**

‘#include emri’ ya

#include "dosya\_adı" ya da

#include < dosya\_adı>  
şeklinde olur ve önişlemciye #include satırının yerini, belirtilen dosyanın alacağını gösterir. Eğer dosya\_adı çift tırnak içine alınmışsa, o zaman önişlemci, dosyayı kaynak program dosyasının saklandığı altdizinde arar. Eğer dosya\_adı açılı parantezler içinde ise,  
o zaman dosya, böyle #include emirleri için aramaların yapıldığı “standart” altdizin(ler)de aranır. Normalde, bu \include altdizini olur.

**Formatlı Ve Formatsız Input Output**Formatlı input-output kullanıcının anlamlandırabildiği giriş çıkışlardır. Bunlara printf() ve scanf() fonksiyonları örnek verilebilir.

Formatsız input-output bilgisayarların anlayabileceği giriş çıkışlardır. Bunlara getchar() ve putchar() örnek verilebilir.

**printf() Fonsiyonu**

Prototipi “ int printf( const char \*restrict format, ... ); ” şeklindedir. Ekrana yazı yazar. Geri dönüş değeri int dir.

**scanf() Fonsiyonu**

Prototipi “int scanf(const char \*format, ...); ” şeklindedir. Kullanıcıdan veri alır. Geri dönüş değeri int dir.

**getchar() ve putchar() Fonksiyonları**

Bu fonksiyonlar bir karakterin girilmesini veya çıktısının yapılmasını sağlarlar. Tipik bir kullanım şöyledir. c’nin char veya int olduğunu farzedin.

c = getchar();

putchar(c);

c ‘nin aldığı değeri putchar() ile yazdırırsak çıktıda ASCII tablosundaki numaragösterilir.