

CME 112- Programming Languages II

Week 6

Examples with Pointers

Assist. Prof. Dr. Caner Özcan

A person who is aware of can only go on his journey to self-discovery.

~D.CUCELOGLU

• Malloc() Function

- ▶ Malloc function is used to allocate a block of memory for one variable.
- ▶ If there is not enough memory available, malloc will return NULL.

```
int *ptr;  
ptr = (int *) malloc(n*sizeof(int));
```



Calloc() Function

- ▶ Calloc function is also used to allocate a block of memory.
- ▶ If there is not enough memory available, calloc will return NULL.
- ▶ Unlike malloc function, it takes two arguments.

```
char *ptr;
```

```
ptr = (char *)calloc(10, sizeof(char));
```

Realloc() Function

- ▶ Realloc is used to resize an allocated memory space.
- ▶ A pointer that will point the starting address of resized memory space and new size are passed to realloc function as parameter.

```
void *realloc(void *ptr, size_t size);
```



Free() Function

- ▶ How important an effective memory management is may be understood when we write large programs.
- ▶ We should avoid consuming unnecessary memory.
- ▶ Every call to an malloc or calloc function you must have a corresponding call to free.

```
int *ptr;  
ptr = (int *) malloc(n*sizeof(int));  
free(ptr);
```



Dynamic Memory Allocation & Arrays 6

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 int main( void )
4 {
5     // Dinamik bir dizi oluşturmak için pointer kullanırız.
6     int *dizi;
7     // Dizimizin kaç elemanlı olacağını eleman_sayisi isimli degiskende tutuyoruz.
8     int eleman_sayisi;
9     int i;
10    printf( "Eleman sayısını giriniz> ");
11    scanf( "%d", &eleman_sayisi );
12    // malloc( ) fonksiyonuyla dinamik olarak dizimizi istedigimiz boyutta oluşturalım.
13    dizi = (int *)malloc( eleman_sayisi * sizeof( int ) );
14    //dizi = (int *)calloc( eleman_sayisi, sizeof( int ) );
15
16    for( i = 0; i < eleman_sayisi; i++ )
17        printf( "Adres:%d\tDeger:%d\n", &dizi[i],dizi[i] );
18
19    // hafizadan temizleme
20    free( dizi );
21
22    while( getchar() != '\n' ) { /*do nothing*/ } ;
23        getchar() ; /* wait */
24    return 0;
25 }
```

Dynamic Memory Allocation & Arrays 7

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 int main( void )
4 {
5     // Dinamik bir dizi oluşturmak için pointer kullanırız.
6     int *dizi;
7     // Dizimizin kaç elemanlı olacağını eleman_sayisi isimli degiskende tutuyoruz.
8     int eleman_sayisi;
9     int i;
10    printf( "Eleman sayısını giriniz> ");
11    scanf( "%d", &eleman_sayisi );
12    // malloc( ) fonksiyonuyla dinamik olarak dizimizi istedigimiz boyutta oluşturalım.
13    dizi = (int *)malloc( eleman_sayisi * sizeof( int ) );
14    //dizi = (int *)calloc( eleman_sayisi, sizeof( int ) );
15
16    for( i = 0; i < eleman_sayisi; i++)
17        printf( "Adres:%d\tDeger:%d\n", dizi[i], dizi[i] );
18
19    // hafizadan temizleme
20    free( dizi );
21
22    while( getchar() != '\n' )
23        getchar(); /* wait for user to press enter
24    return 0;
25 }
```

D:\Akademik\2016\Programlama Dilleri 2\Prog2_Le

```
Eleman sayisini giriniz> 5
Adres:201968 Deger:209360
Adres:201972 Deger:196800
Adres:201976 Deger:1836008284
Adres:201980 Deger:929984365
Adres:201984 Deger:1869567068
```

Function Pointers

- ▶ Pointers can show the address of the function held.
- ▶ A function name is really the starting address in memory of the code that performs the function's task.

int (*fPtr) (int,int)

In this definition, fPtr shows the address of a function that takes two integer parameters and returns an integer value.

int *fPtr (int,int)

In this definition, a function named fPtr is defined that takes two integer parameters and returns an integer pointer.



Function Pointers

```
1  #include <stdio.h>
2  int kare(int);
3  int kup(int);
4  int main(void)
5  {
6      /* bir int deęer alıp geriye int deęer gönderen bir fonksiyonun adresi */
7      int (*islem)(int);
8      int i;
9      char c;
10
11     printf("1-kare alani\n2-kup hacmi\n ");
12     c = getchar();
13     printf("\nSayıyı gir : ");
14     scanf("%d", &i);
15     if (c == '1')
16         islem = kare; /* kare işlevinin adresi islem deęişkenine kopyalanır */
17     else
18         islem = kup;
19     printf("Sonuç = %d\n", islem(i));
20     while( getchar() != '\n' ) { /*do nothing*/ } ;
21     getchar() ; /* wait */
22 }
23 int kare(int s)
24 {
25     return s*s;
26 }
27 int kup(int s)
28 {
29     return s*s*s;
30 }
```

Function Pointers

```

1  #include <stdio.h>
2  int kare(int);
3  int kup(int);
4  int main(void)
5  {
6      /* bir int deęer alıp geriye int deęer gönderen bir fonksiyonun adresi */
7      int (*islem)(int);
8      int i;
9      char c;
10
11     printf("1-karealani\n2-kup hacmi\n ");
12     c = getchar();
13     printf("\nSayıyı gir : ");
14     scanf("%d", &i);
15     if (c == '1')
16         islem = kare; /* kare işlevinin adresi islem deęişkenine
17     else
18         islem = kup;
19     printf("Sonuç = %d\n", islem(i));
20     while( getchar() != '\n' ) { /*do nothing*/ } ;
21     getchar() ; /* wait */
22 }
23 int kare(int s)
24 {
25     return s*s;
26 }
27 int kup(int s)
28 {
29     return s*s*s;
30 }

```

D:\Akademik\2016\Program

```

1-karealani
2-kup hacmi
2
Sayıyı gir : 5
Sonuç = 125

```

D:\Akademik\2016\Progra

```

1-karealani
2-kup hacmi
1
Sayıyı gir : 3
Sonuç = 9

```

Void Pointers

- ▶ Pointers can be defined in void type.
- ▶ We have to specify the type of data for accessing the data that void pointer show.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 int main(void)
5 {
6     void *a;
7     a = (char*) malloc(6);
8     strcpy((char *)a, "12345");
9     printf("%s\n", a);
10    free(a);
11    a = (double*) malloc(sizeof(double));
12    /* deęere eriřirken veri tipi belirt */
13    *(double*)a = 3.123;
14    printf("%f\n", *(double *)a);
15    getchar();
16 }
```

Example 1

- ▶ Write a program that continuously takes a character unless user press ENTER and prints "*" for each character entered from keyboard.
- ▶ When user presses ENTER the program will write all the characters entered since the beginning of data entrance in input order. Character code for "ENTER" is 13.

Example 1

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

int main(){
    char giris[50];
    char *p;
    int i=0,k;
    p=giris;
    while(1){
        *(p+i)=getch();
        if(*(p+i)==13)
            break;
        putchar('*');
        printf("Adres[%d]: %d\n",i, p+i);
        i++;
    }
    printf("\n");
    for(k=0;k<i;k++){
        printf("Adres[%d]: %d\n",k, p+k);
        putchar(*(p+k));
    }
    getch();
    return 0;
}
```

Example 1

```
#include <stdio.h>
#include <stdlib.h>
void main()
{
    char *p;
    int i=0, k;

    p = (char *) malloc(sizeof(char));
    while(1)
    {
        *(p+i) = getch();
        if(*(p+i) == 13) break;
        putchar('*');
        i++;
        p = (char *) realloc(p, (i+1)*sizeof(char));
    }

    putchar('\n');
    for(k=0;k<i;k++)
        putchar(*(p+k));
}
```

Example 2

- ▶ Write a function with prototype given below which interchanges two variables values.

```
void swap (int *, int *)
```

Example 2

- ▶ Write a function which interchanges two variables values.

```
#include <stdio.h>
void swap(int * q,int * p)
{
    int temp = *p;
    *p = *q;
    *q = temp;
}

int main()
{
    int a = 10, b = 2, x = 3, y = 5;
    printf("a,b,x,y: %d,%d,%d,%d\n", a, b, x, y);
    swap(&x, &y);
    swap(&a, &b);
    printf("a,b,x,y: %d,%d,%d,%d\n", a, b, x, y);
}
```


Example 3

- ▶ Write a function with prototype given below which calculates the area and perimeter of a rectangle.

```
void rectangle(int a,int b, int *area, int *perimeter)
```

Example 3

- ▶ Write a function which calculates the area and perimeter of a rectangle.

```
#include <stdio.h>
void dortgen(int a, int b, int *alan, int *cevresi);

int main()
{
    int x, y;
    int alan, cevresi;
    printf("Boşlukla ayrılmış iki değer giriniz: " );
    scanf("%d %d", &x, &y);
    dortgen(x, y, &alan, &cevresi);
    printf("Alanı %d ve çevresi %d dir\n", alan, cevresi);
}

void dortgen(int a,int b, int *alan,int *cevresi)
{
    *alan = a * b;
    *cevresi = 2 * (a + b);
}
```

Example 4

- ▶ Write a function that performs like strlen function. Prototype for this function is as given below:

int uzunluk(char *)

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

int uzunluk(char *);

int main()
{
    char str[100];
    printf("Enter string");
    gets(str);
    printf("Length of string : %d", uzunluk(str));
    getch();
}
```

Example 4

- ▶ Write a function that performs like strlen function. Prototype for this function is as given below:

```
int uzunluk(char *)
```

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

int uzunluk(char *);

int main()
{
    char str[100];
    printf("Enter string");
    gets(str);
    printf("Length of string : %d", uzunluk(str));
    getch();
}
```

```
int uzunluk(char * p)
{
    int n =0;

    while(*p != '\0')
    {
        n++;
        p++;
    }
    return n;
}
```

Example 5

- ▶ Write a function that performs search a character in a given string. Prototype for this function is as given below:

```
char * ara (char *, char)    #include<stdio.h>
                             #include<conio.h>

char *ara(char *, char);

int main() {
    char *sonuc;
    char aranan;
    char str[100];
    printf("Enter string:\n");
    gets(str);
    printf("Enter character to search:\n");
    aranan = getchar();
    printf("Aranan: %c\n", aranan);
    printf("Aranan: %d\n", aranan);
    sonuc = ara(str, aranan);
    if(sonuc==NULL)
        printf("Character not found\n");
    else
        printf("Character found\n");
    getch();
}
```

Example 5

- Write a function that performs search a character in a given string. Prototype for this function is as given below:

```
char * ara (char *, char)    #include<stdio.h>
                             #include<conio.h>
```

```
char *ara(char *, char);
```

```
char *ara(char *p, char c){
    while(*p != '\0'){
        if(*p==c)
            return p;
        p++;
    }
    if(c=='\0')
        return p;
    return NULL;
}
```

```
int main() {
    char *sonuc;
    char aranan;
    char str[100];
    printf("Enter string:\n");
    gets(str);
    printf("Enter character to search:\n");
    aranan = getchar();
    printf("Aranan: %c\n", aranan);
    printf("Aranan: %d\n", aranan);
    sonuc = ara(str, aranan);
    if(sonuc==NULL)
        printf("Character not found\n");
    else
        printf("Character found\n");
    getch();
}
```

Example 5

- Write a function that performs search a character in a given string.

```
char * ara (char *, char) #include<stdio.h>
                          C:\Use<conio.h>
```

```
Enter string:
Today weather is good.
Enter character to search:
i
Aranan: i
Aranan: 105
Character found
```

```
while(*p != '\0'){
    if(*p==c)
        return p;
    p++;
}
if(c=='\0')
    return p;
return NULL;
}
```

```
ara(char *, char);
n() {
    r *sonuc;
    r aranan;
    r str[100];
    printf("Enter string:\n");
```

```
gets(str);
```

```
print
```

```
arana
```

```
printEnter string:
```

```
printtoday weather is good.
```

```
sonucEnter character to search:
```

```
if(so
```

```
Aranan: g
```

```
Aranan: 113
```

```
else Character not found
```

```
getch(
```

```
}
```

Example 6

- ▶ Write a function that converts an unsigned integer to binary.

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

void convertToBinary(unsigned);

int main()
{
    unsigned sayi;
    printf("Enter number");
    scanf("%u", &sayi);
    convertToBinary(sayi);

    getch();
}
```


Example 6

- ▶ Write a function that converts an unsigned integer to binary.

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

void convertToBinary(unsigned);

int main()
{
    unsigned sayi;
    printf("Enter number");
    scanf("%u", &sayi);
    convertToBinary(sayi);

    getch();
}
```

```
void convertToBinary (unsigned x)
{
    int i=1, k;
    unsigned *p;
    p = &x;

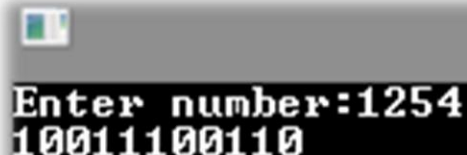
    while(1)
    {
        *(p+i) = x%2;
        x = x/2;

        if(x== 1)
        {
            *(p+i+1) = x;
            break;
        }
        i++;
    }
    for(k=i+1;k>0;k--)
        printf("%u", *(p+k));
}
```

Example 6

```
void convertToBinary (unsigned x)
{
    int i=0,k;
    unsigned *p;
    p = (unsigned *)malloc(sizeof(unsigned));
    *p = x;
    while(1)
    {
        *(p+i) = x % 2;
        x = x/2;

        if(x == 1)
        {
            p = (unsigned *) realloc(p, (i+1)*sizeof(unsigned));
            *(p+i+1) = 1;
            break;
        }
        i++;
        p = (unsigned *) realloc(p, (i+1)*sizeof(unsigned));
    }
    for(k=i+1;k>=0;k--)
        printf("%d", *(p+k));
}
```



Enter number:1254
10011100110

Questions

1)

```

  1
 1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1

```

Pascal Üçgeninin ilk 6 satırı yanda verilmiştir. Kullanıcının girdiği N değeri için pascal üçgeninin ilk N satırını alt alta ekrana yazdıran, algoritmayı çiziniz. Not: Çözümde sadece tek bir dizi kullanılmalıdır.

2) $\cos(X) = 1 - x^2/2! + x^4/4! + \dots + x^N/N!$ şeklinde tanımlanmıştır. Kullanıcının girdiği X ve N için $\cos(X)$ 'i hesaplayan algoritmayı çiziniz.

3) Bir sayı dizisinin ardışık elemanlarının arasındaki mutlak değerce en büyük farkı ve yerini bulan algoritmayı çiziniz.

4)

X		X		X	
	X		X		X
X		X		X	
	X		X		X
X		X		X	
	X		X		X

Kullanıcının girdiği N*N'lik bir matrisin sadece yandaki şekildeki çarpı işaretli hücrelerinin toplamını bulup ekrana yazdıran, algoritmayı çiziniz.

5) Bir dizideki çift sayılara başa, tekleri sona başka bir dizi kullanmadan atan algoritmayı çiziniz.

Örnek:

Giriş dizisi: 5 7 2 9 5 3 8 6

Çıkış dizisi: 2 8 6 3 5 9 7 5

Next Week

- ▶ Struct, Enum and Typedef
- ▶ Singly Linked Linear Lists

References

- ▶ Doç. Dr. Fahri Vatansever, “Algoritma Geliştirme ve Programlamaya Giriş”, Seçkin Yayıncılık, 12. Baskı, 2015.
- ▶ Kaan Aslan, “A’dan Z’ye C Klavuzu 8. Basım”, Pusula Yayıncılık, 2002.
- ▶ Paul J. Deitel, “C How to Program”, Harvey Deitel.
- ▶ “A book on C”, All Kelley, İra Pohl

Q u e s t i o n s
A n y
?



Thanks for listening

CANER ÖZCAN



canerozcan@karabuk.edu.tr

“

Empty your memory with
a free() like a pointer, If you
cast a pointer to be a integer,
it becomes an integer. If you
cast a pointer to a struct, it
becomes a struct. The pointer
can crash...And can Overflow...
BE A POINTER MY FRIEND.

”

-Dennis Ritchie