

# BSM101 Programlama Dilleri I

## Hafta 13 Fonksiyonlar

Dr. Öğr. Üyesi Caner ÖZCAN

# Dizilerin Fonksiyonlara Gönderilmesi

- Bir diziye bir fonksiyona parametre olarak göndermek için parantez kullanmadan sadece dizinin ismi belirtilir.
  - **int** myArray [ **24** ];
  - **myFunction** ( myArray, 24);
- Char dizilerinin aksine diğer türdeki diziler her hangi bir sonlandırma karakteri içermezler.
- Bu sebeple fonksiyonlara dizideki eleman sayısında parametre olarak gönderilir ki, fonksiyon uygun sayıda eleman üzerinde işlem yapsın.

# Dizilerin Fonksiyonlara Gönderilmesi

- Dizilerin fonksiyonlara gönderilmesi referans ile çağırma işlemidir (call by reference).
- Dizinin adı aslında ilk elemanının adresidir.
- Fonksiyon böylece dizinin ilk elemanının hafıza nerede olduğunu bilir.
  - Orijinal hafıza bölgesinde işlem yapılır.
- Dizideki her hangi bir elemanın fonksiyona gönderilmesi ise değer ile çağırmadır (call by value).
  - Fonksiyona herhangi bir indisteki elemanın değeri gönderilir
    - `myArray [3]`
- Bir int dizi ve bir int değeri parametre olarak alan fonksiyon prototipi;
  - `void myArray (int [ ], int )`

# Dizilerin Fonksiyonlara Gönderilmesi

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

//Function prototypes
void function1(int []);
void function2(int );

int main()
{
    int array[SIZE]={1,2,3,4,5}, i;
    printf("\nthe array before the funtions\n");
    for(int i=0;i<SIZE;i++)
        printf("%d ",array[i]);

    function1(array);
    function2(array[0]);

    printf("\nthe array after the functions\n");
    for(i=0;i<SIZE;i++)
        printf("%d ",array[i]);
    printf("\n\n");
    return 0;
}
```

# Dizilerin Fonksiyonlara Gönderilmesi

```
void function1 (int A[])
{
    int i;
    for(i=0;i<SIZE;i++)
        A[i]+=i;
}

void function2(int x)// will value of array[0] change and Why?
{
    x=50;
}
```

# Dizilerin Fonksiyonlara Gönderilmesi

```
void function1 (int A[])
{
    int i;
    for(i=0;i<SIZE;i++)
        A[i]+=i;
}

void function2(int x)// will value of array[0] change and Why?
{
    x=50;
}
```

the array before the funtion  
1 2 3 4 5  
the array after the function  
1 3 5 7 9

# Çok Boyutlu Dizileri Fonksiyonlara Gönderme

- Tek boyutlu dizileri fonksiyona göndermekten farklı değildir.
- Her bir boyut için köşeli parantez kullanın, ilk boyut hariç diğerleri için büyüklük belirtin.
  - `void writeMatrice (int [ ] [ 4 ], int rowNumber);`
  - Bu tanımlama 4 sütuna sahip her matris için farklı satır numaralarına sahip olsalar da geçerli olur.
  - `void writeMatrice (int [ ] [ 3 ] [ 4 ], int rowNumber);`

# Çok Boyutlu Dizileri Fonksiyonlara Gönderme

```
#include <stdio.h>

void printArray( int [][][3]);
void SumArray(int [][][3], int [][][3]);
int main()
{
    int array1[2][3]={1,2,3},{4,5,6};
    int array2[2][3]={1,2,3,4,5,6};

    printf("\nbefor SumArray function\n");
    printArray(array1);

    SumArray(array1,array2);

    printf("\nafter Sumarray function\n");
    printArray(array1);
    printf("\n\n");
    return 0;
}
```



# Çok Boyutlu Dizileri Fonksiyonlara Gönderme

```
void printArray(int arr[][3])
{
    int i, j;
    for(i=0;i<2;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("%d ",arr[i][j]);
        }
        printf("\n");
    }
}

void SumArray(int arr1[][3], int arr2[][3])
{
    int i, j;
    for(i=0;i<2;i++)
    {
        for(j=0;j<3;j++)
        {
            arr1[i][j]+=arr2[i][j];
        }
    }
}
```

# Çok Boyutlu Dizileri Fonksiyonlara Gönderme

```
void printArray(int arr[][3])
{
    int i, j;
    for(i=0;i<2;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("%d ",arr[i][j]);
        }
        printf("\n");
    }
}

void SumArray(int arr1[][3], int arr2[][3])
{
    int i, j;
    for(i=0;i<2;i++)
    {
        for(j=0;j<3;j++)
        {
            arr1[i][j]+=arr2[i][j];
        }
    }
}
```

before SumArray function

1 2 3  
4 5 6

after Sumarray function

2 4 6  
8 10 12

# Örnek 1

```
1  #include <stdio.h>
2  void function(char, char);
3  int main()
4  {
5      char i='A';
6      function(i++, ++i);
7      return 0;
8  }
9  void function(char x, char y)
10 {
11     char tmp;
12     tmp=x;
13     x=y;
14     y=tmp;
15     printf("%c %c", x, y);
16 }
```

# Örnek 1

```
1  #include <stdio.h>
2  void function(char, char);
3  int main()
4  {
5      char i='A';
6      function(i++, ++i);
7      return 0;
8  }
9  void function(char x, char y)
10 {
11     char tmp;
12     tmp=x;
13     x=y;
14     y=tmp;
15     printf("%c %c", x, y);
16 }
```

C	A
---	---

# Örnek 2

```
1  #include <stdio.h>
2
3  void MatrisToArray(int aa[][5], int a[]);
4  void incremant(int x);
5  int decremant(int y);
6
7  int main()
8  {
9      int array[10], matris[2][5]={1,2,3,4,5,6,7,8,9,10}, i;
10
11     MatrisToArray(matris, array);
12
13     for(i=0;i<10;i++)
14     |     printf("%d ",array[i]);
15
16     return 0;
17 }
```

# Örnek 2

```
19 void MatrisToArray(int aa[][5], int a[])
20 {
21     int i, j;
22     for(i=0;i<2;i++)
23         for(j=0;j<5;j++)
24             a[i*5+j]=aa[i][j];
25
26     for(i=0;i<10;i++)
27         incremant(a[i]);
28
29     for(i=0;i<10;i++)
30         a[i]=decremant(a[i]);
31 }
32 void incremant(int x)
33 {
34     x+=10;
35 }
36 int decremant(int y)
37 {
38     return y-10;
39 }
```

# Örnek 2

```
19 void MatrisToArray(int aa[][5], int a[])
20 {
21     int i, j;
22     for(i=0;i<2;i++)
23         for(j=0;j<5;j++)
24             a[i*5+j]=aa[i][j];
25
26     for(i=0;i<10;i++)
27         incremant(a[i]);
28
29     for(i=0;i<10;i++)
30         a[i]=decremant(a[i]);
31 }
32 void incremant(int x)
33 {
34     x+=10;
35 }
36 int decremant(int y)
37 {
38     return y-10;
39 }
```

-9	-8	-7	-6	-5	-4	-3	-2	-1	0
----	----	----	----	----	----	----	----	----	---

# Örnek 3

```
1  /* Simplifies consecutive elements in array 5552211221 */
2
3  #include <stdio.h>
4
5  void shifting(char[],int);
6  void simplifying(char []);
7
8  int main()
9  {
10     int i=0;
11     char array[]="5552211221";
12     simplifying(array);
13     printf("\n\n");
14     puts(array);
15     printf("\n\n");
16     return 0;
17 }
18
```



# Örnek 3

```
19 void simplifying(char array[])
20 {
21     int i;
22     while(array[i]!='\0')
23     {
24         if(array[i]==array[i+1])
25             shifting(array,i);
26         else
27             i++;
28     }
29 }
30
31 void shifting(char a[],int i)
32 {
33     for(;a[i]!='\0';i++)
34         a[i]=a[i+1];
35 }
```

# Örnek 4

```
1  #include <stdio.h>
2
3  int sum(int array[], int size)
4  {
5      int total=0, i;
6      for(i=0;i<size;i++)
7          total+=array[i];
8      return total;
9  }
10
11 void multiply(int array[], int x)
12 {
13     int i;
14     for(i=0;i<5;i++)
15         array[i]*=x;
16 }
17
```

```
18 void print(int array[], int size)
19 {
20     int i;
21     for(i=0;i<size;i++)
22         printf("%d ",array[i]);
23 }
24
25 int main()
26 {
27     int array[5]={1,2,3,4,5};
28
29     multiply(array, sum(array, 5));
30
31     print(array, 5);
32
33     return 0;
34 }
```

# Örnek 4

```
1  #include <stdio.h>
2
3  int sum(int array[], int size)
4  {
5      int total=0, i;
6      for(i=0;i<size;i++)
7          total+=array[i];
8      return total;
9  }
10
11 void multiply(int array[], int x)
12 {
13     int i;
14     for(i=0;i<5;i++)
15         array[i]*=x;
16 }
17
```

```
18 void print(int array[], int size)
19 {
20     int i;
21     for(i=0;i<size;i++)
22         printf("%d ",array[i]);
23 }
24
25 int main()
26 {
27     int array[5]={1,2,3,4,5};
28
29     multiply(array, sum(array, 5));
30
31     print(array, 5);
32
33     return 0;
34 }
```

15 30 45 60 75

# Örnek 5

```
1  #include <stdio.h>
2  float function(void);
3  float calculate_area(float);
4  int main()
5  {
6      float area;
7      area=function();
8      printf("%f",area);
9      return 0;
10 }
11 float function(void)
12 {
13     float r;
14     printf("please enter the radius");
15     scanf("%f",&r);
16     return calculate_area(r);
17 }
18
19 float calculate_area(float r)
20 {
21     float x, pi=3.14;
22     x=pi*r*r;
23     return x;
24 }
```

# Örnek 6

```
1  #include <stdio.h>
2
3  int Anagram(char str1[], char str2[]);
4
5  int main()
6  {
7      char str1[100], str2[100];
8      printf("\n\n Function : whether two given strings are anagram :\n");
9      //Example : pears and spare, stone and tones, listen and slient
10
11     printf(" Input the first String : ");
12     scanf("%s",str1);
13     printf(" Input the second String : ");
14     scanf("%s",str2);
15
16     if(Anagram(str1, str2) == 1)
17     |     printf(" %s and %s are Anagram.\n\n",str1,str2);
18     |
19     else
20     |     printf(" %s and %s are not Anagram.\n\n",str1,str2);
21     |
22     return 0;
23 }
24
```

# Örnek 6

```
25 //Function to check whether two passed strings are anagram or not
26 int Anagram(char str1[], char str2[])
27 {
28     int str1ChrCtr[256] = {0}, str2ChrCtr[256] = {0};
29     int ctr,a,b;
30
31     /* check the length of equality of Two Strings */
32     for(a=0;str1[a]!='\0';a++);
33     for(b=0;str2[b]!='\0';b++);
34     if(a!=b)
35         return 0;
36
37     //count frequency of characters in str1
38     for(ctr = 0; str1[ctr] != '\0'; ctr++)
39         str1ChrCtr[str1[ctr]]++;
40
41     //count frequency of characters in str2
42     for(ctr = 0; str2[ctr] != '\0'; ctr++)
43         str2ChrCtr[str2[ctr]]++;
44
45     //compare character counts of both strings
46     for(ctr = 0; ctr < 256; ctr++)
47         if(str1ChrCtr[ctr] != str2ChrCtr[ctr])
48             return 0;
49
50     return 1;
51 }
```

# QUIZ

Klavyeden Integer tipinde tek boyutlu bir dizi alan ve dizide kaç tane 4'ün katı olan sayı olduğunu döndüren fonksiyonu ve ana programı yazın.

# 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