

# **Bilgisayar Bilimlerine Giriş-II**

**-3-**

**BIL 1002**

**Dokuz Eylül Üniversitesi, Fen Fakültesi,  
Bilgisayar Bilimleri Bölümü**

# const Belirtecini Göstericilerle Kullanmak

## ► const belirteci

- Değişken değiştirilemez.
- const eğer fonksiyonun değişkeni değiştirmesine gerek yoksa kullanılır.
- Eğer bir const değişken değiştirilmeye çalışılırsa derleyici bunu yakalar ve hata mesajı verir.

## ► const göstericiler

- Sabit bir hafıza noktasını gösterirler.
- Tanımlanırken ilk değer atanmalıdır.

<code>int *const myPtr=&amp;x;</code>	//Bir tamsayı için sabit gösterici
<code>const int *myPtr=&amp;x;</code>	//Sabit bir tamsayı için sabit olmayan gösterici
<code>const int *const Ptr=&amp;x;</code>	//Sabit bir tamsayı için sabit bir değişken; x'in kendisi değişse bile *Ptr değişmez.

/\*Sabit bir veriyi gösteren sabit olmayan bir gösterici kullanılarak bir stringin karakterlerini sırayla yazdırmak \*/

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

```
#include <conio.h>
```

```
void printCharacters( const char * );
```

```
int main()
```

```
{
```

```
    char string[] = "String karakterleri yaz";
```

```
    printf( "String:\n" );
```

```
    printCharacters(string);
```

```
    printf("\n");
```

```
    getch();
```

```
    return 0;
```

```
}
```

```
void printCharacters( const char *sPtr )
```

```
{
```

```
    for ( ; *sPtr != '\0'; sPtr++ )
```

```
        printf( "%c ", *sPtr );
```

```
}
```

/\*Sabit bir veriyi sabit olmayan gösterici kullanarak değeri değiştirmeye çalışmak \*/

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

```
void f( const int * );
```

```
int main()
```

```
{
```

```
    int y;
```

```
    f( &y );
```

```
    return 0;
```

```
}
```

```
void f( const int *xPtr ) /*bir tamsayı sabitini göstren gösterici*/
```

```
{
```

```
    *xPtr = 100; /* const nesnesi değiştirilemez*/
```

```
}
```

```
/*Sabit olmayan bir veriyi gösteren  
sabit bir göstericiyi değiştirmeye  
çalışmak */
```

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

```
int main()
```

```
{
```

```
    int x, y;
```

```
    int * const ptr = &x; /*ptr tamsayı  
gösteren bir göstericidir*/
```

```
    *ptr = 7;
```

```
    ptr = &y;
```

```
    return 0;
```

```
}
```

```
/* Sabit bir veriyi gösteren sabit bir  
göstericiyi değiştirmeye çalışmak*/
```

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

```
int main()
```

```
{
```

```
    int x = 5, y;
```

```
    const int *const ptr = &x;
```

```
    printf( "%d\n", *ptr );
```

```
    *ptr = 7;
```

```
    ptr = &y;
```

```
    return 0;
```

```
}
```

# Tipi İşaretçi olan Fonksiyonlar

- Bir fonksiyon tanımlanırken parametreleri işaretçi olduğu gibi, tipi de işaretçi olabilir. Bu o fonksiyonun kendisini çağırana adres göndereceği anlamına gelir.

```
#include<stdio.h>
int *gonder(int a[],int n)
{
    int enk,*p,i;
    enk=a[0];
    p=&a[0];

    for(i=1;i<n;i++)
        if(a[i]<enk){
            enk=a[i];
            p=&a[i];
        }
    return p;
}
```

Dizinin en küçük  
elemanının adresini  
yollayan program

```
int main()
{
    int dizi[]={ 10,20,15,30,25,32,33 };
    int *q;

    q=gonder(dizi,7);
    printf("en kucuk eleman=%d",*q);
    return 0;
}
```

# Dinamik Hafıza Tahsisi

- ▶ Bir dizinin bir programın başında kaç elemanlı olduğu verilerek bildirilirse, derleyici o dizi için gerekli bellek alanını saklı tutar ve o alan başka amaçlar için kullanılamaz. Bu durum programın çalışması süresince dizinin gerekmediği hallerde bile geçerlidir.
- ▶ Dizilerde dinamik çalışma programın yürütülmesi sırasında, dizi için gerekli bellek alanının işletim sisteminden istenmesi ve işi bittiğinde geriye verilmesidir. Bu işi yapabilmek için kütüphanede bulunan bazı fonksiyonlara gereksinim vardır.

# Statik ve Dinamik Dizi

## StatikDizi

- ▶ Bir C programı içerisinde, dizilerin boyutu ve kaç elemanlı olduğu program başında belirtilirse, derleyici o dizi için gereken bellek alanını (bölgesini) program sonlanıncaya kadar saklı tutar ve bu alan başka bir amaç için kullanılamaz.
- ▶ Bu türdeki diziler *statik dizi* olarak adlandırılırlar.
- ▶ Statik dizinin boyutu programın çalışması esnasında değiştirilemez.

# Statik ve Dinamik Dizi

## Dinamik Dizi

- ▶ Programın çalışırken bir dizinin boyutu ve eleman sayısı bazı yöntemler kullanılarak değiştirilebilir. Bu tür dizilere *dinamik dizi* denir.
- ▶ Dinamik diziler için gereken bellek bölgesi, derleyici tarafından işletim sisteminden **istenir**, **kullanılır** ve daha sonra istenirse bu bölge **boşaltılır**.



# Örnek:

`char ad[20];`

- ▶ İfadesinde derleyici, bellekten 20 byte boyutunda sürekli bir alan tahsis edecektir.
- ▶ Bu yer tahsisi, program başlatılmadan önce yapılmaktadır.
- ▶ Yani program çalışırken bu dizinin boyutunu değiştirmeniz mümkün değildir.
- ▶ Fakat bazı durumlarda bellekten boyutu sabit olmayan ve sürekli değişebilen yerler tahsis etmemiz gerekecektir.

# Dinamik Dizi Fonksiyonları

***void\** malloc(*boyut*)**

*boyut* byte kadar yeni bellek alanı ayırır.

***void\** calloc(*adet*, *boyut*)**

*boyut\*adet* byte kadar yeni bellek alanı ayırır.

***void\** realloc(*işaretçi*, *boyut*)**

*işaretçi* tarafından işaret edilen bellek alanının *boyut* byte kadar olacak şekilde değiştirir.

**free(*işaretçi*)**

*işaretçi* için ayrılan bellek alanı geri verilir.

# Örnek:

```
int *p;  
p = (int*)malloc(5 * sizeof(int));  
// 5 elemanlı bir int dizisi tanımlamaya benzer  
...  
free(p);
```

- ▶ *malloc()* ve *calloc()* fonksiyonları, o anda istenen boyutta boş bellek alanı varsa, o kadar alanı ayırır ve o alanın başlangıç adresini programa gönderir.
- ▶ Burada (int\*) yazılmasının nedeni malloc() fonksiyonunun, esnek kullanım için belirsiz yani void\* tipte işaretçi döndürmesidir.
- ▶ *free()* fonksiyonu ile de p işaretçisi için ayrılan bellek alanı geri verilir.

# malloc() ve calloc() Farkı

- ▶ *malloc()* fonksiyonu tahsis ettiği bellekteki bölgelere her hangi bir *ilk değer atama* işlemi uygulamaz.
- ▶ Yani bellekteki değerleri ile beraber size tahsis eder, ilk değer atama işlemi yazılımcıya kalmıştır.
- ▶ *calloc()* fonksiyonu bellek tahsisatı yaparken *malloc()* fonksiyonunu kullanır.
- ▶ Farklı olarak ayırdığı bellek bölgesini *sıfırlamaktadır*.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
```

```
int main(){
```

```
    int *x,i,N,toplam=0;
```

```
    float ortalama;
```

```
    printf("Eleman sayisi ");
```

```
    scanf("%d",&N);
```

```
    x = (int *) malloc(N*sizeof(int));    // N tane int gözü isteniyor (Nx2 byte)
```

```
    printf("Elemanlari girin:\n");
```

```
    for(i=0;i<N;i++){
```

```
        printf("%d.eleman = ",i+1);
```

```
        scanf("%d",&x[i]);    // x[i] ile *(x+i) aynı anlamda !
```

```
        toplam += x[i];
```

```
    }
```

```
    free(x);    // ayrılan yer boşaltılıyor...
```

```
    ortalama = (float) toplam/N;
```

```
    printf("Bu %d sayinin ortlamasi %f dir\n",N,ortalama);
```

```
    getch();
```

```
    return 0;
```

```
}
```

```
Eleman sayisi 3
Elemanlari girin:
1.eleman = 25
2.eleman = 68
3.eleman = 92
Bu 3 sayinin ortlamasi 61.666668 dir
```

```

#include <stdio.h>
#include <stdlib.h>
#define BOYUT 10
int main()
{
    int *a=(int *)malloc(BOYUT*sizeof(int));           //Bellekten BOYUT kadar yer ayrıldı.
    int i;
    for(i=0;i<BOYUT;i++) a[i]=(i+1);
    printf("Dizinin orjinali:\n");
    for(i=0;i<BOYUT;i++) printf("%d ",a[i]);
    a=(int *)realloc(a,(BOYUT*2)*sizeof(int));         //Dizi boyutu iki katına çıkarılıyor.
    for(i=BOYUT;i<BOYUT*2;i++) a[i]=(i+1);
    printf("\nDizinin boyutu iki katina cikarildigi hali:\n");
    for(i=0;i<BOYUT*2;i++) printf("%d ",a[i]);
    return 0;
}

```

```

Dizinin orjinali:
1 2 3 4 5 6 7 8 9 10
Dizinin boyutu iki katina cikarildigi hali:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

```

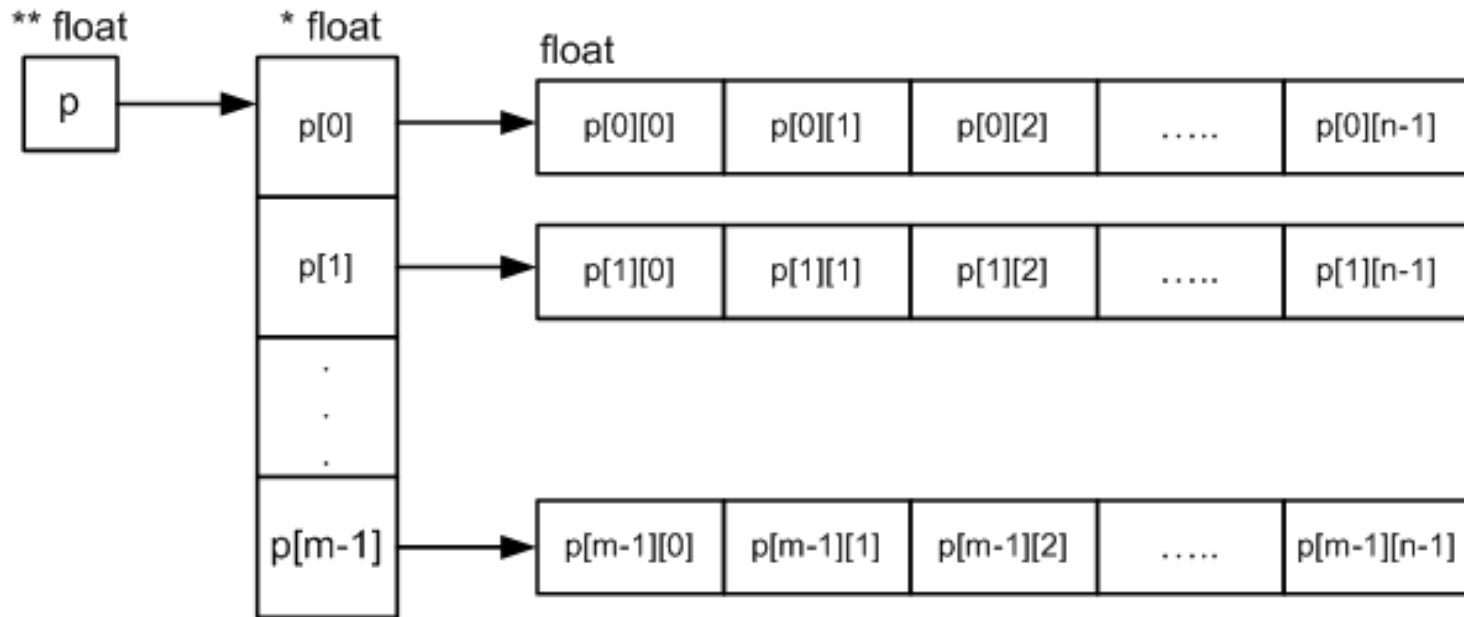
## 2 Boyutlu Dizide Dinamik Tahsisi

```
int **A;  
int A_rows = 3;  
int A_cols = 2;  
A = (int **)malloc(A_rows * sizeof(int *));  
If(A == NULL){  
    printf("bellekte yer ayrılamadı!\n");  
    exit(-1);  
}  
for(i=0; i< A_rows; i++){  
    A[i] = (int *)malloc(A_cols * sizeof(int));  
}
```

1. Boyut için bellekte gerekli yeri ayır!!

1. boyuttaki her bir bellek bölgesi için bellekte gerekli yeri ayır!!

## 2 Boyutlu Dizide Dinamik Tahsisi





# Örnek: 2 Boyutlu Dinamik Hafıza Tahsisi İle Matris Çarpımı

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

```
int main()
```

```
{
    int m, n, p, q, c, d, k, sum = 0;
    int **first, **second, **multiply;

    printf("İlk matsin boyutlarini giriniz:\n");
    scanf("%d%d", &m, &n);

    printf("İlk matrisin elemanlarini giriniz:\n");
    first = (int**)malloc(m * sizeof(int *));

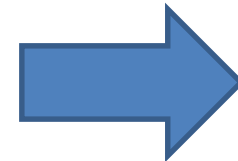
    for (c = 0; c < m; c++)
    {
        first[c] = (int*)malloc(n* sizeof(int));
        for (d = 0; d < n; d++)
            scanf("%d", &first[c][d]);
    }
}
```

```
printf("İkinci matsin boyutlarini giriniz\n");
scanf("%d%d", &p, &q);
```

```
if (n != p)
    printf("Girilen matrisler çarpilamaz.\n");
else
{
    printf("İkinci matrisin elemanlarinigiriniz\n");

    second=(int**)malloc(p*sizeof(int*));

    for (c = 0; c < p; c++)
    {
        second[c]=(int*)malloc(q*sizeof(int));
        for (d = 0; d < q; d++)
            scanf("%d", &second[c][d]);
    }
}
```



# Örnek: 2 Boyutlu Dinamik Hafıza Tahsisi İle Matris Çarpımı

```
|lk matrsin boyutlarini giriniz:
2 2
|lk matrisin elemanlarini giriniz:
1 2 4 5
|kinci matrsin boyutlarini giriniz
2 3
|kinci matrisin elemanlarini giriniz
1 2 3 4 5 6
Carpim:-
9      12      15
24     33     42
```

```
multiply=(int**)malloc(m*sizeof(int*));
for (c = 0; c < m; c++) {
    multiply[c]=(int*)malloc(q*sizeof(int*));

    for (d = 0; d < q; d++) {
        for (k = 0; k < p; k++) {
            sum = sum + first[c][k]*second[k][d];
        }
        multiply[c][d] = sum;
        sum = 0;
    }
}

printf("Carpim:-\n");
for (c = 0; c < m; c++) {
    for (d = 0; d < q; d++)
        printf("%d\t", multiply[c][d]);
    printf("\n");
}

return 0;
}
```

# Pointerlar ve Fonksiyonlar

- Fonksiyonlar dizileri pointer olarak da geri döndürebilir:

```
int *foo(void)
{
    static int array[100];
    ...
    return array;
}
```

# Pointerlar ve Fonksiyonlar

- C'de fonksiyon değer ile çağrılırsa çağırılan fonksiyonlarda değişkenleri doğrudan değiştirme yolu yoktur.

```
swap(a,b);  
...  
...  
...  
void swap(int x, int y) /* YANLIŞ */  
{  
    int temp;  
    temp = x;  
    x = y;  
    y = temp;  
}
```

# Pointerlar ve Fonksiyonlar

- Peki ya pointer kullanırsak:

```
swap(&a,&b);  
.  
.  
.  
void swap(int *px, int *py)  
{  
    int temp;  
    temp = *px;  
    *px = *py;  
    *py = temp;  
}
```

# Örnek: Pointer yardımı ile sıralama

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

void swap(int *px, int *py)
{
    int temp;
    temp = *px;
    *px = *py;
    *py = temp;
}

void sort(int *pointer, int size){
    int i, j, temp;
    for(i = 0; i < size; i++){
        for(j = i + 1; j < size; j++){
            if(pointer[j] < pointer[i]){
                swap((pointer+i),(pointer+j));
            } //if
        } //for
    } //for
} //void sort
```

```
int main()
{
    int n,i;
    int *p;
    scanf("%d",&n);
    p=(int*)malloc(n*sizeof(int));
    for(i=0;i<n;i++)
        scanf("%d",(p+i));

    sort(p,n);
    for(i=0;i<n;i++)
        printf("%d\n",*(p+i));

    return 0;
}
```

# Fonksiyonları Gösteren Göstericiler

- ▶ Bir fonksiyonu gösteren gösterici
  - ▶ Fonksiyonunun adresini tutar.
  - ▶ Bir dizi isminin ilk elemanın adresi olmasına benzer.
  - ▶ Fonksiyon isimleri, fonksiyonun görevini yapan kodun hafızadaki başlangıç adresidir.
- ▶ Fonksiyonları gösteren göstericiler
  - ▶ Fonksiyonlara geçirilebilir.
  - ▶ Fonksiyonlardan geri döndürülebilir.
  - ▶ Fonksiyon gösteren diğer göstericilere atanabilir.

```

#include <stdio.h>
#include <conio.h>
void yazdir(int,int,int(*)(int,int));
int kucuk(int,int);
int buyuk(int,int);
int main()
{
    int a,b,secim;
    printf("Iki tamsayi giriniz: ");
    scanf("%d%d",&a,&b);
    printf("Buyuk olani bulmak icin 1"
    " kucuk olani bulmak icin 2 giriniz: ");
    scanf("%d",&secim);
    if(secim==1) yazdir(a,b,buyuk);
    else yazdir(a,b,kucuk);
    getch();
    return 0;
}

```

```

void yazdir(int a,int b,int(*kar)(int,int))
{
    int sonuc=0;
    sonuc=(*kar)(a,b);
    printf("Sonuc: %d",sonuc);
}
int kucuk(int a,int b)
{
    if(a<b) return a;
    else return b ;
}
int buyuk(int a,int b)
{
    if(a>b) return a;
    else return b ;
}

```

```

Iki tamsayi giriniz: 25 56
Buyuk olani bulmak icin 1 kucuk olani bulmak icin 2 giriniz: 1
Sonuc: 56

```

```

Iki tamsayi giriniz: 25 56
Buyuk olani bulmak icin 1 kucuk olani bulmak icin 2 giriniz: 2
Sonuc: 25_

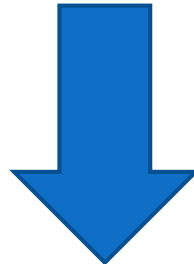
```



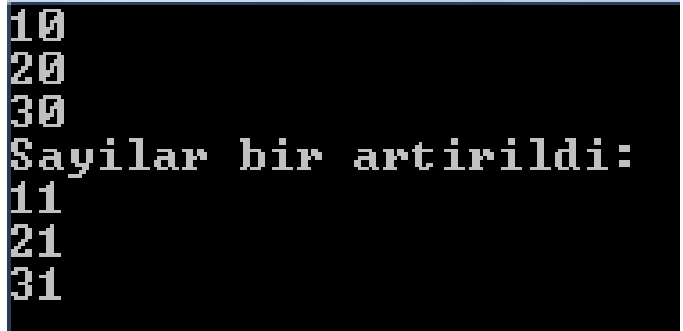
ÖRNEKLER

**Soru 1:** Bir dizideki 3 elemanın önce ilk halini yazdırıp daha sonrasında bu 3 elemanı referansa göre çağıran bir fonksiyonda birer artırıp tekrar ekrana yazdıran bir program yazınız.

```
#include <stdio.h>
#include <conio.h>
void artir (int*, int*);
void yazdir (const int*, const int*);
int main ()
{
    int sayi[3] = {10,20,30};
    yazdir (sayi,sayi+3);
    printf("Sayılar bir artırıldı:\n");
    artir (numbers,numbers+3);
    yazdir (numbers,numbers+3);
    getch();
    return 0;
}
```



```
void artir (int* start, int* stop)
{
    int * current = start;
    while (current != stop) {
        ++(*current);
        ++current;
    }
}
```



```
10
20
30
Sayilar bir artirildi:
11
21
31
```

```
void yazdir (const int *start, const int *stop)
{
    const int * current = start;
    while (current != stop) {
        printf("%d\n",*current);
        ++current;
    }
}
```

## Soru 2

- Klavyeden girilen belirli sayıyı alıp bu sayı kadar dışardan girilecek elemanı bir gösterici yardımıyla dizi gibi tutan ve ekrana yazdıran bir program yazınız.

**Not:** *malloc()* fonksiyonu kullanarak dinamik hafıza tahsisi yapılacaktır.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    int i,n;
    int * pData;
    printf ("Kac sayi gireceginiz yaziniz: ");
    scanf ("%d",&i);
    pData = (int*) malloc (i*sizeof(int));
    if (pData==NULL) return 0;
    for (n=0;n<i;n++)
    {
        printf ("%d. sayiyi giriniz: ",n+1);
        scanf ("%d",&pData[n]);
    }
    printf ("Girdikleriniz: ");
    for (n=0;n<i;n++) printf ("%d ",pData[n]);
    free (pData);
    return 0;
}
```

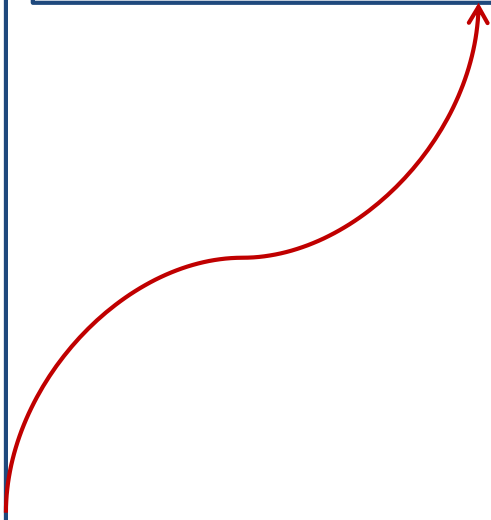
```
Kac sayi gireceginiz yaziniz: 3
#1. sayiyi giriniz: 25
#2. sayiyi giriniz: 89
#3. sayiyi giriniz: 45
Girdikleriniz: 25 89 45 _
```

## Soru 3

- ▶ Bir dizide tutulan beş sayıyı önce ekrana yazdırıp daha sonra bu sayıların karekökünü alıp tekrar ekrana yazdıran bir program yazınız.
- ▶ Programda karekökü alma işlemini bir fonksiyon gerçekleştirecek fonksiyonun çıktısı bir *gösterici* olacak.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <conio.h>
#define BOYUT 5
double *karekok(double*,int);
int main()
{
    double a[5]={ 1,2,4,8,16};
    double *b;
    int i;
    printf("A dizisi: ");
    for(i=0;i<BOYUT;i++) printf("%.4lf ",a[i]);
    printf("\n");
    b=karekok(a,BOYUT);
    printf("B dizisi: ");
    for(i=0;i<BOYUT;i++) printf("%.4lf ",b[i]);
    printf("\n");
    getch();
    return 0;
}
```

```
double *karekok(double *d,int boyut)
{
    int i;
    double *c;
    c=(double *)calloc(boyut,sizeof(double));
    for(i=0;i<boyut;i++) *(c+i)=sqrt(*(d+i));
    return c;
}
```



```
A dizisi: 1.0000 2.0000 4.0000 8.0000 16.0000
B dizisi: 1.0000 1.4142 2.0000 2.8284 4.0000
```

## Soru 4

- ▶ Fonksiyonları gösteren göstericilerin en yaygın kullanımı, menüler sunan sistemlerdir. Kullanıcının menüden (1-3 arasında) bir seçim yapması istenir. Her seçim farklı bir fonksiyon sayesinde gerçekleştirilir.
- ▶ Her fonksiyonu gösteren göstericiler, fonksiyonları gösteren göstericiler dizisi içinde tutulur. Kullanıcının seçimi dizinin belirteci olarak kullanılır ve dizideki gösterici fonksiyonu çağırır.
- ▶ Buna benzer bir program yazın. Çağrılacak fonksiyonlar birer tamsayı argüman alsın ve ekrana çağırılan fonksiyonun kaçınıcı olduğunu yazsın.



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

```
#include <conio.h>
```

```
void function1( int );
```

```
void function2( int );
```

```
void function3( int );
```

```
int main()
```

```
{
```

```
void (*f[ 3 ])( int ) = { function1, function2, function3 };
```

```
int choice;
```

```
printf( "1 ile 3 arasinda bir sayi giriniz,"
```

```
" cikmak icin bunlar disinda herhangi bir sayi giriniz: " );
```

```
scanf( "%d", &choice );
```

```
while ( choice >= 1 && choice < 4 ) {
```

```
    (*f[ choice-1 ])( choice );
```

```
    printf( "1 ile 3 arasinda bir sayi giriniz,"
```

```
" cikmak icin bunlar disinda herhangi bir sayi giriniz: " );
```

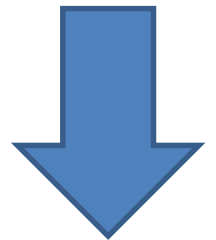
```
    scanf( "%d", &choice );
```

```
}
```

```
printf( "Program tamamlandi.\n" );
```

```
return 0;
```

```
}
```



```
void function1( int a )
{
    printf( "Girdiginiz deger: %d"
           " oyleyse function1 cagrildi\n\n", a );
}
void function2( int b )
{
    printf( "Girdiginiz deger: %d"
           " oyleyse function2 cagrildi\n\n", b );
}
void function3( int c )
{
    printf( "Girdiginiz deger: %d"
           " oyleyse function3 cagrildi\n\n", c );
}
```

```
1 ile 3 arasinda bir sayi giriniz, cikmak icin bunlar disinda herhangi bir sayi
giriniz: 1
Girdiginiz deger: 1 oyleyse function1 cagrildi

1 ile 3 arasinda bir sayi giriniz, cikmak icin bunlar disinda herhangi bir sayi
giriniz: 2
Girdiginiz deger: 2 oyleyse function2 cagrildi

1 ile 3 arasinda bir sayi giriniz, cikmak icin bunlar disinda herhangi bir sayi
giriniz: 3
Girdiginiz deger: 3 oyleyse function3 cagrildi

1 ile 3 arasinda bir sayi giriniz, cikmak icin bunlar disinda herhangi bir sayi
giriniz: 4
Program tamamlandi.
```

## Soru 5

-1 girilinceye kadar dışardan eleman almaya devam eden bir dizinin başlangıç boyutu dışardan girilmiş olsun. Dizi boyutu doldukça boyutunu, var olan boyutunun karesini alarak genişleten bir fonksiyona sahip C programını yazınız.

```

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
int artir(int *p,int *n)
{
    int m;
    if(*n==1) m=(*n)*2;
    else m=(int)pow(*n,2);
    p=(int *)realloc(p,m*sizeof(int));
    if(p!=NULL) {
        printf("\nYeni Bellek"
            " Boyutu:%d\n\n",m);
        *n=m;
        return 1;
    }
    return 0;
}

```

```

int main()
{
    int n,s,b,j,i=0;
    printf("Boyut giriniz:");
    scanf("%d",&n);
    int *a=(int *)malloc(n*sizeof(int));
    do{
        printf("%d. elemani giriniz:",(i+1));
        scanf("%d",&s);
        if(s==-1) break;
        *(a+i)=s;
        i++;
        if(i>n){
            b=artir(a,&n);
            if(b==0){
                printf("Bellek ayrilamadi!");
                break;
            }
        }
    }while(1);
    printf("\n\nGirdiginiz dizi:\n");
    for(j=0;j<i;j++) printf("%d\n",a[j]);
    free(a);
    getch();
    return 0;
}

```

SON