

# ALGORİTMA VE PROGRAMLAMA I

Yrd. Doç. Dr. Deniz KILINÇ  
[deniz.kilinc@cbu.edu.tr](mailto:deniz.kilinc@cbu.edu.tr)

YZM 1101

Celal Bayar Üniversitesi Hasan Ferdi Turgutlu  
Teknoloji Fakültesi

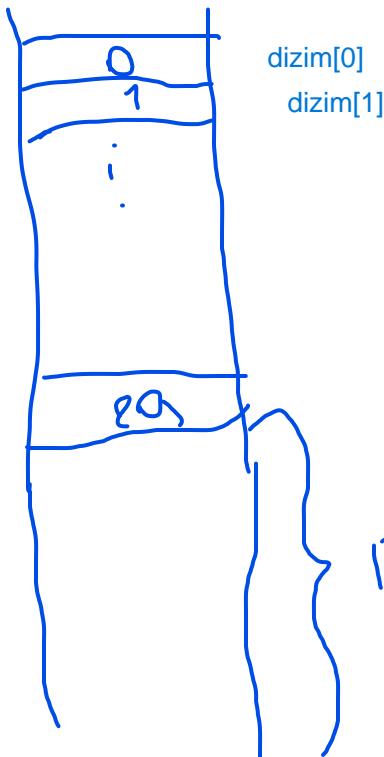
# Genel Bakış...

2

- Diziler
  - Dizi Nedir?
  - Dizilerin Bildirimİ
  - Dizilere Başlangıç Değeri Verme
- Dizilerde Arama Algoritmaları
  - Dizilerde Arama
  - Doğrusal Arama (Linear Search)
  - İkili Arama (Binary Search)
- Çok Boyutlu Diziler

# 6. BÖLÜM

3



çözüm 1  
const int Boyut

çözüm2  
sizeof(dizinin kapasitesini verir)

## Diziler

dizinin kapasitesi vs dizinin eleman sayısı  
30 olabilir ama içine 2 eleman koyarsın 2 elemanlı olur

# Dizi Nedir?

4

- Veri yapısı türlerinden bir tanesidir (Array, Struct, Pointer, Class ...).
- İçerisinde birden fazla,
  - aynı tip ve
  - aynı isimdeki veriyi **bellekte** depolayabilen değişkenlere **dizi** denir.
- **Örneğin;** bir sınıfındaki öğrencilerin notlarını saklamak için dizileri kullanmak mümkündür.

# Dizilerin Bildirimi

5

- Bir dizi çok sayıda değişken barındırdığından, bunları birbirinden ayırt etmek için **indis** adı verilen bilgiler kullanılır.
- C Programlama Dili'nde, bir dizi hangi tipte tanımlanmış olursa olsun **başlangıç indis** her zaman **0**'dır.
- Diziler tanımlanırken;
  - dizinin **adı**,
  - dizinin **boyutu**,
  - dizi elemanlarının hangi **tipte** olacağı belirtilmelidir.

# Dizilerin Bildirimi (devam...)

6

- Bir dizinin bildirim işleminin genel biçimi aşağıdaki gibidir:

*veriTipi dizi\_adı [eleman\_sayısı];*

- Örneğin; **double** türündeki 8 adet öğrenci notunu bellekte tutmak için aşağıdaki gibi bir dizi tanımlayabiliriz:

*double ogrenci\_notu[8];*

# Dizilerin Bildirimi (devam...)

7

- Şu andaki bilgilerimizle bunu nasıl yapabiliriz?

```
double ogrenci_notu1;  
double ogrenci_notu2;  
double ogrenci_notu3;  
double ogrenci_notu4;  
double ogrenci_notu5;  
double ogrenci_notu6;  
double ogrenci_notu7;  
double ogrenci_notu8;
```

# Örnek: Dizilerin Bildirimi (devam...)

8

<b>ogrenci_notu[0]</b>	45	1. eleman
<b>ogrenci_notu[1]</b>	56	2. eleman
<b>ogrenci_notu[2]</b>	78	3. eleman
<b>ogrenci_notu[3]</b>	93	4. eleman
<b>ogrenci_notu[4]</b>	78	5. eleman
<b>ogrenci_notu[5]</b>	69	6. eleman
<b>ogrenci_notu[6]</b>	77	7. eleman
<b>ogrenci_notu[7]</b>	90	8. eleman

# Örnek: Dizilerin Bildirimi (devam...)

9

- Dizinin ismi **ogrenci\_notu** dur.
- Dizinin 8 elemanı
  - $ogrenci\_notu[0]$ ,  $ogrenci\_notu[1]$ , ..... ,  $ogrenci\_notu[7]$  şeklinde gösterilmiştir.
- $ogrenci\_notu[0]$  içinde tutulan değer 45,  $ogrenci\_notu[1]$  içinde tutulan değer 56,  $ogrenci\_notu[7]$  içinde tutulan değer 90 dır.
- Bu dizinin ilk iki elemanının içinde tutulan değerlerin toplamını yazdırma isteseydik;
  - `printf("Sonuc: %f", ogrenci_notu[0] + ogrenci_notu[1]);`
  - Sonuc: 101

# Örnek: Dizilerin Bildirimi (devam...)

10

- Bu dizinin yedinci elemanın değerini ikiye bölüp, oluşan sonucu x değişkenine atasaydık;
  - $x = \text{ogrenci\_notu}[6] / 2;$



Dizinin yedinci elemanı ile yedinci dizi elemanı arasındaki farkı önemlidir. Dizi indisleri 0'dan başladığını için "dizinin yedinci elemanı" 6 indisine sahiptir. "yedinci dizi elemanı" ise 7 indisine sahiptir ve aslında dizinin sekizinci elemanıdır.

# Dizilere Başlangıç Değeri Verme

11

- Bir dizi, doğal olarak bazı veriler içerecektir.
- Diziye aynı anda birden fazla değer atanabilir. Bunun için söz konusu değerler **{ }** işaretleri arasında **virgül** ile ayrılarak yazılırlar.
- **Örnek:**
  - ✓ **float** kutle[5] = { 8.471, 3.683, 9.107, 4.739, 3.918 };
  - ✓ **int** maliyet[3] = { 25, 72, 94 };
  - ✓ **double** a[4] = { 10.0, 5.2, 7.5, 0.0};
- Küme parantezleri sonlandırıcı **;** karakteri ile bitmektedir.

# Dizilere Başlangıç Değeri Verme (devam...)

12

- Bir dizinin uzunluğu belirtilmeden de başlangıç değeri atamak mümkündür.
- **Örnek:**
  - `int a[] = { 100, 200, 300, 400 };`
  - `float v[] = { 9.8, 11.0, 7.5, 0.0, 12.5 };`
- Derleyici bu şekilde bir atama ile karşılaşlığında, küme parantezi içindeki eleman sayısını hesaplar ve dizinin o uzunlukta açıldığını varsayar.
- Yukarıdaki örnekte, **a** dizisinin **4**, **v** dizisinin **5** elemanlı olduğu varsayıılır.

# Dizilere Başlangıç Değeri Verme (devam...)

13

- Dizilere başlangıç değeri atarken, tüm elemanlara değer vermeden de atama yapmak mümkündür.
- **Örnek:**
  - `int sayilar[20] = { 0 };`
    - Tüm dizi elemanlarına 0 değeri atanır...
  - `int sayilar[20] = { 1, 2, 3 };`
    - Dizinin ilk 3 elemanına 1, 2 ve 3 değerleri atanır. 4'ten itibaren olan dizi elemanlarına 0 değeri atanır...
- Sayısal tipteki dizi elemanlarına 0 değeri, metin tipindeki dizi elemanlarına NULL değeri atanır.

# Örnek-1:

## Dizi elemanlarına değer atama ve okuma

14

- Tek boyutlu, **5 elemanlı**, sayısal (int) bir dizi tanımlanarak:
  - Dizi elemanlarına dizi indislerinin 3 katı **for** döngüsü içerisinde değer olarak atanacaktır.
  - Yine başka bir **for** döngüsü içerisinde bu değerler ekrana yazdırılacaktır.

# Örnek-1:

## Dizi elemanlarına değer atama ve okuma (devam...)

15

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

int main()
{
    int ornek_dizi[5]; //DIZI TANIMLAMA
    int i;

    for (i=0; i<5; i++) {
        ornek_dizi[i] = i * 3; //DEGER ATAMA
    }

    for (i=0; i<5; i++) {
        printf("%d. dizi eleman degeri: %d\n", i, ornek_dizi[i]); //DEGER OKUMA
    }

    return 0;
}
```

```
0. dizi eleman degeri: 0
1. dizi eleman degeri: 3
2. dizi eleman degeri: 6
3. dizi eleman degeri: 9
4. dizi eleman degeri: 12
```

## Örnek-2: Dizi atamaları

NEGATIF İNDİKS İLE SKULLANILMAZ  
pozitif tam sayılar kullanılır

16

- ```
int x[5];           // 5 elemanlı sayısal x dizisi
int i = 2;

• x[0] = 20;        // Geçerli atama
• x[2.3] = 5;       // Geçersiz atama
• x[2*i - 3] = 3;   // Geçerli atama, x[1] dizi elemanına 3
                     değerini atar
• x[i++];          // Önce x[2] dizi elemanına erişilir daha sonra
                     i değişkenine 3 değeri atanır
• x[(int) x[1]];    // x[3] dizi elemanına erişilir
```

## Örnek-3:

### Klavyeden sayısal değer girme

17

- Klavyeden maksimum 10 tane sayısal değer girilecektir.
- Girilen sayılar bir dizide saklanacaktır.
- Sayı girme işlemi 0 girilene kadar devam edecektir.
- 0 değeri girildiği anda 0 sayısı hariç girilen diğer tüm değerler diziden okunarak ekrana yazdırılacaktır.

## Örnek-3:

# Klavyeden sayısal değer girme (devam..)

18

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

int main()
{
    int dizi[10];
    int i, j;

    for (i=0; i<10; i++) {
        printf("%d. sayıyı giriniz:", i);
        scanf("%d", &dizi[i]);
        if (dizi[i] == 0)
            break;
        j = i;
    }

    for (i=0; i<=j; i++) {
        printf("%d \n", dizi[i]);
    }

    return 0;
}
```

# Dizi Kullanımlarında Dikkat Edilmesi Gereken En Önemli Nokta!

19

- Dizi boyunca döngü kullanırken dizi indisleri asla **0'ın altına inmemeli** ve her zaman **dizideki toplam eleman sayısından az olmalıdır** (**büyüklük-1**).
- Döngü devam şartının bu aralığın dışındaki elemanlara ulaşılmasını engellediğinden emin olmamız gereklidir.
- Dizi sınırlarının dışındaki elemanları kullanmanın yaratacağı hatalar (genelde ciddi hatalardır) sistemden sisteme farklılık gösterir.



# Örnek-4: Dizi elemanı değeri kadar ekrana \* karakteri yazdırma (Uygulama Dersi)

20

- 10 elemanlı { 19, 3, 15, 7, 11, 9, 13, 5, 17, 1 } grafik isimli bir dizi oluşturulacaktır.
- Dizideki elemanlar tek tek okunarak her dizi elemanın sayısı kadar ekrana \* karakteri yazdırılacaktır.
- Ekran çıktısı aşağıdaki gibi olacaktır.

```
0. eleman degeri: 19 -->*****  
1. eleman degeri: 3 -->***  
2. eleman degeri: 15 -->*****  
3. eleman degeri: 7 -->*****  
4. eleman degeri: 11 -->*****  
5. eleman degeri: 9 -->*****  
6. eleman degeri: 13 -->*****  
7. eleman degeri: 5 -->*****  
8. eleman degeri: 17 -->*****  
9. eleman degeri: 1 -->*
```

# Örnek-4: Dizi elemanı değeri kadar ekrana \* karakteri yazdırma (Uygulama Dersi)

21

```
int main()
{
    const BOYUT = 10;
    int grafik[10] = { 19, 3, 15, 7, 11, 9, 13, 5, 17, 1 };
    int i, j;

    for (i=0; i<=BOYUT-1; i++) {
        printf("%4d. eleman degeri:%5d -->", i, grafik[i]);

        for (j=1; j<=grafik[i]; j++)
        {
            printf("*");
        }
        printf("\n");
    }
    return 0;
}
```

# Dizilerde Arama ve Sıralama Algoritmaları

22

- Arama ve sıralama algoritmaları (search and sort algorithms) programlama ve yazılım geliştirme dünyasında hem akademik hem de endüstri açısından önemli bir yere sahiptirler.
- Özellikle büyük veri kaynakları ile çalışırken, aradığınız veriye en hızlı şekilde ulaşmanız doğru algoritmayı kullanmanıza bağlıdır.



# Dizilerde Arama

23

- Bir dizinin, belli bir arama değerine eşit olan bir değer içerip içermediğine karar vermek gerekebilir.
- Dizinin belirli bir elemanını bulma sürecine **arama** denir.
- Ders kapsamında iki tane arama tekniği üzerinde durulacaktır:
  - Doğrusal Arama (Linear / Sequential Search)
  - İkili Arama (Binary Search)

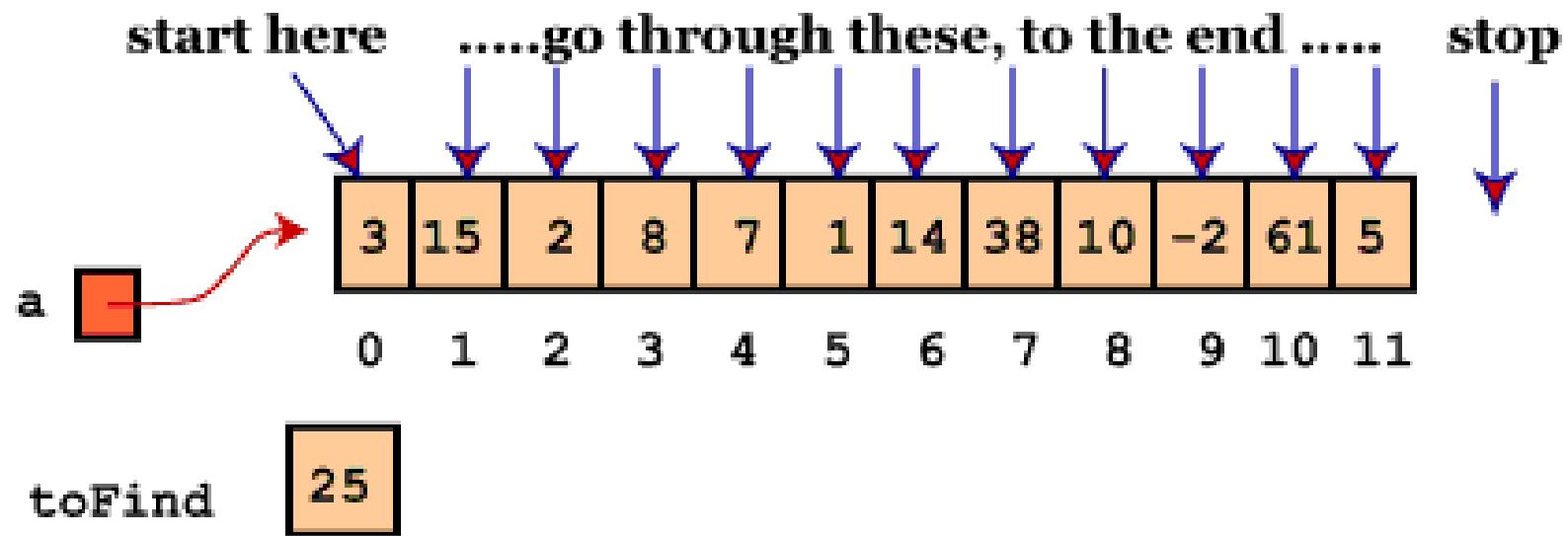
# *Doğrusal Arama (Linear Search)*

24

- Dizinin ilk elemanından başlanarak dizinin her elemanı, arama değeriyle karşılaştırılır.
- Aranan değer bulunduğunda ilgili dizi elemanın **indisi** değer olarak döndürülür.
- Aranan değer bulunamazsa -1 değeri döndürülür,
- Dizi herhangi bir şekilde sıralanmadığından değer ilk ya da son elemanda bulunabilir.
- Dolayısıyla, program ortalama olarak, arama değeriyle dizinin elemanlarının yarısını karşılaştırmalıdır.

# *Doğrusal Arama (Linear Search)*

25



## Örnek-5: *Doğrusal Arama* (devam...)

26

- 100 elemanlı tam sayı tipinde veri tutan bir dizi tanımlayınız.
- Dizinin tüm elemanlarına değer olarak dizi indislerinin 2 katı değer atayınız.
- Daha sonra aranacak veriyi klavyeden giriniz.
- Eğer aranan veri dizide bulunursa, bulunduğu dizinin indis ile birlikte ekrana yazdırınız.
- Eğer aranan bilgi bulunamazsa ekrana bulunamadığını dair bilgi veriniz.

```
Aranacak bilgiyi giriniz:76
Aranan bilgi 76 --> 38. indiste bulundu..
Process returned 0 (0x0)  execution time : 5.414 s
Press any key to continue.
```

## Örnek-5: *Doğrusal Arama* (devam...)

27

```
int main()
{
    int aranacak_dizi[100];
    int aranan_bilgi, i;
    int bulunan_indis = -1;

    //Dizi elemanları otomatik olarak çift sayılarla dolduruluyor.
    for (i=0; i<100; i++) {
        aranacak_dizi[i] = i * 2;
    }
    //Aranacak bilgi okunuyor
    printf("Aranacak bilgisi giriniz:");
    scanf("%d", &aranan_bilgi);
    //Doğrusal arama
    for (i=0; i<100; i++) {
        if (aranan_bilgi == aranacak_dizi[i]) {
            bulunan_indis = i;
            printf("Aranan bilgi %d --> %d. indiste bulundu..\n", aranan_bilgi, bulunan_indis);
            break;
        }
    }
    if (bulunan_indis == -1)
        printf("Aranan bilgi %d bulunamadi\n", aranan_bilgi);
    return 0;
}
```

```
Aranacak bilgisi giriniz:76
Aranan bilgi 76 --> 38. indiste bulundu..
Process returned 0 (0x0)   execution time : 5.414 s
Press any key to continue.
```

```
bas orta son
if (key <dizi[orta] dizinin orta elemanı )
    dizi ortanın soluna bak
if(key>dizi orta
    dizi ortanın sağına bak
else
    aradı mı buldu?
```

# İkili Arama (Binary Search)

- İkili arama, sıralı bir dizide, belirli değerin bulunmasına yönelik bir algoritmadır.
- Bu teknikteki her bir adımda, aranan değerin, dizinin orta değerine eşit olup olmadığı kontrol edilir.
  - Eşit olmaması durumunda aranan değerin orta değer tarafından ikiye ayrılan kısımlardan hangisinde olduğu kontrol edilir.
  - Aranan değeri içeren kısım bir sonraki adımda arama yapılacak dizi olur ve bu sayede arama yapılan listedeki eleman sayısı her adımda yarıya indirilmiş olur.

bu kod çalışması için sıralı olmalı

# İkili Arama (devam...)

```
#include <stdio.h>

int main()
{
    int c, baslangic, son, orta, n, aranan_deger, dizi[100];

    printf("\n Dizinin eleman sayisini giriniz: ");
    scanf("%d", &n);

    for (c=0;c<n;c++)
        printf("\n Dizinin %d. elemanini giriniz: ", c);
        scanf("%d", &dizi[c]);
    }

    printf("\n Aranacak degeri giriniz: ");
    scanf("%d", &aranan_deger);

    baslangic = 0;
    son = n - 1;
    orta = (baslangic+son)/2;

    while(baslangic<=son)
    {
        if (dizi[orta]<aranan_deger)
            baslangic = orta + 1;
        else if (dizi[orta] == aranan_deger)
        {
            printf("\n Aranan < %d > degeri dizinin %d. elemani olarak bulundu.\n", aranan_deger, orta);
            break;
        }
        else
            son = orta - 1;
        orta = (baslangic + son)/2;
    }
    if (baslangic>son)
        printf("\n Aranan %d degeri bulunamadi.\n", aranan_deger);

    return 0;
}
```

kaç eleman doldurmak istiyorsan

```
Dizinin eleman sayisini giriniz: 5
Dizinin 0. elemanini giriniz: 2
Dizinin 1. elemanini giriniz: 3
Dizinin 2. elemanini giriniz: 6
Dizinin 3. elemanini giriniz: 8
Dizinin 4. elemanini giriniz: 11
Aranacak degeri giriniz: 8
Aranan < 8 > degeri dizinin 3. elemani olarak bulundu.
```

# Örnek-6: 3 Öğrenci Not Girişi ve Hesaplama

30

- 3 öğrencinin 3 adet sınav notu bilgisi klavyeden girilecektir.
- Her öğrenci için tek boyutlu ayrı bir dizi tanımlamalısınız ve not girişlerini **for döngüsünde yapmalısınız**.
  - ogr1Not, ogr2Not, ogr3Not
- Notların girişi tamamlandıktan sonra:
  - **Ekran Çıktısı 1:** 1. öğrencinin 1.sınav notu ile 2. öğrencinin 2.sınav notu ve 3. öğrencinin 3.sınav notunu toplayarak ekranda gösteriniz.
  - **Ekran Çıktısı 2:** Her sınav için ağırlıklı not ortalamasını bulunuz ve ekrana yazdırınız. **for döngüsü kullanınız**.

```
1. ogrencinin 1. notunu giriniz:90
2. ogrencinin 1. notunu giriniz:80
3. ogrencinin 1. notunu giriniz:100
1. ogrencinin 2. notunu giriniz:45
2. ogrencinin 2. notunu giriniz:65
3. ogrencinin 2. notunu giriniz:80
1. ogrencinin 3. notunu giriniz:97
2. ogrencinin 3. notunu giriniz:65
3. ogrencinin 3. notunu giriniz:34
Ekran ciktisi 1:189
```

YZM 1101

```
1. sinav ortalamasi:90
2. sinav ortalamasi:63
3. sinav ortalamasi:65
```

# Örnek-6: 3 Öğrenci Not Girişi ve Hesaplama

(devam...)

31

```
int main()
{
    int ogr1Not[3], ogr2Not[3], ogr3Not[3];
    int i, j;

    for (i = 0; i < 3; i++)
    {
        printf("1. öğrencinin %d. notunu giriniz:", i+1);
        scanf("%d", &ogr1Not[i]);
        printf("2. öğrencinin %d. notunu giriniz:", i+1);
        scanf("%d", &ogr2Not[i]);
        printf("3. öğrencinin %d. notunu giriniz:", i+1);
        scanf("%d", &ogr3Not[i]);
    }

    int ciktil = ogr1Not[0] + ogr2Not[1] + ogr3Not[2];
    printf("Ekran ciktisi 1:%d\n\n", ciktil);

    for (i = 0; i < 3; i++)
    {
        printf("%d. sınav ortalaması:%d\n", i + 1, (ogr1Not[i] + ogr2Not[i] + ogr3Not[i])/3);
    }
    return 0;
}
```

# Çok Boyutlu Dizi

32

- Bir dizi aşağıdaki gibi bildirildiğinde bir boyutlu (tek indisli) dizi olarak adlandırılır. Bu tip dizilere **vektör** denir.

```
int x[5];
```

- Bir dizi birden çok boyuta sahip olabilir. Örneğin iki boyutlu y dizisi şöyle tanımlanabilir:

```
int y [5] [10];
```

- İki boyutlu diziler **matris** olarak adlandırılır.
- İlk boyuta **satır**, ikinci boyuta **sütün** denir. y matrisinin eleman sayısı  $5 \times 10 = 50$  dir.

# Çok Boyutlu Dizi (devam...)

33

- Çok boyutlu dizi örnekleri:

| Dizi Çeşiti                               | Genel Bildirimi                                   | Örnek                         |
|-------------------------------------------|---------------------------------------------------|-------------------------------|
| Tek boyutlu diziler<br><u>(Vektörler)</u> | <b>tip</b> dizi_adı[eleman_sayısı]                | <b>int</b> veri [10] ;        |
| İki boyutlu diziler<br><u>(Matrisler)</u> | <b>tip</b> dizi_adı[satır_sayısı][sutun_sayısı]   | <b>float</b> mat [5] [4] ;    |
| Cök boyutlu diziler                       | <b>tip</b> dizi_adı[boyut_1][boyut_2]...[boyut_n] | <b>double</b> x [2] [4] [2] ; |

# Çok Boyutlu Dizilerin Bildirimi

34

- Çok boyutlu diziler tek boyuta indirgenerek bellekte tutulurlar.
- Tek indisli dizilerde olduğu gibi, çok indisli dizilere de başlangıç değeri vermek mümkündür.
- Örneğin 3 satır ve 4 sütunlu ( $3 \times 4 = 12$  elemanlı) bir x matrisinin elemanları şöyle tanımlanabilir:

```
int x[3][4] = { 11,34,42,60, 72,99,10,50, 80,66,21,38};
```

```
int x[3][4] = { 11,34,42,60, /* 1. satır elemanları */  
                72,99,10,50, /* 2. satır elemanları */  
                80,66,21,38}; /* 3. satır elemanları */
```

# Örnek1: Çok Boyutlu Dizi Bildirimi ve Dizi Elemanlarını Yazdırma

35

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

int main()
{
    int i, j;
    int x[3][4] = {11,34,42,60, /* 1. satır elemanları */
                   72,99,10,50, /* 2. satır elemanları */
                   80,66,21,38}; /* 3. satır elemanları */
    for(i=0; i<3; i++)
    {
        for(j=0; j<4; j++)
            printf("%4d",x[i][j]);

        printf("\n");
    }
    return 0;
}
```

|    |    |    |    |
|----|----|----|----|
| 11 | 34 | 42 | 60 |
| 72 | 99 | 10 | 50 |
| 80 | 66 | 21 | 38 |
| -  | -  | -  | -  |

# Örnek2: Satranç Tahtası

36

- Bir satranç oyunundaki 64 bölgeyi nasıl tanımlarız?

`int Satranc [8][8];`

|              |     |     |     |     |     |     |     |     |
|--------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Satranc      | 1,1 | 1,2 | 1,3 | 1,4 | 1,5 | 1,6 | 1,7 | 1,8 |
|              | 2,1 | 2,2 | 2,3 | 2,4 | 2,5 | 2,6 | 2,7 | 2,8 |
|              | 3,1 | 3,2 | 3,3 | 3,4 | 3,5 | 3,6 | 3,7 | 3,8 |
|              | 4,1 | 4,2 | 4,3 | 4,4 | 4,5 | 4,6 | 4,7 | 4,8 |
|              | 5,1 | 5,2 | 5,3 | 5,4 | 5,5 | 5,6 | 5,7 | 5,8 |
| Satranc(6,3) | 6,1 | 6,2 | 6,3 | 6,4 | 6,5 | 6,6 | 6,7 | 6,8 |
|              | 7,1 | 7,2 | 7,3 | 7,4 | 7,5 | 7,6 | 7,7 | 7,8 |
|              | 8,1 | 8,2 | 8,3 | 8,4 | 8,5 | 8,6 | 8,7 | 8,8 |

# İki Matrisin Toplamı

37

- İki matrisin karşılıklı elemanları toplanarak, toplam matris elde edilir.
- Bunu matris ifadeleriyle gösterelim:

$$[t_{i,j}] = [a_{ij}] + [b_{ij}]$$

# İki Matrisin ToplAMI (devam...)

38

```
int main()
{
    int a[2][3] = {5, 3, 7, 0, 1, 2};
    int b[2][3] = {1, 2, 3, 4, 5, 6};
    int c[2][3];
    int i, j;

    printf("A Matrisi: \n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++)
            printf("%4d", a[i][j]);
        printf("\n");
    }

    printf("B Matrisi: \n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++)
            printf("%4d", b[i][j]);
        printf("\n");
    }

    printf("\nC Matrisi:\n");
    for(i=0; i<2; i++){
        for(j=0; j<3; j++){
            c[i][j] = a[i][j] + b[i][j];
            printf("%4d", c[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```



```
A Matrisi:
 5   3   7
 0   1   2
B Matrisi:
 1   2   3
 4   5   6
C Matrisi:
 6   5   10
 4   6   8
```

# Örnek: 3 Öğrenci Not Girişi ve Hesaplama

39

- 3 öğrencinin 3 adet sınav notu bilgisi klavyeden girilecektir.
- Öğrenciler için 2 boyutlu bir dizi tanımlamalısınız ve not girişlerini **for döngüsünde yapmalısınız**.
  - ogrNotlar [3][3]
- Notların girişi tamamlandıktan sonra:
  - **Ekran Çıktısı 1:** 1.öğrenci notu ve 3.öğrencinin 3.sınav notu ve ekrana yazdırınız. **for d**

```
1. ogrencinin 1. notunu giriniz:90
1. ogrencinin 2. notunu giriniz:80
1. ogrencinin 3. notunu giriniz:70
2. ogrencinin 1. notunu giriniz:50
2. ogrencinin 2. notunu giriniz:30
2. ogrencinin 3. notunu giriniz:50
3. ogrencinin 1. notunu giriniz:40
3. ogrencinin 2. notunu giriniz:90
3. ogrencinin 3. notunu giriniz:60
Ekran ciktisi 1:180
```

```
1. sinav ortalamasi:60
2. sinav ortalamasi:66
3. sinav ortalamasi:60
```

# Örnek: 3 Öğrenci Not Girişi ve Hesaplama

(devam...)

40

```
int main()
{
    int ogrNotlar[3][3];
    int i, j;

    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++)
        {
            printf("%d. öğrencinin %d. notunu giriniz:", i+1, j+1);
            scanf("%d", &ogrNotlar[i][j]);
        }
    }

    int ciktil = ogrNotlar[0][0] + ogrNotlar[1][1] + ogrNotlar[2][2];
    printf("Ekran ciktisi 1:%d\n\n", ciktil);

    for (i = 0; i < 3; i++)
    {
        printf("%d. sınav ortalaması:%d\n", i + 1,
               (ogrNotlar[0][i] + ogrNotlar[1][i] + ogrNotlar[2][i])/3);
    }
    return 0;
}
```

# İki Matrisin Çarpımı

41

- Göz önünde bulundurulması gereken en önemli koşul, çarpımı yapılacak birinci matrisin sütun sayısının ikinci matrisin satır sayısına eşit olması gerektidir.
- İki matrisin çarpımı aşağıdaki şekilde gösterilir:

$$[c_{ij}] = \sum_{j=1}^n a_{ij} b_{jk}$$

# İki Matrisin Çarpımı (devam...)

42

$$[a_{ij}] = \begin{bmatrix} 5 & 7 & 9 \\ 0 & 3 & 0 \\ 7 & 5 & 1 \end{bmatrix} \quad [b_{ij}] = \begin{bmatrix} 3 & 3 & 1 \\ 2 & 1 & 3 \\ 1 & 0 & 0 \end{bmatrix}$$



$$[c_{ij}] = \begin{bmatrix} 38 & 22 & 26 \\ 6 & 3 & 9 \\ 32 & 26 & 22 \end{bmatrix}$$

# İki Matrisin Çarpımı (devam...)

43

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

int main()
{
    int a[3][3] = {5,7,9, 0,3,0, 7,5,1};
    int b[3][3] = {3,3,1, 2,1,3, 1,0,0};
    int c[3][3] = {};
    int i,j,k,toplam;

    printf("C Matrisi:\n");
    for(i=0; i<3; i++){
        for(j=0; j<3; j++){

            for(toplam=0, k=0; k<3; k++)
                toplam += a[i][k]*b[k][j];

            c[i][j] = toplam;
            printf("%4d",c[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

|    |    |    |
|----|----|----|
| 38 | 22 | 26 |
| 6  | 3  | 9  |
| 32 | 26 | 22 |

# KAYNAKLAR

44

- N. Ercil Çağiltay ve ark., C DERSİ PROGRAMLAMAYA GİRİŞ, Ada Matbaacılık, ANKARA; 2009.
- Milli Eğitim Bakanlığı "Programlamaya Giriş ve Algoritmalar Ders Notları", 2007
- <http://tr.wikipedia.org/wiki/Code::Blocks>
- <http://www.codeblocks.org>
- <http://www.AlgoritmaveProgramlama.com>
- <http://www1.gantep.edu.tr/~bingul/c>



Algoritma ve Programlama