BSM101 Programlama Dilleri I

Hafta 10
Diziler

Dr. Öğr. Üyesi Caner ÖZCAN

Diziler

- Bilgisayarlar yardımıyla yapılan işlemlerde, çok sayıda veri girilmesi ve girilen verilerin işlenerek belirli bir sistematiğe göre sıralanması gerekebilir.
- Belirli bir düzende olan verilerin işlenmesi hem daha kolay hem de daha pratiktir.
- Bu nedenle bilgisayar programlarında çoklu verileri işlemek için "dizi" olarak adlandırılan sıralı veri alanları kullanılır.
- Tek isimle adlandırılan bu veri alanları belleğe ardışık olarak yerleşirler.

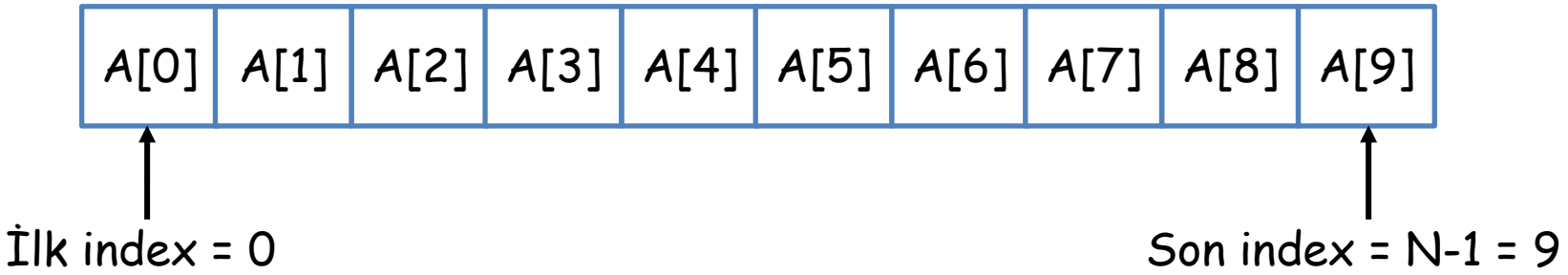
Diziler

- Aynı amaç için birden fazla aynı tip değişkene ihtiyacımız olur.
- Örneğin, 100 kişilik bir sınıfın "Programlama Dilleri" dersinden aldığı yılsonu notlarını tek tek değişkenlere aktarmak yerine (100 tane değişken adı gerekli), bunları ‘Notlar’ isimli bir dizide tutulabilir.
- Bu şekilde birçok değişken adı ve alanı kullanımına gerek kalmaz.
- Bilgiler tek bir isimle belirli bir yapı altında tutulur ve hızlı bir şekilde işlenirler.

Diziler

- Birden fazla aynı tip değişkeni bir arada tutan veri yapısıdır.
- En basit tipi bir boyutlu olanıdır.
  - Bir boyutlu dizinin elemanları bir satırda bir biri ardına dizilmiş şekilde kabul edilir.
- Kodlama:

```
#define N 10
…
int  A[N];
```



Diziler

- Dizinin n. elemanı c[n-1] ile gösterilir.
  - c[0]+ c[1]+ c[2]+…….c[n-1]
- Dizi elemanları normal değişkenler gibidir.
  - c[0] = 3;
  - printf(“%d”, c[0]);
- İndis numarası üzerinde işlemler gerçekleştirilebilir. a= 2, b=3 ise
  - c[a+b] += 8; // c[5] eleman değerine 8 ekler.
- Dizinin ilk üç elemanının değerleri toplamını yazdırmak için
  - printf(“%d”, c[0]+c[1]+c[2]);

Dizilere İlk Değer Atama

- Dizilere tanımlama sırasında ilk değer atanabilir.

```
int A[10]={8, 4, 10, 2, 5, 6, 7, 8, 9, 4};
```

- Eğer ilk değerler dizinin eleman sayısından az ise kalan elemanların değeri 0 olur.

```
int A[10]={1, 2, 3, 4};
/* A[10] dizisinin ilk değerleri {1, 2, 3, 4, 0, 0, 0, 0, 0, 0}*/
```

- Eğer ilk değerlerle bir dizi tanımlıyorsak, dizinin boyutunu boş bırakabiliriz.

```
int A[]={1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
/* A dizisinin 10 elemanı var A[0]..A[9] */
```

Dizilere İlk Değer Atama

- Dizi elemanlarının başlangıç değerleri otomatik olarak sıfır olmaz. Bunun için en azından ilk eleman değeri sıfır yapılmalıdır.

```
int n[5] = {0}; // tüm elemanların değeri 0 olur
```

- Eğer gereğinden fazla başlangıç değeri varsa hata oluşur.

```
int n[5] = {1, 2, 3, 4, 5, 6}; //altı adet başlangıç değeri
```

Dizi Kullanımı

- Dizilerin her bir elemanına ulaşmak için her elemanın indeksini kullanmamız gerekiyor.
- Bir indeks elemanın dizideki yerini ifade ediyor.
- Dizinin elemanları peş peşe sıralanmıştır. (arada boşluk yok)
- Dizinin her elamanı sırasıyla tanımlanır ve bu sıralama 0 dan başlar.

Dizi Kullanımı

- Örnek

```
#define MAX_OGR_SAYISI 5
…
int notlar[MAX_OGR_SAYISI];


…
notlar[0] = 98;
notlar[1] = 87;
notlar[2] = 92;
notlar[3] = 79;
notlar[4] = 85;
```

Dizi Kullanımı

- Uyarı!
  - C indexlerin doğru aralıkta olup olmadığını kontrol etmiyor. (yani, index değerleri, [ ] operatorü kullanılırken dizinin sınırları içerisinde mi değil mi diye kontrol edilmiyor.)

```
#define MAX_OGR_SAYISI 5
…
int notlar[MAX_OGR_SAYISI];

…

notlar[53] = 98;
notlar[5] = 98;
```

Dizi Kullanımı

- Dizinin elemanlarına ulaşılırken genelde döngüler kullanılır, ve döngünün her iterasyonunda dizinin bir elemanı üzerinde çalışılır.

- En sık kullanılan döngü for döngüsüdür. Çünkü döngü ifadesinde açıkça hem ilk değer atamaları hem de indeks değişkeni kullanılabiliyor:

```
notlar[0] = 0;
notlar[1] = 0;
notlar[2] = 0;
notlar[3] = 0;
notlar[4] = 0;
```

```
int i;
for(i = 0; i < MAX_OGR_SAYISI; i++)
        notlar[i] = 0;
```

Örnek: Okuma

```
#include <stdio.h>
#define SIZE 5

int main(void)
{
   int i;
   double a[SIZE];
   printf("%d tane dizi elemanı gir: ", SIZE);
   /* dizi elemanlarını oku*/
   for(i = 0; i < SIZE; i++)
        scanf("%lf", &a[i]);
   return 0;
}
```

```
5 tane dizi elemanı gir: 1.2 3.4 5.6 7.8 9.0
```

Örnek: Yazma

```
#include <stdio.h>
#define SIZE 5

int main(void)
{
   int i;
   double a[SIZE] = { 1.2, 3.4, 5.6, 7.8,
                      9.0 };
   /* Dizi elemanlarını göster*/
   for(i = 0; i < SIZE; i++)
       printf("a[%d] = %.2lf\n", i, a[i]);

   return 0;
}
```

```
a[0] = 1.20
a[1] = 3.40
a[2] = 5.60
a[3] = 7.80
a[4] = 9.00
```

Örnek: En Büyük Eleman

```
#include<stdio.h>

#define SIZE 5

int main(void)
{
    int i;
    double a[SIZE] = { 1.2, 3.4,
                       5.6, 7.8,
                       9.0 };
    double max = 0.0;
    /* Dizideki en büyük elemanı bul*/
    for(i = 0; i < SIZE; i++)
         if (a[i] > max)
                max = a[i];
    printf("max = %.2lf\n", max);
    return 0;
}
```

```
max = 9.00
```

Örnek: Dizi Toplamı

```
#include <stdio.h>

int main(void) {
   int i, N, A[100], B[100], C[100];
   printf("Dizinin eleman sayisini gir:\n");
   scanf("%d", &N);
   for(i = 0; i < N; i++){  /* Dizi elemanlarını oku*/
        printf("A[%d]=",i);
        scanf("%d",&A[i]);
   }
   for(i = 0; i < N; i++){  /* Dizi elemanlarını oku*/
        printf("B[%d]=",i);
        scanf("%d",&B[i]);
   }
   for(i = 0; i < N; i++){  /* Dizi elemanlarının toplamı*/
        C[i] = A[i] + B[i];
        printf("C[%d]=%d\n", i, C[i]);
   }
   return 0;
}
```

Örnek: Dizi Ortalama ve Standart Sapma

```
#include <stdio.h>
#include <math.h>
#define N 10
int main(){
   int   i;
   float x[N], toplam = 0.0, ort, std_sap = 0.0;
   /* ortalama hesabı */
   for(i=0; i<N; i++){
      printf("%d. sayi : ",i+1);
      scanf("%f",&x[i]);
      toplam += x[i];
   }
   ort = toplam/N;
   /* standart sapma hesabı */
   for(toplam = 0.0, i=0; i<N; i++)
       toplam += pow(x[i]-ort, 2.0);
   std_sap = sqrt( toplam/(N-1) );
   printf("Ortalama        = %f\n",ort);
   printf("Standart sapma = %f\n",std_sap);
   return 0;
}
```

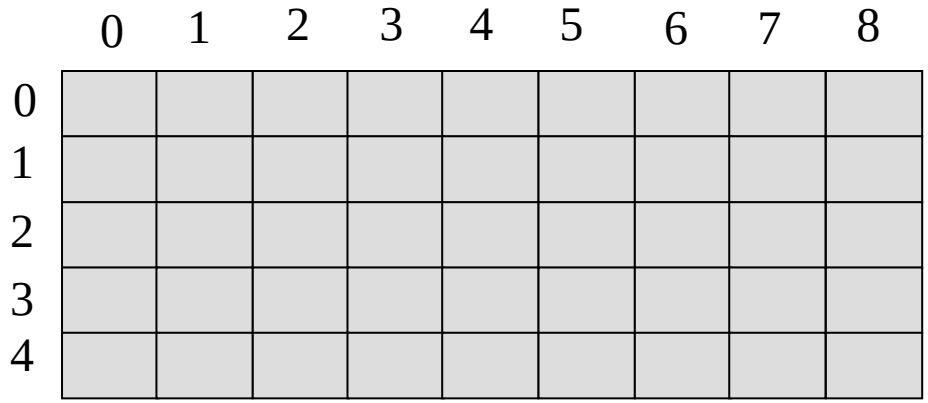
Örnek: Rasgele Sayı Oluşturucu

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

int main() {
  int c, n;
  // Initialization, should only be called once
  srand(time(NULL));

  printf("Ten random numbers in [1,100]\n");

  for (c = 1; c <= 10; c++) {
    n = rand() % 100 + 1;
    printf("%d\n", n);
  }
  return 0;
}
```

rand(); //Returns a pseudo-random integer between 0 and RAND_MAX.

Çok Boyutlu Diziler

- Bir dizi birden fazla boyutlu olabilir.
- Örneğin 3x4 bir matris için 2 boyutlu bir dizi kullanırız.
- Üç boyutlu öklid uzayındaki x, y, z noktalarını saklamak için de 3 boyutlu bir diziyi tercih ederiz.
- i satırında ve j sütunundaki elemana ulaşmak için M[i][j] şeklinde yazarız.
  - Örneğin, 2-boyutlu dizi (matris) aşağıdaki gibi tanımlanır
  - int M[5][9]; /* 5 satır ve 9 sütundan oluşuyor */
  - Kavramsal olarak, M dizisi aşağıdakine benzer.



2 Boyutlu Dizilere Ulaşmak

```
/* değer atama */
for (i=0; i<5; i++){
  for (j=0; j<9; j++){
    M[i][j] = 0;
  }
}
```

```
/* Toplama */
toplam = 0;
for (i=0; i<5; i++){
  for (j=0; j<9; j++){
    toplam += M[i][j];
  }
}
```

```
/* min ve max bulma */
min = M[0][0];
Max = M[0][0];
for (i=0; i<5; i++){
  for (j=0; j<9; j++){
    if (M[i][j]<min) min=M[i][j];
    if (M[i][j]>max) max=M[i][j];
  }
}
printf("min: %d, max: %d\n", min, max);
```

Çok Boyutlu Dizilere İlk Değer Atama

- Çok boyutlu dizilere ilk değer atamak için iç içe geçmiş bir boyutlu dizi yükleyicilerini kullanabiliriz.

```
int M[5][9] = { {1, 1, 1, 1, 0, 1, 1, 1, 1},
                {0, 1, 0, 1, 0, 1, 0, 1, 0},
                {1, 0, 0, 1, 1, 1, 0, 0, 1},
                {0, 0, 0, 0, 1, 1, 1, 1, 1},
                {1, 1, 1, 1, 0, 0, 0, 1, 1}};
```

- Eğer ilk değerler çok boyutlu dizinin elemanlarından az ise kalanlar 0 ile doldurulur.

```
int M[5][9] = { {1, 1, 2, 1, 2, 0, 0, 1, 1},
                {0, 0, 0, 1, 1, 1, 1, 2, 1}};
        /* 2, 3 ve 4 üncü satırlar 0 ile doldurulacak*/
```

Çok Boyutlu Dizilere İlk Değer Atama

- Eğer içerdeki liste bir satırın eleman sayısından az ise satırın kalan kısmı 0 ile doldurulur.

```
int M[5][9] = { {1, 1, 0, 0, 1, 1, 1, 1, 1},
                {0, 1, 1, 2, 1, 1},
                {1, 1, 2, 2, 3}};

/* M[1][6], M[1][7], M[1][8] 0 olacaklar*/
/* M[2][5], M[2][6], M[2][7], M[2][8] 0 sıfır olacaklar*/
/* 3 ve 4 üncü satırlar tümü 0 ile doldurulacaklar */
```

Daha Yüksek Boyutlu Diziler

- Bir dizi herhangi bir boyutta tanımlanabilir

```
int Kup[8][8][8];    /* ebatları 8 olan bir küp */
int Prizma[4][6][10]; /* ebatları 4x6x10 olan bir
                       * dikdörtgenler prizması
                       */
/* ilk değerler atanabilir*/
float A[4][6][8] = {{{1, 2, 3}, {3, 4}}, {{3, 4}}};

Kup[2][3][4] = 2;
Prizma[3][5][8] = 6;
A[0][0][4] = 3.34;
```

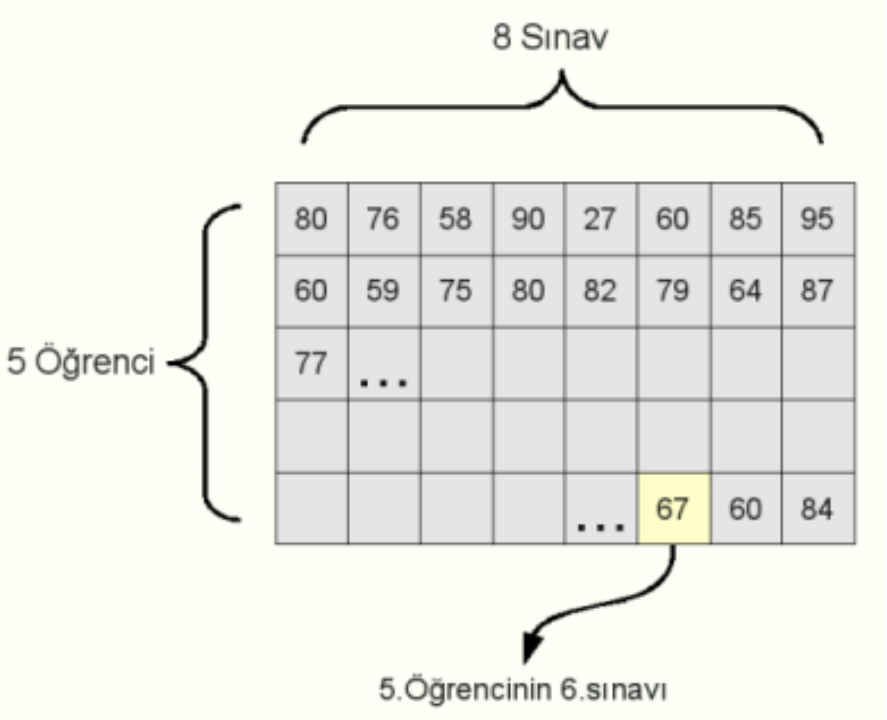
Çok Boyutlu Diziler

- 5 kişilik bir öğrenci grubu için 8 adet test uygulansın. Bunların sonuçlarını saklamak için 2 boyutlu bir dizi kullanalım.

```
#include<stdio.h>
int main( void ) {
        // 5 adet ogrenci icin 8 adet sinavi  temsil etmesi icin bir ogrenci tablosu
        // olusturuyoruz. Bunun icin 5x8 bir matris  yaratilmasi gerekiyor.
        int ogrenci_tablosu[ 5 ][ 8 ];
        int i, j;
        for( i = 0; i < 5; i++ ) {
                for( j = 0; j < 8; j++ ) {
                        printf( "%d no.'lu ogrencinin ", ( i + 1 ) );
                        printf( "%d no.'lu sınavı> ", ( j + 1 ) );
                        // Tek boyutlu dizilerdeki gibi deger atiyoruz
                        scanf( "%d", &ogrenci_tablosu[ i ][ j ] );
                }
        }
        return 0;
}
```

Çok Boyutlu Diziler

- Bu programı çalıştırıp, öğrencilere çeşitli not değerleri atadığımızı düşünelim. Bunu görsel bir şekle sokarsak, aşağıdaki gibi bir çizelge oluşur:



- Tabloya bakarsak, 1.öğrenci sınavlardan, 80, 76, 58, 90, 27, 60, 85 ve 95 puan almış gözüküyor. Ya da 5.öğrencinin, 6.sınavından 67 aldığını anlıyoruz. Benzer şekilde diğer hücrelere gerekli değerler atanıp, ilgili öğrencinin sınav notları hafızada tutuluyor.

Örnek: İki Matrisin Toplamı

```
#include <stdio.h>
#define SAT 2
#define SUT 3

int main(){
   int a[SAT][SUT] = {5, 3, 7,  0, 1, 2};

   int b[SAT][SUT] = {1, 2, 3,  4, 5, 6};
   int c[SAT][SUT];
   int i, j;

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

Örnek: İki Matrisin Toplamı

```
    puts("B Matrisi:");
    for(i=0; i<SAT; i++){
        for(j=0; j<SUT; j++)
            printf("%4d",b[i][j]);
        printf("\n");
    }
    puts("\nC Matrisi:");
    for(i=0; i<SAT; i++){
         for(j=0; j<SUT; j++){
            c[i][j] = a[i][j] + b[i][j];
            printf("%4d",c[i][j]);
         }
         printf("\n");
    }
  return 0;
}
```

Örnek: Diziyi Matrise Çevirme

```
#include <stdio.h>
#include <stdlib.h>
 int main(){
    int dizi[100];
    int a[100][100];
    int i, j, n, x, sat, sut;
    printf("Diziniz kac elemandan olusacak? > ");
    scanf("%d",&n);
     for(x=0; x<n; x++){
        printf("Dizinin [%d] . elemanini gir > ",x+1);
        scanf("%d",&dizi[x]);
    }
    printf("\nMatrisin Satir sayisini gir > ");
    scanf("%d",&sat);
    printf("matrisin sutun sayisini gir > ");
    scanf("%d",&sut);
```

Örnek: Diziyi Matrise Çevirme

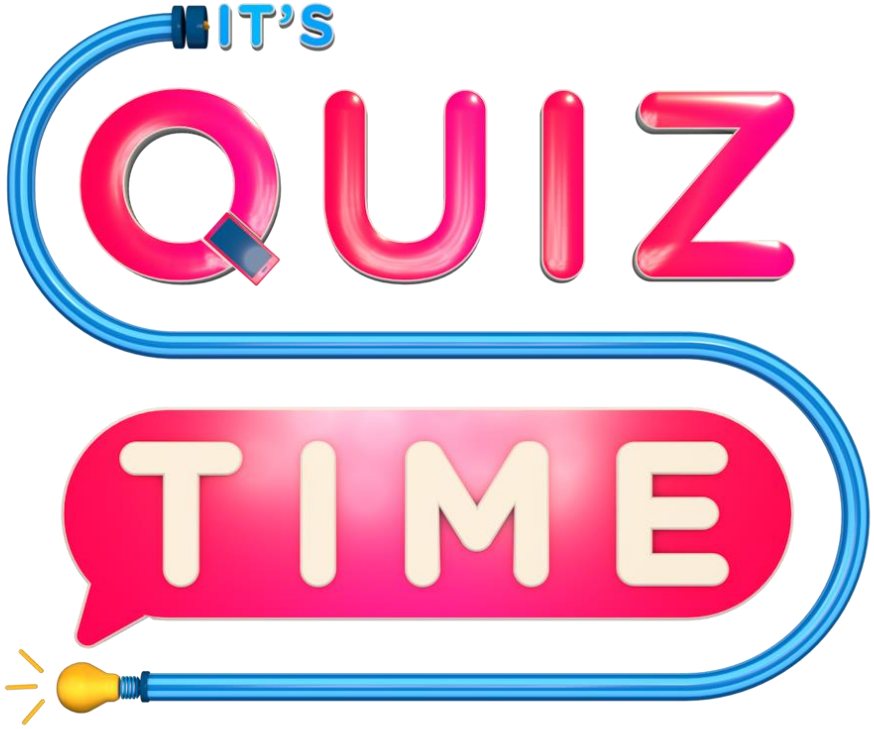
```
if(n%sat==0 && n%sut==0){
        x=0;
        for(i=0; i<sat; i++){
           for (j=0; j<sut; j++){
              a[i][j]=dizi[x];
              x++;
           }
        }
        printf("\n\nMATRIS > \n");
        for(i=0; i<sat; i++){
           for(j=0; j<sut; j++)
              printf("%3d", a[i][j]);
           printf("\n");
        }
    } else
           printf("HATA! Dizi Eleman Sayisi Satir veya Sutuna Tam bolunmelidir");
     return 0;
}
```

Örnek: Simetrik Matris

```
#include <stdio.h>
#include <conio.h>
 int main(){
     int a[100][100];
    int simetri=1;
    int x, y, i, j;
     printf("Matrisin satir sayisini gir > ");
    scanf("%d",&x);
    printf("Matrisin sutun sayisini gir > ");
    scanf("%d",&y);
     printf("Matrisin degerlerini gir > ");
     for(i=0; i<x; i++) {
        for(j=0; j<y; j++) {
            printf("\n Deger [%d] [%d] --> ", i+1, j+1);
            scanf("%d", &a[i][j]);
        }
    }
```

Örnek: Simetrik Matris

```
//NOT: a[i][j]==a[j][i] ise bu matris simetriktir.
for(i=0; i<x; i++){
   for(j=0; j<y; j++){
        if(a[i][j]!=a[j][i])
             simetri=0;
             break;
        }
   }
}
 if(simetri==1)
    printf("\nMatris Simetriktir.\n");
 else
    printf("\nMatris Simetrik Degildir\n");

return 0;
}
```



Write C program to find reverse of the given array.

Kaynaklar

- Doç. Dr. Fahri Vatansever, “Algoritma Geliştirme ve Programlamaya Giriş”, Seçkin Yayıncılık, 12. Baskı, 2015.
- J. G. Brookshear, “Computer Science: An Overview 10th Ed.”, Addison Wisley, 2009.
- Kaan Aslan, “A’dan Z’ye C Klavuzu 8. Basım”, Pusula Yayıncılık, 2002.
- Paul J. Deitel, “C How to Program”, Harvey Deitel.
- Bayram AKGÜL, C Programlama Ders notları