DİZİLER

Dizi Nedir?

Şu ana kadar en basit veri yapıları (data structure) olan değişkenlerle karşılaştık. Programlamada kullanılan bir başka veri yapısı da dizilerdir (array).

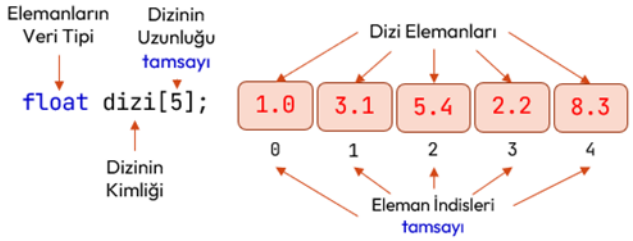
Matematikte diziler; bir sıralı elemanlardan oluşan bir listedir. Sıralı elemanların sayısına dizinin uzunluğu (array size) denir. Elemanların sırası indis (index) olarak adlandırılır. Kümenin aksine aynı elemanlar dizide farklı konumlarda birkaç kez bulunabilir. Elemanlara terim adı da verilir.

Örneğin Elemanları 1'den 10'a kadar sayıların karesinden oluşan bir dizi; $(a_k)_{k=1}^{10}, a_k = k^2$ veya $(a_1, a_2, a_3, \ldots, a_{10})$ veya $(1,4,9,\ldots,100)$ olarak gösterilir. Örneğin;

- $('K','İ','T','A','P')$ Dizisi ile $('P','A','T','İ','K')$ dizisi birbirinden farklıdır. Aynı elemanlardan oluşmasına rağmen elemanların sıraları farklıdır.
- $(1,3,5,1,2,1,7)$ Dizisinde birden farklı indiste aynı 1 elemanı vardır. Bu dizinin geçersiz olduğu anlamına gelmez.
- Matematikte sonsuz elemanı olan sonsuz diziler tanımlanabilir. Ama programlamada hep sonlu elemanlı diziler yani uzunluğu (size) belirli diziler kullanılır.

Tek boyutlu Diziler

C++ Dilinde sonlu elemanlı diziler aşağıdaki gibi tanımlanır;



Şekil 21. Tek Boyutlu Dizi Tanımlama

1. Dizi içinde, benzer veri tipindeki (data type) veri öğelerini bulundurur ve bu öğeler bitişik bellek bölgesini paylaşırlar.
2. Dizi elemanları, ilkel veri tipleri (int, float, char) veya daha sonra göreceğimiz kullanıcı tanımlı tip olan yapı (struct) veya gösterici (pointer) olabilir.
3. Aynı değer birkaç farklı yerde dizi içinde bulunabilir.
4. Dizinin veri tipi, dizideki elemanlarının veri tipini belirler. Bir başka deyişle elemanların veri tipi dizinin veri tipiyle aynı olmalıdır.
5. Dizinin uzunluğu tamsayı değişmez (integer literal) olup dizi kimliklendirmesi sırasında belirtilmelidir. Derleyici buna göre bellekte yer ayırır. Dolayısıyla dizi, bir kez kimliklendirildiğinde dizinin uzunluğu değiştirilemez.
6. Dizinin indisi (index) de her zaman sıfırdan başlar ve sıra belirttiğinden her zaman tamsayıdır (integer).

C dilinde beş tamsayı elemanı olan bir dizi aşağıdaki şekilde tanımlanır.

```
int tamsayiDizisi[5];
```

Beş gerçek sayı elemanı olan bir dizi ise aşağıdaki şekilde tanımlanır.

```
float reelsayiDizisi[5];
```

Dizinin elemanlarına tanımlama sırasında başlangıç değeri (initial value) değer verilebilir;

```
int tamsayiDizisi1[5]= {1, 2, 3, 4, 5}; //Elemanlar sırasıyla 1,2,3,4,5
int tamsayiDizisi2[5]= {0}; //Elemanların Hepsi 0
```

Dizinin elemanlarına hepsine başlangıç değeri verilmeyebilir;

```
float a[5] = {1.0, 2.0, [4] = 12.0}; // 1.Eleman 1.0, 2.Elaman 2.0 ve 5.eleman 12.0
```

Dizinin elemanlarına ayrı ayrı erişilip değerleri değiştirilebilir ya da yeni değer ataması yapılabilir;

```
int a[5]={0};
a[0]=1; //Birinci elemana 1 değeri atandı
a[1]++; //İkinci elemanın değeri artırıldı
a[2]--; //Üçüncü elemanın değeri azaltıldı
a[4]=12; //Beşinci elemana 12 değeri atandı.
```

Diziler belleği verimli kullanan ve tek bir değişken ile elemanlara erişim sağlayan bir çözüm sunar. Bir dizideki elemanlar, bellekte bitişik konumda yer aldığından herhangi bir öğeye kolaylıkla erişebiliriz. Dizi kullanmanın başlıca üstünlükleri şunlardır:

– İndisleri kullanarak dizi öğelere rastgele ve hızlı erişim. Her öğenin bir indisi (index) olduğundan doğrudan erişilebilir ve değiştirilebilir.
– Birden fazla öğeden oluşan tek bir dizi oluşturduğundan daha az kod satırı yazılır.
– Daha az kod satırı yazılarak sıralama yapılabilir.

Dizi kullanmanın zayıf yönleri ise;

– Kimliklendirme sırasında karar verilen değişmez (literal) sayıda elemanın üzerinde işlem yapılır.
– Dizi dinamik değildir. Araya elaman ekleme veya çıkarma yapılamaz.

Örnek olarak klavyeden girilen 5 adet tamsayıyı, giriş sırasının tersinden ekrana yazan C programını verebiliriz;

```
#include <iostream>
using namespace std;
int main() {
  int dizi[5]; /* 5 elemanı tamsayı olan bir dizi tanımlandı*/
  int indis; /* dizi elemanlarını gezecek ve indis olarak kullanılacak
                tamsayı bir değişken tanımlandı */

  for (indis =0; indis <5; indis ++){
    cout << indis << ". sayiyi girin:";
    cin >> dizi[ indis ]; /* indis ile belirtilen elemana klavyeden
                             olunan değer atanıyor */
  }

  cout << "Tersten Sayilar:" << endl;
  for (indis=4; indis >=0; indis --)
    cout << "dizi[" << indis << "]=" << dizi[ indis ] <<endl;
}
/* Program Çıktısı:
0. sayiyi girin:1
1. sayiyi girin:3
2. sayiyi girin:6
3. sayiyi girin:8
4. sayiyi girin:3
Tersten Sayilar:
dizi[4]=3
dizi[3]=8
dizi[2]=6
dizi[1]=3
dizi[0]=1

...Program finished with exit code 0
*/
```

Bir başka örnek; Klavyeden girilen 10 adet sınav notuna göre, ortalamanın üstünde olan notları ekrana yazan C programı;

```
#include <iostream>
using namespace std;
#define BOYUT 10
int main() {
    unsigned notlar[BOYUT]; /* Her terimi pozitif olan 10 elemanlı dizi */
    float ortalama, toplam=0;
    int indis; //indis için tamsayı değişken
    for (indis=0; indis<BOYUT; indis++){
        cout << indis << ". notu girin:";
        cin >> notlar[indis];
        toplam+=notlar[indis]; // Her eleman girildiğinde toplanıyor
   }
   ortalama=toplam/BOYUT;
   cout << "Ortalamanın (" << ortalama << ") Üzerindeki Notlar:"<< endl;
   for (indis=0; indis<BOYUT; indis++)
       if (notlar[indis]>ortalama)
           cout << "notlar [" << indis << "]=" << notlar[indis] << endl;
}
/* Program Çıktısı:
1. notu girin:85
2. notu girin:45
3. notu girin:60
4. notu girin:75
5. notu girin:40
6. notu girin:25
7. notu girin:35
8. notu girin:60
9. notu girin:50
Ortalamanın (57.5) Üzerindeki Notlar:
notlar [0]=100
notlar [1]=85
notlar [3]=60
notlar [4]=75
notlar [8]=60

...Program finished with exit code 0
*/
```

Bir başka örnek olarak 10 adet satış miktarlarının, ortalamaya olan uzaklıkları yani sapma (deviation) ve karesi alınmış sapmalar yani varyans (variance) ile standart sapmayı yazdıran C program verilebilir.

```
#include <iostream>
#include <iomanip>  // setw() ve setprecision için
#include <cmath>
using namespace std;
#define BOYUT 10
int main() {
    float satislar[BOYUT]={100.0,850.0,500.0,600.0,750.0,650.0,450.0,800.0,900.0,110.0};
    float ortalama, toplam=0, varyansToplam=0, standartSapma;
    int indis;
    for (indis=0; indis<BOYUT; indis++)
        toplam+= satislar[indis];
    ortalama=toplam/BOYUT;
    cout << "Ortalama:" << ortalama << endl;
    cout << setw(5);
    for (indis=0; indis<BOYUT; indis++) {
        float sapma= satislar[indis]-ortalama;
        varyansToplam+=sapma*sapma;
```

```
        cout << "Sapma: " << setw(5) << sapma
             << " Varyans: " <<  setw(8) << sapma*sapma << endl;
        /*
         setw(5): izleyen değişken konsola yazılırken 5 karakter
                  genişliğinde yazılsın.
         setw(8): izleyen değişken konsola yazılırken 8 karakter
                  genişliğinde yazılsın. */
    }
    standartSapma=sqrt(varyansToplam/BOYUT);
    cout << "Standart Sapma:"<< setprecision(8) << standartSapma;
    /* setprecision(8): izleyen reel sayı değişkeni konsola
                        yazılırken 9 karakter genişliğinde yazılsın. */
}
/* Program Çıktısı:
Ortalama:571
Sapma:  -471 Varyans:   221841
Sapma:   279 Varyans:    77841
Sapma:   -71 Varyans:     5041
Sapma:    29 Varyans:      841
Sapma:   179 Varyans:    32041
Sapma:    79 Varyans:     6241
Sapma:  -121 Varyans:    14641
Sapma:   229 Varyans:    52441
Sapma:   329 Varyans:   108241
Sapma:  -461 Varyans:   212521
Standart Sapma:270.49768

...Program finished with exit code 0
*/
```

Ortalamaya olan uzaklıklar yani sapma verilerin ortalamaya ne kadar yakın olduğunu gösteren dağılımdır. Sapmaların toplamı sıfır olabileceğinden karelerinin toplamı yani varyans hesaplanır. Varyansın eleman sayısına bağlı karekökü de standart sapmayı verir. Standart sapmanın büyük olması verilerin ortalamadan daha uzak yayıldıklarını; küçük bir standart sapma ise verilerin ortalama etrafında daha çok yakın gruplaştıklarını gösterir.

Bir başka örnek olarak 5 elemanlı diziye girilen elemanlardan tekil (unique) olanlarını bulan program verilebilir.

```
#include <iostream>
using namespace std;
#define BOYUT 5
int main() {
    int dizi[BOYUT];
    int indis;
    for (indis=0; indis<BOYUT; indis++) {
        cout << indis << ". Sayıyı Girin:";
        cin >> dizi[indis];
    }
    cout << "Dizi içinde tekil (unique) olan rakamlar:"<< endl;
    for (indis=0; indis<BOYUT; indis++) {
        int indis2, sayac = 0;  // Her elemanda sayacı sıfırla
        for (indis2 = 0; indis2 < BOYUT; indis2++)
            if (indis != indis2) //elemanın kendisini kontrol etmiyoruz
                if (dizi[indis] == dizi[indis2])
                    sayac++;
        if (sayac == 0)
            cout << "dizi["<< indis << "]:" << dizi[indis]
                 << " tekil olarak dizide bulundu."<< endl;
  }
}
```

```
/* Program Çıktısı:
0. Sayıyı Girin:23
1. Sayıyı Girin:12
2. Sayıyı Girin:23
3. Sayıyı Girin:14
4. Sayıyı Girin:14
Dizi içinde tekil (unique) olan rakamlar:
dizi[1]:12 tekil olarak dizide bulundu.

...Program finished with exit code 0
*/
```

Bir başka örnek olarak 5 elemanlı diziye girilen en büyük elemanına en küçük elemanını ekleyen program verilebilir;

```
#include <iostream>
#include <iomanip> // setw()
using namespace std;
#define BOYUT 5
int main() {
    int dizi[BOYUT]={10,12,3,4,6};
    int enBuyuk, enKucuk, enKucukIndex, enBuyukIndex, i ;
    cout << "ÖNCE:" << endl; //Dizinin İlk hali konsola yazdırılıyor.
    for (i = 0; i < BOYUT; i++)
        cout << setw(4) << dizi[i];
    cout << endl;
    enKucuk=dizi[0]; enBuyuk=dizi[0]; /* İlk elemanlar hem en küçük
                                         hem de en büyük olsun. */
    enKucukIndex=0,enBuyukIndex=0; /* En küçük ve en büyük elemanların
                                      indisi 0 olsun. */
    for (i=0; i<BOYUT; i++){ /* En küçük ve en büyük eleman ile
                                indislerini bulan kısım. */
        if (dizi[i]<enKucuk) {
            enKucuk=dizi[i];
            enKucukIndex=i;
        }
        if (dizi[i]>enBuyuk) {
            enBuyuk=dizi[i];
            enBuyukIndex=i;
       }
    }
    cout << "En Büyük Eleman: dizi[" << enBuyukIndex << "]:"
         << dizi[enBuyukIndex] << endl;
    cout << "En Küçük Eleman: dizi["<< enKucukIndex << "]: "
         << dizi[enKucukIndex] << endl;
    dizi[enBuyukIndex]+=dizi[enKucukIndex]; //En büyük elemana en küçüğünü ekleme
    cout << "SONRA:" << endl; //Dizinin son hali konsola yazdırılıyor.
    for (i = 0; i < BOYUT; i++)
        cout << setw(4) << dizi[i];
}
/* Program Çıktısı:
ÖNCE:
  10  12   3   4   6
En Büyük Eleman: dizi[1]:12
En Küçük Eleman: dizi[2]: 3
SONRA:
  10  15   3   4   6

...Program finished with exit code 0
*/
```

İki Boyutlu Diziler

Şu ana kadar gördüğümüz tek boyutlu dizilerdi. İki boyutlu diziler matematikten de bildiğimiz üzere matris (matrix) olarak adlandırılır. İki boyutlu dediğimizde en-boy ve satır-sütun gibi kavramlar akla gelmelidir. Matris işlemleri gibi bazı problemlerde; bir dizinin her bir elemanının da dizi olması istenir. Bu tür iki boyutlu dizilerde en içteki dizinin boyutu Kimliklendirmede sağda yer alır.

Aşağıda iki boyutlu matris tanımlamalarına örnek verilmiştir;

```
float matris[2][3];  /* Her bir elemanı 3 elemanlı bir dizi olan 2 boyutlu dizi:
                        2 satır üç sütunlu bir matris */
//Aşağıda 2x3 matrise ilk değer verme:
float matris2[2][3]= {
                       {1.0,2.0,3.0}, //Birinci Satır: 3 Elemanlı bir dizi
                       {2.0,4.0,6.0}  //İkinci Satır: 3 Elemanlı bir dizi
                     };
int karematris[2][2];
//2x2 matrise ilk değer verme:
int karematris2[2][2]= {
                         {1,2}, //Birinci Satır: 2 elemanlı bir dizi
                         {5,6}  //Birinci Satır: 2 elemanlı bir dizi
                        };
int karematris3[3][3];
//İlk Değerlieri DÜZ verme (flat initialization):
int karematris4[2][2]= { 1, 2, 3, 4 }; /* Toplamda 2x2=4 elemanı var.
                                          Bütün elemanlara peşpeşe ilk
                                          değer verilebilir */
int karematris5[3][3]= { 11, 22, 33, 21, 22, 23, 31, 32, 33};
/* Toplamda 3x3=9 elemanı var. Bütün elemanlara peşpeşe ilk değer verilebilir */
```

Örnek olarak 3x4'lük matris elemanlarını klavyeden girip, tablo halinde ekrana yazdıran programı verebiliriz;

```
#include <iostream>
#include <iomanip> // setw()
using namespace std;
#define SATIR 3
#define SUTUN 4
int main() {
    int matris[SATIR][SUTUN];
    int i,j;
    /*
    for (j=0; j<SUTUN; j++)  //Birinci Satır Okuyalım
         cin >> matris[0][j];
    for (j=0; j<SUTUN; j++)  //İkinci Satır Okuyalım
        cin >> matris[1][j];
    for (j=0; j<SUTUN; j++)  //Üçüncü Satır Okuyalım
        cin >> matris[2][j];
    //Bunun yerine iç içe iki for tanımlanır;
    */
    for (i=0; i<SATIR; i++) //İkinci boyut için i indisi
        for (j=0; j<SUTUN; j++)  //Birinci boyut için j indisi
            cin >> matris[i][j];
    cout <<"TABLO:"<< endl;
    for (i=0; i<SATIR; i++) {  //İkinci Boyut İçin
        for (j=0; j<SUTUN; j++)  //Birinci Boyut İçin
            cout << setw(4) << matris[i][j];
        cout << endl;
    }
}
/* Program Çıktısı:
```

```
1 3 4 5 8 7 6 5 4 4 5 6 8 9 0 2
TABLO:
   1   3   4   5
   8   7   6   5
   4   4   5   6

...Program finished with exit code 0
*/
```

Bir başka örnek olarak elemanları klavyeden girilen 3x2'lık matrisin satır ve sütun toplamlarını ekrana yazan programı verilebilir;

```
#include <iostream>
using namespace std;
#define SATIR 3
#define SUTUN 2
int main() {
    int matris[SATIR][SUTUN]; //Matris Tanımı
    int i,j;  //i sutun için, j satır için tanımlandı
    for (i=0; i<SATIR; i++) {  //İkinci Boyut (SATIR) İçin
        cout << i << ". Satırı Girin:";
        for (j=0; j<SUTUN; j++)  //Birinci Boyut (SUTUN) İçin
            cin >> matris[i][j];
    }
    int satirToplamlar[SATIR]={0}; //Bütün elemanları 0 olan dizi
    for (i=0; i<SATIR; i++) {  // İkinci Boyut İçin
        for (j=0; j<SUTUN; j++) // Birinci Boyut İçin
            satirToplamlar[i]+=matris[i][j];
        /* İçdeki for bitince tüm satırdaki elemanlar toplanmış oldu */
    } /* Dışdaki for ile de her bir satır için toplam tekrarlanıyor */
    cout << "Satır Toplamları:" << endl;
    for (i=0; i<SATIR; i++)
        cout << i << ". Satır Toplamı:" <<satirToplamlar[i] << endl;
    int sutunToplamlar[SUTUN]={0}; //Bütün elemanları 0 olan dizi
    for (j=0; j<SUTUN; j++) // Birinci Boyut İçin
        for (i=0; i<SATIR; i++) {  // İkinci Boyut İçin
            sutunToplamlar[j]+=matris[i][j];
        /* İçdeki for bitince tüm sutundaki elemanlar toplanmış oldu*/
    }/* Dışdaki for ile de her bir sutun için toplam tekrarlanıyor */
    cout << "Sütun Toplamları:" << endl;
    for (j=0; j<SUTUN; j++)
        cout << j << ". Sütun Toplamı:" << sutunToplamlar[j] << endl;
}
/* Program Çıktısı:
0. Satırı Girin:2 8
1. Satırı Girin:13 17
2. Satırı Girin:16 34
Satır Toplamları:
0. Satır Toplamı:10
1. Satır Toplamı:30
2. Satır Toplamı:50
Sütun Toplamları:
0. Sütun Toplamı:31
1. Sütun Toplamı:59

...Program finished with exit code 0
*/
```

Çok Boyutlu Diziler

Şu ana kadar gördüğümüz tek boyutlu diziler veya iki boyutlu diziler olan matrislerdir. Çok boyutlu dizilere örnek olarak En-Boy-Yükseklik içeren 3 boyutlu diziler verilebilir. Üç boyutlu dizilerde dizinin her bir elemanı bir matristir. Çok boyutlu dizilerde en içteki dizinin boyutu kimliklendirmede sağda yer alır. Aşağıdaki örnekte; çeşitli boyutlarda diziler tanımlanmıştır,

```
int ucbuyutludizi1[3][2][2];
// ucbuyutludizi1: elemanları 3 adet 2x2 matris olan üç boyutlu bir dizi
int ucbuyutludizi1[2][3][3];
// ucbuyutludizi2: elemanları 2 adet 3x3 matris olan üç boyutlu bir dizi
int ucbuyutludizi1[4][3][2];
// ucbuyutludizi3: elemanları 4 adet 3x2 matris olan üç boyutlu bir dizi
int dortbuyutludizi1[5][2][3][2];
/* dortbuyutludizi1: elemanları 5 adet olan ve her bir elemanı 2 adet 3x2 matris olan dört
boyutlu bir dizi */
```

Aşağıda üç boyutlu dizilerde bir ilk değer verme kod örneği verilmiştir;

```
#define DERINLIK 3
#define SATIR 3
#define SUTUN 3
int ucBoyutluDizi[DERINLIK][SATIR][SUTUN]={
    { {1,2,3},{4,5,6},{7,8,9} },
    { {11,12,13},{14,15,16},{17,18,19} },
    { {21,22,23},{24,25,26},{27,28,29} }
};
```

Çok boyutlu dizi örneği olarak şöyle bir örnek verilebilir; 3 farklı ders alan 10 öğrenci, her bir dersten 4 değişik not almaktadır. Bu notların ağırlıkları %10, %30, %20 ve %40'tır. Notlar rastgele 0 ile 100 arasında verilecektir. Her bir sınavın ortalaması ile her bir öğrencinin ağırlıklı not ortalamalarını bulan C programı yazınız.

```
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;
#define KACDERS 3
#define KACOGRENCI 10
#define KACDEGISIKNOT 4
int main() {
    int notlar[KACDERS][KACDEGISIKNOT][KACOGRENCI];
    int agirlik[KACDEGISIKNOT]={10,30,20,40};
        //Odevler:%10, Vize:%30, Hızlı Sınav:%20, Final:%40
    int i,j,k; //indis değişkenleri
    srand(time(NULL)); // rastgele için
    for (k=0; k< KACDERS; k++)
        for (i=0; i< KACOGRENCI; i++)
            for (j=0; j< KACDEGISIKNOT; j++)
                notlar[k][j][i]=rand()%101;
    //Her bir sınavın Ortalaması hesaplanacak:
    for (k=0; k< KACDERS; k++) {
        cout << k << ".Ders İçin:"<< endl;
        for (j=0; j<KACDEGISIKNOT; j++)  {
            float sinavToplam=0.0;
            for (i=0; i<KACOGRENCI; i++) {
                sinavToplam+=notlar[k][j][i];
            }
            cout << j << ". Sınav Ortalaması:" << sinavToplam/KACOGRENCI << endl;
        }
    }
    //Her bir Öğrencinin Ağırlıklı Notu hesaplanacak:
```

```
    for (k=0; k< KACDERS; k++) {
        cout << k << ".Ders İçin:" << endl;
        for (i=0; i<KACOGRENCI; i++) {
            float agirlikliNot=0.0;
            for (j=0; j<KACDEGISIKNOT; j++)  {
                agirlikliNot+=notlar[k][j][i]*agirlik[j]/100.0;
            }
           cout << i << ". Öğrenci Ortalaması:" << agirlikliNot << endl;
        }
    }
}
/* Program Çıktısı:
0.Ders İçin:
0. Sınav Ortalaması:58.4
1. Sınav Ortalaması:46.7
2. Sınav Ortalaması:54.6
3. Sınav Ortalaması:49.4
1.Ders İçin:
0. Sınav Ortalaması:41
1. Sınav Ortalaması:48.6
2. Sınav Ortalaması:37.1
3. Sınav Ortalaması:51.7
2.Ders İçin:
0. Sınav Ortalaması:54.7
1. Sınav Ortalaması:37.7
2. Sınav Ortalaması:43.3
3. Sınav Ortalaması:60.1
0.Ders İçin:
0. Öğrenci Ortalaması:62.2
1. Öğrenci Ortalaması:40.7
2. Öğrenci Ortalaması:60.3
3. Öğrenci Ortalaması:42.1
4. Öğrenci Ortalaması:51.9
5. Öğrenci Ortalaması:33.4
6. Öğrenci Ortalaması:58.7
7. Öğrenci Ortalaması:49.9
8. Öğrenci Ortalaması:46.1
9. Öğrenci Ortalaması:60
1.Ders İçin:
0. Öğrenci Ortalaması:49.6
1. Öğrenci Ortalaması:25.9
2. Öğrenci Ortalaması:30.5
3. Öğrenci Ortalaması:53
4. Öğrenci Ortalaması:55.4
5. Öğrenci Ortalaması:69.3
6. Öğrenci Ortalaması:48.3
7. Öğrenci Ortalaması:68.8
8. Öğrenci Ortalaması:38
9. Öğrenci Ortalaması:29
2.Ders İçin:
0. Öğrenci Ortalaması:28.3
1. Öğrenci Ortalaması:44
2. Öğrenci Ortalaması:42.9
3. Öğrenci Ortalaması:27
4. Öğrenci Ortalaması:52.9
5. Öğrenci Ortalaması:79
6. Öğrenci Ortalaması:51.5
7. Öğrenci Ortalaması:40
8. Öğrenci Ortalaması:78.1
9. Öğrenci Ortalaması:51.1
```

```
...Program finished with exit code 0
*/
```

Parametre Olarak Diziler

Parametre olarak gönderilen dizinin boyutu kadar köşeli parantez açılır ve kapatılır, ilk köşeli parantez içerisine eleman sayısını ifade eden değer yazılmaz, fakat diğerlerine eleman sayıları verilmek zorundadır. Bu durumda tek boyutlu diziler parametre olacak ise dizinin boyutu yazılmayabilir. Aşağıda dizileri parametre olarak alan fonksiyon bildirim örnekleri bulunulmaktadır;

```
void diziIsleyenFonksiyon1(int tekBoyutluDizi[],int boyut);
void diziIsleyenFonksiyon2(int pDizi[][2],int ikinciBoyut,int birinciBoyut);
void diziIsleyenFonksiyon3(int pDizi[][3][2],
                                int ucuncuBoyut,
                                int ikinciBoyut,
                                int birinciBoyut);
```

Aynı fonksiyonları aşağıdaki şekilde çağırabiliriz;

```
unsigned notlar[10]; //10 öğrenci notu barındırır.
unsigned dersNotlari[2][10]; //2 dersi alan 10 öğrenci için notlar;
unsigned bolumDersNotlari[4][2][10];
         //4 bölümde 2 dersi alan 10 öğrenci için notlar;
//...
diziIsleyenFonksiyon1(notlar,10);
diziIsleyenFonksiyon2(dersNotlari,2,10);
diziIsleyenFonksiyon3(bolumDersNotlari,4,2,10);
//...
```

Aşağıda bir diziyi okuyan, ekrana yazan ve eleman ortalamalarını hesaplayan fonksiyonlara sahip bir program verilmiştir;

```
#include <iostream>
#include <iomanip> // setw()
using namespace std;
void diziOku(int pDizi[],int pUzunluk);
void diziYaz(int pDizi[],int pUzunluk);
float diziOrtalama(int pDizi[],int pUzunluk);
int main() {
  int dizi[5];
  float ortalama;
  diziOku(dizi,5); /* fonsiyon geri dönüş değeri olmadığından
                      bir değişkene atanmıyor! */
  diziYaz(dizi,5); /* fonsiyon geri dönüş değeri olmadığından
                      bir değişkene atanmıyor! */
  ortalama=diziOrtalama(dizi,5); /* fonksiyon geri dönüş değeri
                                    bir değişkene atanıyor. Atandığı değişkenin
                                    tipi ile fonksiyonun geri dönüş tipi
                                    aynı olmalı! */
  cout << "Dizi Ortalaması:" <<ortalama ;
  return 0;
}
void diziOku(int pDizi[],int pUzunluk){
  int sayac;
  cout << pUzunluk << " Elemanlı Dizi Okunacaktır:";
  for (sayac=0; sayac < pUzunluk; sayac++)
     cin >> pDizi[sayac];
}
void diziYaz(int pDizi[],int pUzunluk){
  int sayac;
  cout << pUzunluk << " Elemanlı Dizi:" << endl;
  for (sayac=0; sayac < pUzunluk; sayac++)
```

```
     cout << setw(5) << pDizi[sayac] ;
  cout << endl;
}
float diziOrtalama(int pDizi[],int pUzunluk){
  int sayac;
  float toplam=0;
  for (sayac=0; sayac < pUzunluk; sayac++)
     toplam+=pDizi[sayac];
  return toplam/pUzunluk;
}
/* Program Çıktısı:
5 Elemanlı Dizi Okunacaktır:10 20 30 40 50
5 Elemanlı Dizi:
   10   20   30   40   50
Dizi Ortalaması:30

...Program finished with exit code 0
*/
```

Foreach Talimatı

Yalnızca bir dizideki öğeler arasında döngü oluşturmak için kullanılan `foreach` döngüsü (foreach loop) aralık tabanlı döngü (range based loop) olarak da bilinir. Bu döngü ileride göreceğimiz *Konteyner Şablonları* başlığındaki veri yapılarıyla da kullanılır. Aşağıdaki gibi yazılır;

```
for (veritipi dizi-elemanını-temsil-eden-değişken : dizi-kimliği)
       döngü-kodu;
```

Buna örnek olarak aşağıda bir dizi programı örneği verilebilir;

```
float dizi[10] = {10.0, 20.2, 30.3, 40.4, 50.5, 60.6, 70.7, 80.8, 90.9, 100.0};

for (float f : dizi) {
  cout << c << ", ";
}
```