

PROGRAMLAMA TEMELLERİ

BÖLÜM 9

ÖĞR. GÖR. HAKAN CAN ALTUNAY

ÖZYİNELEMELİ (RECURSIVE) FONKSİYONLAR

Direk veya dolaylı olarak kendi kendini çağıran fonksiyonlara özyinelemeli fonksiyon denir. Pek çok algoritmanın özyinelemeli olarak yazılması kolay ve anlaşılırdır. Döngüler kullanılarak yazılan tüm uygulamalar özyinelemeli fonksiyon kullanarak da gerçekleştirilebilir.

Pozitif bir tam sayının faktöriyelini alan program aşağıda gösterilmiştir.

```
#include <stdio.h>
int fakt(int); /* fonksiyon ön bildirimi */
void main()
{   int n;
    printf("Pozitif bir tam sayi giriniz:"); scanf("%d", &n);
    printf("Sayinin faktoriyeli: %d\n", fakt(n));
}

int fakt(int i)
{   if (i > 1)
        return i*fakt(i-1);
    else
        return 1;
}
```

Başka bir örnek;

```
#include <stdio.h>
void fonk(int); /* fonksiyon ön bildirimi */

void main()
{   int n;
    printf("Pozitif bir tam sayi giriniz: ");
    scanf("%d", &n);
    fonk(n);
}

void fonk(int i)
{   printf("Gelen sayi: %d\n", i);
    if (i > 1)
        fonk(i - 1);
}

Örnek program çıktısı:
Pozitif bir tam sayi giriniz: 4
Gelen sayi: 4
Gelen sayi: 3
Gelen sayi: 2
Gelen sayi: 1

printf fonksiyonunun if koşulunda sonraya atılması durumunda;
void fonk(int i)
{   if (i > 1)
        fonk(i - 1);
    printf("Gelen sayi: %d\n", i);
}

Örnek program çıktısı:
Pozitif bir tam sayi giriniz: 4
Gelen sayi: 1
Gelen sayi: 2
Gelen sayi: 3
Gelen sayi: 4
```

DİZİLERİ FONKSİYONLARA GÖNDERME

Fonksiyonlara gönderilen parametre değerlerinin fonksiyon içerisinde değiştirilmesi mümkün değildir. Fonksiyona sadece değişkenin değeri gönderilir. Fonksiyon içerisindeki yapılan işlemlerden parametrenin kendisi etkilenmez. Dizilerde ise durum farklıdır. Dizi fonksiyona gönderilirse elemanlar değiştirilebilir. Dizinin sadece herhangi bir elemanı gönderildiğinde ise değer değiştirilemez. Diziyi fonksiyona gönderirken sadece adını parametre olarak yazmak yeterlidir.

Aşağıdaki örnekte dizi eleman değerlerinin fonksiyon içerisinde nasıl değiştiği gösterilmiştir.

```
#include <stdio.h>
void kareleri(int []);
void main()
{   int a[10];
    int i;
    for (i=0; i<10; i++)
        a[i] = i + 1;
    printf("Dizinin eleman degerleri:\n");
    for (i=0; i<10; i++)
        printf("%d ",a[i]);
    kareleri(a);
    printf("\nKare alma islemi sonrasi degerler:\n");
    for (i=0; i<=9; i++)
        printf("%d ",a[i]);
}

void kareleri(int a[])
{   int i;
    for (i=0; i<=9; i++)
        a[i] = a[i] * a[i];
}
```

Dizi elemanlarının teker teker gönderilmesi durumunda ise aynı sonucu almak için aşağıdaki örnekte gösterildiği gibi elemanları adres değerleriyle birlikte göndermek gerekir.

```

#include <stdio.h>
int kareleri(int);
void main()
{
    int a[10];
    int i;
    for (i=0; i<10; i++)
        a[i] = i + 1;
    printf("Dizinin eleman degerleri:\n");
    for (i=0; i<10; i++)
        printf("%d ",a[i]);
    for (i=0; i<10; i++)
        a[i] = kareleri(a[i]);
    printf("\nKare alma islemi sonrasi degerler:\n");
    for (i=0; i<=9; i++)
        printf("%d ",a[i]);
}

int kareleri(int b)
{
    return b * b;
}

```

FONKSİYONLARI REFERANS İLE ÇAĞIRMA

Fonksiyonlar çağrılırken parametrelerin bir kopyası çıkartılıp fonksiyona gönderilir. Bir fonksiyonun birden fazla değer döndürebilmesi için pointerlara (işaretçilere) ihtiyaç duyulur.

```

#include <stdio.h>
int kare(int);
void main()
{
    int i = 5;
    printf("Oncesi :%d\n", i);
    printf("Karesi :%d\n", kare(i));
    printf("Sonrasi:%d\n", i);
}

int kare(int k)
{
    k++;
    int kare;
    kare = k * k;
    return kare;
}

```

Program çıktısı:
 Oncesi :5
 Karesi :36
 Sonrasi:5

Yukarıdaki örnekte gönderilen parametrenin kopyası fonksiyona gönderildiği için fonksiyon içerisinde yapılan değişiklikler fonksiyonun çağrıldığı yeri etkilemez. Eğer gönderilen parametredeki değişikliklerin çağrıldığı yerde de

geçerli olması isteniyorsa parametreyi adresiyle birlikte göndermek gerekir. Bu nedenle o fonksiyonda bu adresi karşılayan bir pointer değişkene ihtiyaç duyulur.

```
#include <stdio.h>
int arttir(int *);
void main()
{
    int i = 5;
    printf("Oncesi  %d\n", i);
    printf("Karesi  %d\n", arttir(&i));
    printf("Sonrasi %d\n", i);
}

int arttir(int *k)
{
    (*k)++;
    int kare;
    kare = *k * *k;
    return kare;
}

Program çıktısı:
Oncesi :5
Karesi :36
Sonrası:6
```

KAYNAKLAR:

- 1- ALGORİTMA GELİŞTİRME VE PROGRAMLAMAYA GİRİŞ, FAHRİ VATANSEVER, SEÇKİN YAYINEVİ
- 2- VERİ YAPILARI VE ALGORİTMALAR, RİFAT ÇÖLKESEN, PAPATYA BİLİM YAYINEVİ
- 3- ALGORİTMA VE PROGRAMLAMA MANTIĞI, BURAK TUNGUT, KODLAB