ITU ACM Student Chapter Course Program

Introduction to C

Week 4

Instructor

Mehmet Yiğit Balık

Prepared by

Mehmet Yiğit Balık & Mihriban Nur Koçak & Emir Oğuz

İki Boyutlu Diziler

C dilinde iki ve daha fazla boyuta sahip diziler tanımlanabilir. Bir önceki derste işlenmiş diziler tek boyutlu dizilerdir. İki boyutlu diziler satır ve sütunlardan oluşmuş diziler olarak tanımlanabilir. Burada satırlar boyutlardan birini, sütunlar bir diğerini temsil eder. Tek boyutlu dizilerde her bir elemanı temsilen tek bir sıra değeri vardır. İki boyutlu dizilerde ise her bir elemanı temsil etmek için iki sıra değeri (satır ve sütun) birlikte kullanılır.

	Column 1	Column 2	Column 3	Column 4
Row 1	x[0][0]	x[0][1]	x[0][2]	x[0][3]
Row 2	x[1][0]	x[1][1]	x[1][2]	x[1][3]
Row 3	x[2][0]	x[2][1]	x[2][2]	x[2][3]

İki boyutlu bir dizi şu şekilde tanımlanabilir:

double x[3][4];

İki boyutlu dizilere de ilk değer verilebilir. İki boyutlu bir dizinin elemanlarına süslü parantez içinde virgüllerle ayrılan kümeler kullanılarak ilk değer verilir.

```
#include <stdio.h>
#include <stdlib.h>
int main(){
    //buradaki 3 satırı temsil eder, 4 ise sütunu temsil eder.
    double x[3][4] ={{1,2,3,4},{5,6,7,8},{9,10,11,12}};
```

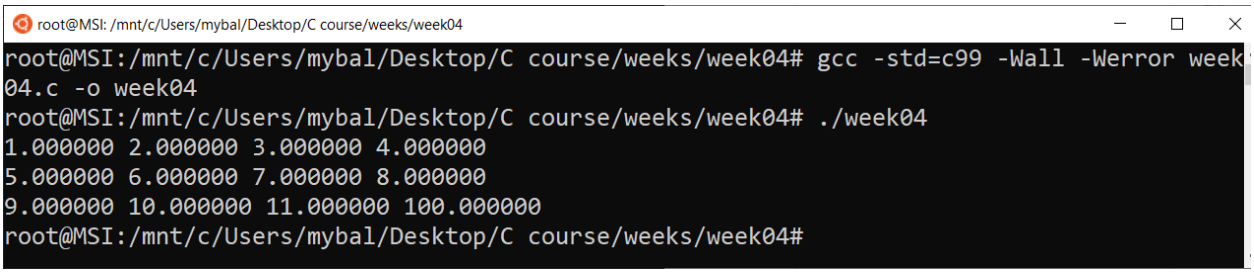
```c
    for(int i = 0;i < 3;i++){
        for(int j = 0;j < 4;j++){
            printf("%lf ",x[i][j]);
        }
        printf("\n");
    }
    return EXIT_SUCCESS;
}
```

```
root@MSI: /mnt/c/Users/mybal/Desktop/C course/weeks/week04
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# gcc -std=c99 -Wall -Werror week
04.c -o week04
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# ./week04
1.000000 2.000000 3.000000 4.000000
5.000000 6.000000 7.000000 8.000000
9.000000 10.000000 11.000000 12.000000
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04#
```

Tek boyutlu bir dizinin elemanlarını bastırabilmek için tek bir **for** döngüsü yeterliydi. Fakat iki boyutlu bir dizi için bu yeterli olmayacaktır. Çünkü bahsedildiği gibi iki boyutlu diziler satır ve sütunlardan oluşurlar. Yukarıdaki örnekte görüldüğü üzere iki boyutlu bir diziyi bastırabilmek için iç içe iki **for** döngüsü kullanılmıştır. Bu iç içe döngüde **i** değişkeni satırları temsil eder, **j** değişkeni ise **i**'nin temsil ettiği satırdaki sütunları temsil eder. Örneğin **i** değişkeninin değeri 0 iken, döngü 0. satır için çalışmaya başlar. Bu sırada **j** değişkeni de 0'dan 3'e kadar olan sütunları tek tek gezer. Bu işlem sona erdiğinde **i** değişkeninin değeri bir artar ve **j** değişkeni tekrar 0'dan 3'e kadar bu sefer 1. satır için çalışır.

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

int main(){
    double x[3][4] ={{1,2,3,4},{5,6,7,8},{9,10,11,12}};
    x[2][3] = 100;
    for(int i = 0;i < 3;i++){
        for(int j = 0;j < 4;j++){
            printf("%lf ",x[i][j]);
        }
        printf("\n");
    }
    return EXIT_SUCCESS;
}
```



Örnekte de görüldüğü üzere her dizide olduğu gibi iki boyutlu dizilerde de değiştirilecek elemanın sıra değeri belirtilerek bu eleman değiştirilebilir.

İki Boyutlu Dizilerin Fonksiyonlarla Kullanımı

Tek boyutlu dizilerde olduğu gibi iki boyutlu diziler de fonksiyonlarla beraber kullanılabilir. Burada en çok dikkat edilmesi gereken nokta iki boyutlu dizinin sütun büyüklüğüdür. Sütun büyüklüğü sabit bir sayı olarak belirlenmelidir ve dizinin belirtildiği her yerde sütun değeri de kendisine ait köşeli parantez içerisinde belirtilmelidir. Bunu yaparken sütun değerini macro olarak belirlemek tavsiye edilen bir yöntemdir. Sütun değeri macro olarak belirlendikten sonra artık değiştirilemez bir değerdir ve tüm fonksiyonlar tarafından kullanılabilir. Bu durumda iki boyutlu diziyi fonksiyona gönderirken beraberinde sadece satır değeri gönderilir.

```c
#include <stdio.h>
#include <stdlib.h>
#define SUTUN 5

void deger_ara(int satir,int dizi[][SUTUN],int aranan_deger){
    for(int i = 0;i < satir;i++){
        for(int j = 0;j < SUTUN;j++){
            if(dizi[i][j] == aranan_deger){
                printf("Dizide aradiginiz deger (%d) bulunmaktadir\n",aran
an_deger);
                return;
            }
        }
    }
    printf("Dizide aradiginiz deger (%d) bulunmamaktadir\n",aranan_deger);
}

int main(){
    int satir = 5;
    int dizi[][SUTUN] = {{1,2,3,4,5},
```

```
                        {6,7,8,9,10},
                        {11,12,13,14,15},
                        {16,17,18,19,20},
                        {21,22,23,24,25}};
    int aranan_deger;
    printf("Dizide aramak istediginiz degeri giriniz: ");
    scanf("%d",&aranan_deger);
    deger_ara(satir,dizi,aranan_deger);
    return EXIT_SUCCESS;
}
```

```
root@MSI: /mnt/c/Users/mybal/Desktop/C course/weeks/week04
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# gcc -std=c99 -Wall -Werror
04.c -o week04
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# ./week04
Dizide aramak istediginiz degeri giriniz: 12
Dizide aradiginiz deger (12) bulunmaktadir
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# ./week04
Dizide aramak istediginiz degeri giriniz: 100
Dizide aradiginiz deger (100) bulunmamaktadir
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# ./week04
Dizide aramak istediginiz degeri giriniz: 32
Dizide aradiginiz deger (32) bulunmamaktadir
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# ./week04
Dizide aramak istediginiz degeri giriniz: 20
Dizide aradiginiz deger (20) bulunmaktadir
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04#
```

Sık Kullanılan Dizi Algoritmaları

Dizilerle işlem yaparken sık kullanılan bazı algoritmalar vardır:

- Dizinin en küçük değeri bulma
- Dizinin en büyük değeri bulma
- Dizinin elemanlarını büyükten küçüğe veya küçükten büyüğe sıralama (bubble sorting)

Bir dizideki en büyük değeri bulmak için önce dizinin ilk elemanının değeri **max_deger** olarak belirlenir. Sonrasında dizinin her bir elemanı güncel **max_deger** ile karşılaştırılarak eğer karşılaştırılan eleman güncel **max_deger**'den büyükse, **max_deger** karşılaştırılan elemanın değerini alarak güncellenir. Bu işlem dizinin son elemanı da

max_deger ile karşılaştırıldıktan sonra sona erer. Sonuç olarak **max_deger** dizinin en büyük değerini almış olur.

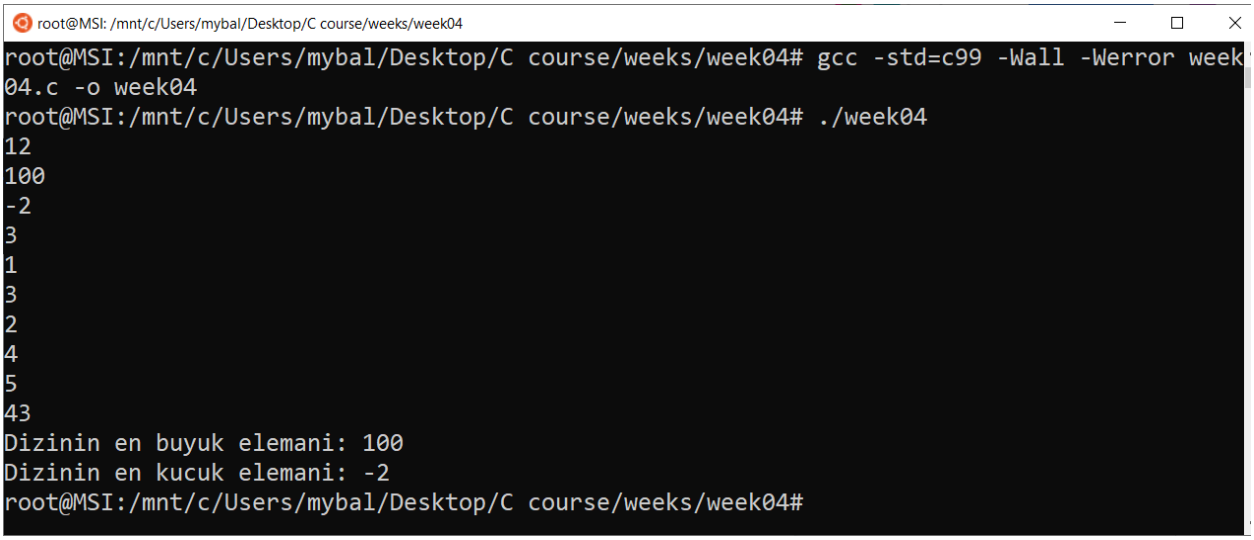
Bir dizideki en küçük değeri bulmak için önce dizinin ilk elemanının değeri **min_deger** olarak belirlenir. Sonrasında dizinin her bir elemanı güncel **min_deger** ile karşılaştırılarak eğer karşılaştırılan eleman güncel **min_deger'**den küçükse, **min_deger** karşılaştırılan elemanın değerini alarak güncellenir. Bu işlem dizinin son elemanı da **min_deger** ile karşılaştırıldıktan sonra sona erer. Sonuç olarak **min_deger** dizinin en küçük değerini almış olur.

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

int maksimum_deger(int dizi[], int boyut){
    //ilk adimda dizinin ilk elemani max deger olarak varsayilir
    int max_deger = dizi[0];
    //for dongusu kullanilarak en buyuk bulunur
    for(int i = 0;i < boyut;i++){
        if(max_deger < dizi[i]){
            max_deger = dizi[i];
        }
    }
    return max_deger;
}

int minumum_deger(int dizi[],int boyut){
    int min_deger = dizi[0];
    for(int i = 0;i < boyut;i++){
        if(dizi[i] < min_deger){
            min_deger = dizi[i];
        }
    }
    return min_deger;
}

int main(){
    int size = 10;
    int arr[size];
    for(int i = 0;i < size;i++){
        scanf("%d",&arr[i]);
    }
    int max_deger = maksimum_deger(arr,size);
    int min_deger = minumum_deger(arr,size);
    printf("Dizinin en buyuk elemani: %d\n",max_deger);
    printf("Dizinin en kucuk elemani: %d\n",min_deger);
    return EXIT_SUCCESS;
}
```



Örnekte bir diziyi elemanlarının değerinin büyüklüğüne göre sıralamak için kullanılan algoritmanın adı **Bubble Sort** algoritmasıdır. Bubble sort, en basit sıralama algoritmalarından biridir. Karşılaştırma temelli olan bu algoritmada, listedeki her bir eleman yanındaki eleman ile karşılaştırılır. Eğer ilk elemanın değeri, ikinci elemanın değerinden büyükse, iki eleman yer değiştirir. Daha sonra ikinci ve üçüncü elemanların değerleri karşılaştırılır. İkinci elemanın değeri üçüncü elemanın değerinden büyükse bu iki eleman yer değiştirir, küçükse yer değiştirmezler ve bu işlem, tüm liste sıralanana kadar bu şekilde devam eder. Birbirleriyle karşılaştırılan değerlerin yer değiştirmesi gereken durumlarda bu iki değeri de kaybetmemek adına, değerlerden birini saklayacak geçici bir değişken tanımlanır. Bu değişkene iki değerden biri atanır. Örneğin **deger1** ve **deger2** isimli karşılaştırma sonucu yer değiştirmesi gereken iki değişken olsun. Bu durumda yeni bir **gecici_degisken** tanımlansın. Önce **gecici_degisken**'e değer olarak **deger1** atansın sonrasında **deger1**'e **deger2** atansın ve en sonunda **deger2**'ye **gecici_degisken** atansın. Tüm bu işlemler yapıldığında **deger1** ve **deger2**'nin değerlerinin değişmiş olduğu görülür.

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

void kucukten_buyuge(int arr[], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
```

```
    {
        for (j = 0; j < n - i - 1; j++)
        {
            if (arr[j] > arr[j + 1])
            {
                int gecici_degisken = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = gecici_degisken;
            }
        }
    }
}

void buyukten_kucuge(int arr[], int n)
{
    int i, j;
    for (i = 0; i < n - 1; i++)
    {
        for (j = 0; j < n - i - 1; j++)
        {
            if (arr[j] < arr[j + 1])
            {
                int gecici_degisken = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = gecici_degisken;
            }
        }
    }
}

void printArray(int arr[], int size)
{
    int i;
    for (i = 0; i < size; i++)
        printf("%d ", arr[i]);
    printf("\n");
}

int main()
{
    int buyukluk = 10;
    int arr[buyukluk];
    for (int i = 0; i < buyukluk; i++)
    {
```

```
        printf("Lutfen %d. elemani giriniz: ", i);
        scanf("%d", &arr[i]);
    }
    kucukten_buyuge(arr, buyukluk);
    printf("Kucukten buyuge siralanmis: \n");
    printArray(arr, buyukluk);

    buyukten_kucuge(arr, buyukluk);
    printf("Buyukten kucuge siralanmis: \n");
    printArray(arr, buyukluk);

    return EXIT_SUCCESS;
}
```

```
root@MSI: /mnt/c/Users/mybal/Desktop/C course/weeks/week04
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# gcc -std=c99 -Wall -Werror week
04.c -o week04
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# ./week04
Lutfen 0. elemani giriniz: -90
Lutfen 1. elemani giriniz: 100
Lutfen 2. elemani giriniz: 50
Lutfen 3. elemani giriniz: 2
Lutfen 4. elemani giriniz: 1
Lutfen 5. elemani giriniz: 0
Lutfen 6. elemani giriniz: -2
Lutfen 7. elemani giriniz: -1
Lutfen 8. elemani giriniz: 32
Lutfen 9. elemani giriniz: -87
Kucukten buyuge siralanmis:
-90 -87 -2 -1 0 1 2 32 50 100
Buyukten kucuge siralanmis:
100 50 32 2 1 0 -1 -2 -87 -90
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04#
```

Karakter Dizilerine (Strings) Giriş

String'ler esasında karakterlerden yani harflerden oluşan dizilerdir örneğin *merhaba* 7 karakterden oluşan bir kelimedir.

```
char c[] = "merhaba";
```

Bu c char dizisi aslında 7 değil 8 karakterden oluşur bu görünmeyen 8. karakter ***\0 (null terminator)*** karakteridir. Bu karakter kelimenin yani char dizisinin sonuna geldiğini

belirtir. Eğer bu karakter dizinin sonuna konulmaz ise bilgisayarda güvenlik sıkıntılarIna yol açabilir. Bu konu ilerleyen haftalarda işlenecektir.

Özet olarak;

```
char c[] = "merhaba";
```

ve

```
char c[8] = {'m', 'e', 'r', 'h', 'a', 'b' ,'a', '\0'};
```

aynı tanımlamalardır.

String'leri print etmek için kullanılan formatlama yöntemi **%s** şeklindedir ve **\0** karakterine kadar olan bütün karakterleri printler. Bu **for** döngüsü ile de yapılabilir.

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

int main(){
    char c[] = "merhaba";
    char k[8] = {'m','e','r','h','a','b','a','\0'};
    printf("%s\n",c);
    printf("%s\n",k);
    // for dongusu ile printleme
    for(int i = 0;c[i] != '\0';i++){
        printf("%c",c[i]);
    }
    printf("\n");
    return EXIT_SUCCESS;
}
```

```
root@MSI: /mnt/c/Users/mybal/Desktop/C course/weeks/week04
merhabaroot@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# gcc -std=c99 -Wall -
or week04.c -o week04
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04# ./week04
merhaba
merhaba
merhaba
root@MSI:/mnt/c/Users/mybal/Desktop/C course/weeks/week04#
```