1

Biçem Belirteçleri (Format Specifiers)

Bilgisayara, girdiler ve çıktılar insanın anlayacağı biçemdedir. Harfler, sayılar ve diğer krakterler, kültürlere bağlı olan simgelerdir. Bir bakıma, onlar birer resimdir, birer ikondur. Hiçbir editör onları olduğu gibi kaydedemez. Üstelik bilgisayar, insanın kullandığı karakterleri ve kolay algılamak için kullandığı biçemleri anlamaz. Örneğin, çok haneli bir sayıyı kolay algılamak için, latin alfabesinde, sayıyı binliklerine ayırıp yazarız. Bu biçemler, bilgisayarın anlayacağı biçemler değildir.

Hangi alfbeyi ve hangi editörü kullanırsak kullanalım, bilgisayara gönderdiğimiz veriler onun anlayacağı ikili sisteme (binary system) dönüşür. Çıktılar, gene bizim anlayabilmemiz için, kendi kültürümüzde kullandığımız alfabeye ve yazım biçemine dönüştürülür. Bu işi yapan özel yazılımlar vardır.

Bilgisayara (daha doğrusu dönüşümü yapan yazılıma ve fonksiyonlara) giren ve çıkan verinin tipini belirtmek gerekir. C dilinde, bilgisayara girilecek verinin tipini belirtmek için, altı tane tip belirteci kullanırız: %c, %d, %f, %s, %u %ld. Bunlar temel veri tiplerini belirtmeye yararlar. Bunlara ek olarak %e, %g, %G, %o, %p, %x, %X belirteçleri de kullanılır.

Tablo 1.1'de başlıcaları listelenen biçem belirteçlerini örnekler üzerinde inceleyecek ve nasıl kullanıldıklarını örneklerle göreceğiz.

Standart giriş birimi dediğimiz klavyeden girişte, scanf() fonksiyonunu kullanıyoruz. Girilecek verinin tipini belirtmek için scanf() fonksiyonunda parametre olarak, tablodaki ilk altı belirteç yeterlidir.

Standart çıktı birimi dediğimiz ekrana ya da yazıcıya bilgisayardan gelen çıktıları *printf()* fonksiyonu ile yazdırıyoruz. Bu ad *print format*'ın

kısaltmasıdır. Adından anlaşıldığı gibi, *printf()* fonksiyonu, çıktının veri tipini belirler ve onu insanın anlayacağı biçeme dönüştürür.

İlk iş, ana bellekte binary formatta yazılı olan verinin tipini belirlemektir. Çünkü, ana bellekte yazılı olan bir binary veriyi *int, float, char vb* olarak yorumlamak mümkündür. *Bunu printf()* fonkiyonunda parametre olarak kullanılan biçem belirteçleri yapar. Dolayısıyla, bu operatörlerin ilk işleri, binary veriyi istenen veri tipine dönüştürmektir.

Sözkonusu dönüşüm eylemi birincil önemde olmasına karşın, çıktıyı kolay algılamamıza yetmeyebilir. Özellikle sayısal tiplerin, içinde yaşadığımız kültürdeki yazılış biçmine girmesini isteriz. Bu eylemi gene biçem belirteçlerine yaptırmak için, onlara yeni işlevler yükleriz. Örneğin, çıktıyı sağa ya da sola yanaşık yaz, sayıları binliklerine ayır, kesir hanesini 2 yap, vb Dönüştürme eylemi tek olsa bile, dönüştürülen verinin yazılış biçimi de, insanın algılamasına etki eder. O nedenle tablodaki ilk altı tip belirtecini, aynı zamanda biçemleyici olarak kullanmamıza olanak sağlar.

Belirteç	*Açıklama*
%c char	Tek karakter
%d (%i)	int, signed integer
%f	float
%s	array karakter dizimi (string)
%u	int unsigned decimal
%ld	long double
%e (%E)	float , double üstel
%g (%G)	float , double
%o	unsigned octal
%p	pointer adresi
%x (%X)	i unsigned hex

Tablo 1.1: Biçem Belirteçleri (Format Specifiers)

Program 1.1.

```
1 void main()
  {
    int i=10;
    printf("%d",i);
   }

  /**
  output
  10
  */
```

Program 1.2.

```c
void main()
{
  char ch='a';
  printf("%c",ch);
 }

/**
output
a
*/
```

Program 1.3.

```c
void main()
{
  float f1=32.156;
  printf("%f",f1);
 }

/**
output
32.156
*/
```

Program 1.4.

```c
void main()
void main()
{
  float f1=32456.156;
  printf("%e",f1);
 }

/**
output
3.2456156 e+004
*/
```

Program 1.5.

```c
void main()
{
  printf("%g",123.22345);
 }

/**
output
123.223
*/
```

Program 1.6.

```c
void main()
{
  int a=10;
  printf("%i",a);
 }

/**
output
10
*/
```

Program 1.7.

```c
void main()
{
  int a;
 printf("%u",&a);
 }

/**
output
2686854  // değişkenin bellek adresidir
*/
```

Program 1.8.

```c
void main()
{
 printf("%o",8);
 }

/**
output
10
*/
```

Program 1.9.

```c
void main()
{
 printf("%x",10);
 }

/**
output
a
*/
```

Program 1.10.

```
void main()
{
 int  a;
 printf("hello%n",&a); // now 'a' will be assigned with number of
  characters in message ( "hello" )
 printf("%d",a);
}

/**
output
hello5
*/
```

Program 1.11.

```
void main()
{
 char ch[20];
 printf("enter your name:");
 gets(ch);
 printf("%s",ch);
}

/**
output
enter your name: amit verma
amit verma
*/
```

Program 1.12.

```
void main()
{
  int i=32769;
  printf("%ld",i);
 }

/**
output
32769
*/
```