7.Bölüm

FONKSİYONLAR

Programın farklı yerlerinde benzer işlemler yapılması gerekiyorsa gerekli komutlar tekrarlanmalıdır. Fonksiyonlar bu konuda yardımcı olurlar. Fonksiyonlar programınızı ayrı ve kolay anlaşılabilir parçalara bölmenize olanak sağlarlar.

FONKSİYON BİLDİRİMLERİ

Bütün fonksiyonlar iki parçadan oluşurlar;

- bildirim
- gövde

Basit bir fonksiyon bildirimi olan main( )'i gördük. Bunu izleyen blok, main( ) fonksiyonunun gövdesidir.

Fonksiyonların genel sintaksı aşağıdaki gibidir.

```
tip fonksiyon_ismi (parametre1, parametre2, ..., parametreN)
parametre bildirimleri;
blok;
```

Tip fonsiyonun verceği değerin tipini gösterir.

ÖRNEK

```
# include <stdio.h>
merhaba_de( )
{
printf("Merhaba\n");
printf("Bu bir fonksiyon ile yazıldı\n");
}
main( )
{
printf("Bir defa çağır\n");
merhaba_de( );
printf("Bir kaç defa çağır\n");
merhaba_de( );
merhaba_de( );
```

```
}
```

Fonksiyon bildirimi main( ) fonsiyonundan önce yapılmıştır. Fonksiyon yalnızca fonksiyon ismi, parantezler ve bir ;'den oluşan tek bir tümce kullanılarak çağrılabiliyor.

Derleyici ;' fonksiyon bildirilerini fonksiyon çağrılarından ayırmak kullanır.

RETURN TÜMCESİ

Bir fonksiyondan kendisini çağıran programa geri dönmek için return tümcesi kullanılır. Return'u herhangi bir terim izleyebilir.

```
Return (terim)
```

şeklindedir. Çağıran fonksiyon gerekirse gönderilen terimi göz ardı edebilir.

```
int toplam(a, b)
int a,b;
{
int c;
c=a+b;
return c;        }
```

Bu program ana programdan iki değer alır (a ve b) bunlar toplanır ve sonuç c'ye yerleştirilir. Sonra fonksiyon bu sonucu ana programa göndererek fonksiyonu sona erdirir.

PARAMETRELER

Parametre program ile fonksiyon arasında bilgi aktarmakta kullanılan özel bir değişkendir. Parametre tanımlamaları fonksiyon isminden hemen sonra tanımlanır. Fonksiyon isminden önce tip bildirimi yapılır, bu fonksiyonun ana programa hangi tipte değer göndereceğini belirtir.

Herhangi bir değer vermeyen bir fonksiyonun tipi void olarak bildirilir. Bir fonksiyonun gönderdiği değerleri kullanabilmek için bir değişkeni fonksiyonun ismine eşitlemek gerekir.

ÖRNEK

```
# include <stdio.h>
int toplam(a, b)
int a, b;
{
int c;
c=a+b;
return c;
}
main( )
```

```
{
int a;
a=toplam(3, 4);
printf("toplam= %d\n", a);
}
```

Bir bildirimde tip belirtilmezse C int tipi bir değer istediğimizi varsayar. Bununla birlikte bazı hatalarla karşılaşmamak için tip belirtilmelidir.

Fonksiyon bir değer vermesini istemiyorsanız void sözcüğünü kullanmalısınız.

YEREL DEĞİŞKENLER

Yukarıdaki örnekte toplam( ) fonksiyonundaki a, main( )'deki a'dan farklıdır. Çünkü a yerel bir değişkendir. Bu değişken ve parametreler yalnızca bildirdikleri fonksiyonun içinde tanınabilirler. Hiç bir fonksiyon başka bir fonksiyonun içinde bildirilen parametre ve değişkenleri tanıyamaz. Bu nedenle bir programda aynı isimde birden fazla değişken kullanılabilir.

PROTOTİP

Prototip C'ye fonsiyonun formatını gösterir. C'de bunu kullanır ve fonksiyon çağrısı yapılmadan önce bütün değerlerin uygun tiplerine çevrilmelerini sağlar. Bildirim şekli şöyledir;

```
tip fonsiyon_ismi(tip,tip, ...,tip);
```

Bir bildirimden farklı olarak prototipin sonuna bir noktalı virgül konulmalıdır.

```
İnt toplam(int, int);
```

gibi

ÖRNEK

```
# include <stdio.h>
int toplam(int,int);
int toplam(a,b)
int a,b;
{
int c;
c=a+b;
return c;        }
main( )
{
int a;
a=toplam(3.1,4.2);
printf("a'nın değeri= %d\n", a);
```

```
}
```

parametre kullanmayan fonksiyonlar için

```
void merhaba_de(void);
```

şeklinde prototip tanımlanır.

İÇ İÇE FONKSİYONLAR

```
# include <stdio.h>
int toplam(int,int);
void bekle(void);
toplam(a,b)
int a,b;
{
int c;
printf("%d ile %d toplanacak\n",a,b);
bekle( );
c=a+b;
return c;
}
void bekle(void)
{
printf("Devam etmek için herhangi bir tuşa basın\n");
getch( );
}
main( )
{
int x,y;
y=5;
x=toplam(4,y);
printf("sonuç=%d\n",x);
}
```