CPE101 Programming Languages I

Week 12
Character Array (String)

Assoc. Prof. Dr. Caner ÖZCAN

String Definition

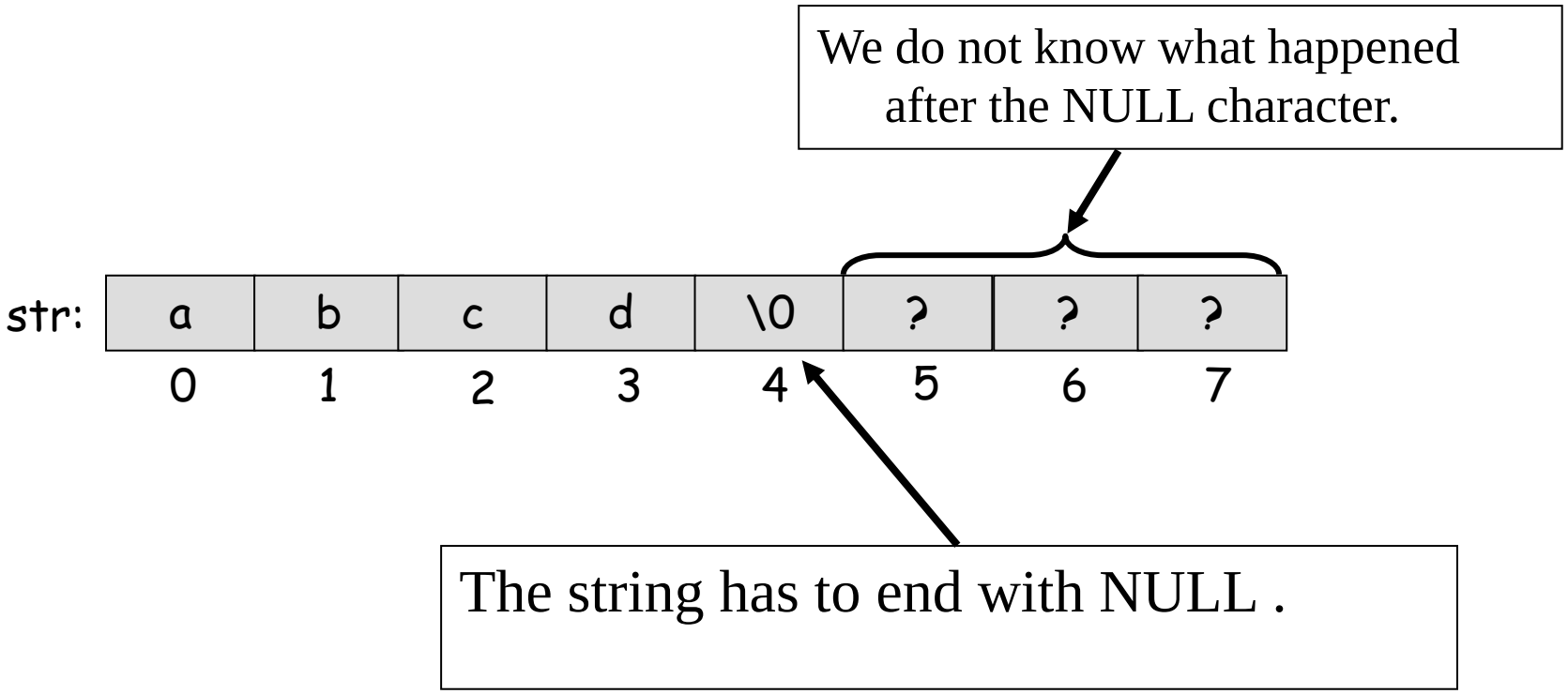
- We learned arrays and multidimensionalarrays.
- What we call String is actually an array.
- Arrays with variable type ‘char’ are called as ‘String’.
- For example, we store integers in an integer (int) array while we store chars (char) in a string.
- Names, addresses, usernames, phones, etc. ... We use strings for anything that can be verbally expressed.

String Definition

- String is a character array ending with NULL character ‘ \0 ’.
- Example: char str[8];
  - It creates an array that can take up to 8 characters.
  - If it is to be used as the string str may take up to 7 characters and array has to end with NULL character ‘ \0 ’.

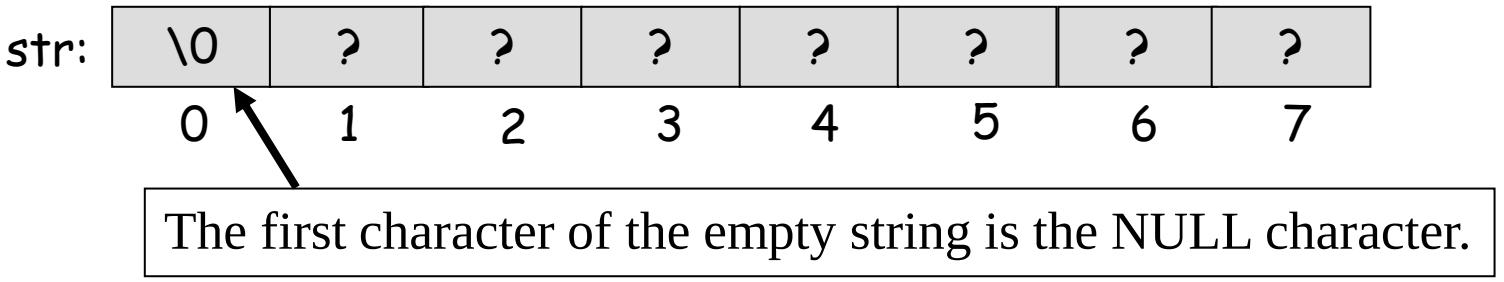
String Representation

- If we store “abcd” in str, it will appear in the following manner.



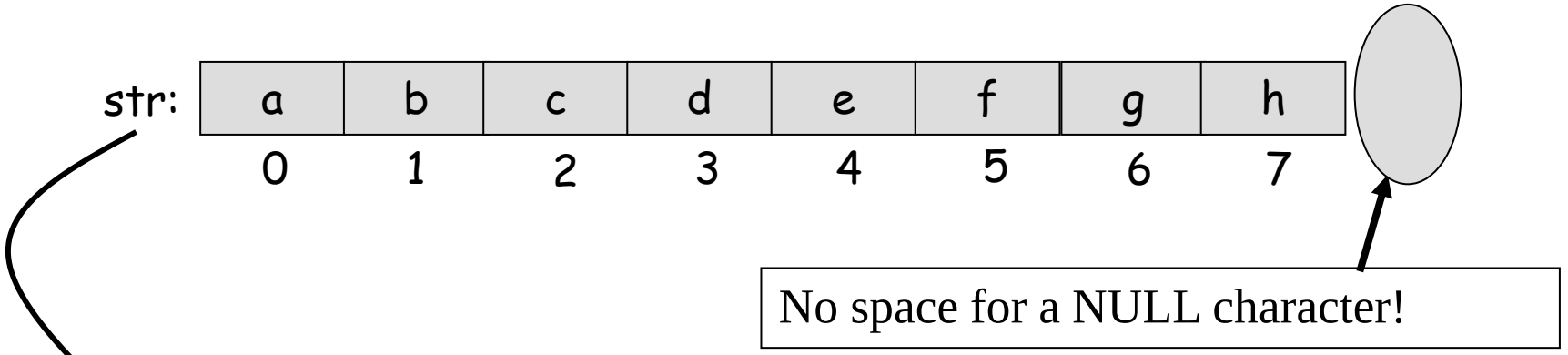
Empty String

- Empty string “” is refers to a character string in which the first element is null character ‘ \0 ’.



String Maximum Length

- A string of 8 characters, for example "abcdefgh" cannot be stored in str.



- This is a string with 8 characters.
- But NOT the string. String MUST always end up with a NULL character!

String: WARNING

- Just again, a statement like char str [8] simply emphasizes that we can store up to 8 characters in str.

- We may want to store more than 8 characters in str at any point during the execution of the program.

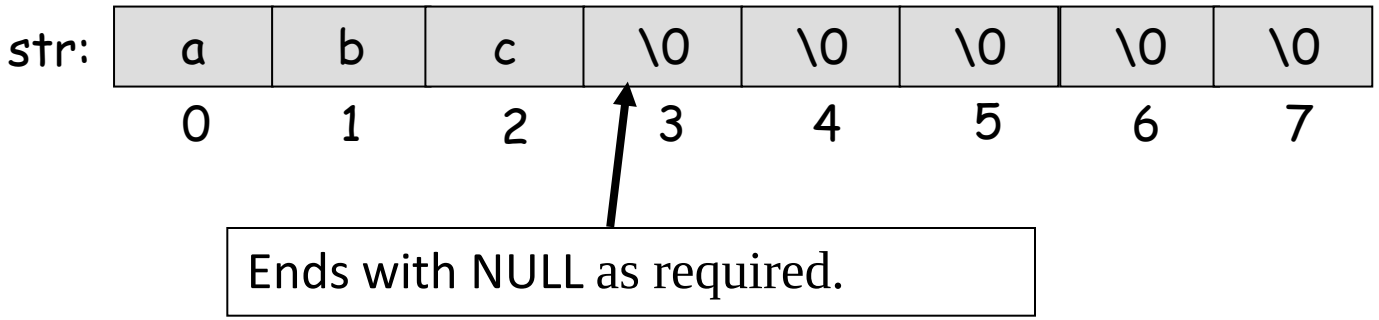
- But if "str" stores string, we store maximum 8-1 = 7 characters, and always has to end with a NULL.

String: Initialization

- An array of characters can be defined as follows during the initial value assignment.

```
char str[8] = {'a', 'b', 'c'};
```

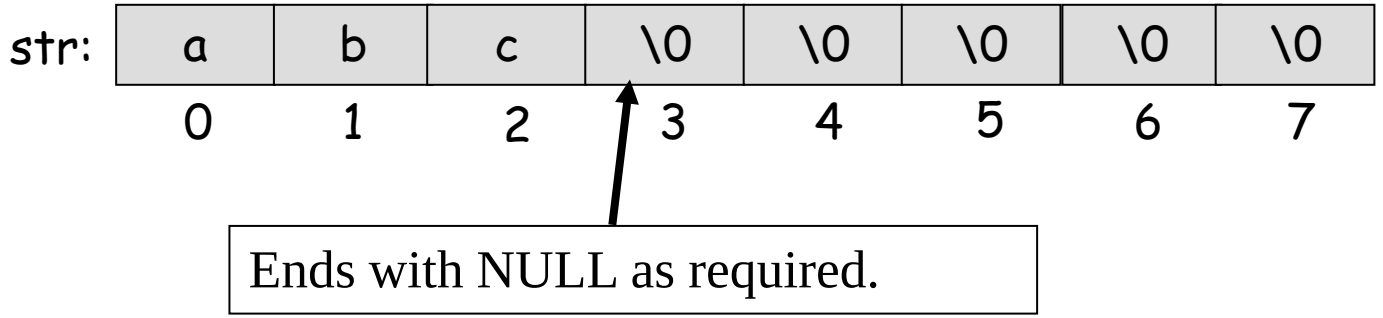
- Remember unspecified elements are filled with ' \0 ', NULL character that is happening.
  - Therefore, the above statement corresponds to the following string.



String: Initialization

- If a character array is to be stored in a string, the initial value can be simply assigned as follows.
  - Only string is placed inside double quotes. This is called a string literal.

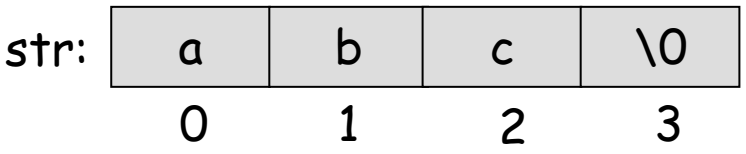
```
char str[8] = "abc"; /* same with previous */
```



String: Initialization

- If length of the array is not specified at the define time, so the compiler allocates string length + NULL characters.

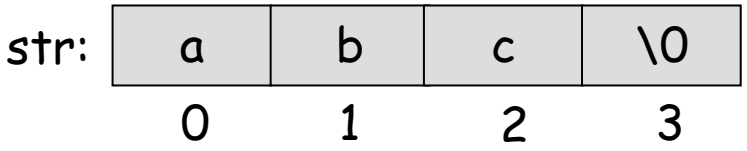
```
char str[] = "abc";
```



String: Initialization

- Strings are generally defined as follows.

```
char *str = "abc";
```



- The difference between the previous definition with this definition: strings defined with this way is being READ-ONLY and can not be change.

- char str[]="abc"; You can change the form of the strings defined as desired.

Writing a String

- C offers two functions to print the string s.
  - (1) puts(str); (2) printf("%s", str);

```
char str1[]="my first string";

/* print string and cursor go to newline.*/
puts(str1);

/* begin to print from where the string pointer is*/
printf("%s", str1);

/* Allocate 40 empty space and write string to in it based
   on the right. */
printf("%40s", str1);

/* Allocate 40 empty space and write string to in it on the
   left. */
printf("%-40s", str1);
```

Writing a String

```
char str1[]="my first string";

/* write first 10 character from string,
 * based on the right */
printf("%.10s", str1);

/* Allocate 40 space and prints just ten character,
    * based on the right */
printf("%40.10s", str1);

/* Allocate 40 space and prints just ten character,
    * based on the left */
printf("%-40.10s", str1);
```

Reading a String

- C offers two functions to get string from the keyboard.

  (1) gets(str); (2) scanf("%s", str);

```
char str2[80];

/* reads the entered string until you enter'\n' */
gets(str2);

/* all whitespace characters (space, tab, newline)
    * read entry until the next empty character. */
scanf("%s", str2);
```

Reading a String

```
char str2[80];

/* skip all whitespace characters (space, tab, newline)
    * read entry until the next empty character. */
scanf("%s", str2);

/*if input is like that:_ Supposing the space */
_ _xyz123_ _ _45_ _67
```

- scanf will pass the first two spaces and str2 will be "xyz123" .
- Then it will see the space and will stop reading.
- The next scanf ("% s", ...), will skip these gaps and read "45".

Reading a String

- If you wish, you can write your own read function, which will read the input until "Enter" is entered.

```
char *ReadLine(char *str){
  char ch;
  char *p = str;
  while((ch=getchar()) != '\n')
     *p++=ch;
  *p = '\0';  /* The end of the string is ended with NULL
   characters. */
  return str;
} /* end-ReadLine */

main(){
  char str[80];

  ReadLine(str);
  printf("Entered row= <%s>\n", str);
} /* end-main */
```

Reading a String

- Another version can be until you enter "Enter" or number of "n" characters entered.

```
char *ReadNLine(char *str, int n){
  char ch;
  char *p = str;
  while (n-- > 0){
    if ((ch = getchar()) == '\n') break;
    *p++ = ch;
  } /* end-while */
  *p = '\0'; /* stringin sonunu NULL karakter yap */
  return str;
} /* end-ReadNLine */

main(){
  char str[80]; char *p = NULL;
    p = ReadNLine(str, 79); /* can get a maximum of 79
   characters */
  printf("Entered row= <%s>\n", p);
} /* end-main */
```

String Operations

- C standard library contains many functions to manipulate strings.
  - To use these functions, you need to add the file <string.h>.
    #include <string.h>
- Some important functions:
  - strlen(const char *str);
  - strcpy(char *str1, const char *str2);
  - strcat(char *str1, const char *str2);
  - strcmp(const char *str1, const char *str2);
- We will enter the details of these functions in the coming semester.

Example: String Length

```
#include <stdio.h>
int main(void){
   char s[40];
   int  k = 0;

   /* read array */
   printf("Write something : ");
   gets(s);

   /* count character until terminator character */
   while( s[k]!='\0' )
       k++;
    printf("Array length : %d\n" ,k);

 return 0;
}
```

Example: String Reverse

```
#include <stdio.h>
int main(void){
   char s[40], temp;
   int  i, n;
   /* read array */
   printf("Write something : ");
   gets(s);
   /* until terminator character */
   for(n=0; s[n] != '\0'; n++);
   for(i=0; i<n/2; i++){
      temp = s[n-i-1];
      s[n-i-1] = s[i];
      s[i]     = temp;
   }
   printf("Reverse           : %s\n", s);
 return 0;
}
```

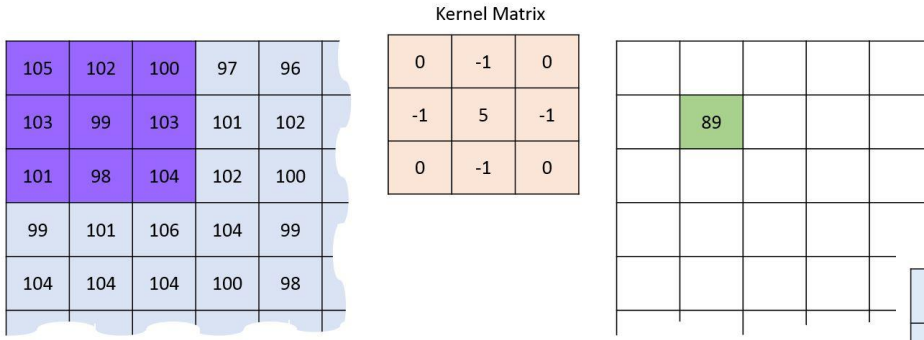

HOMEWORK
NEXT

ÖDEV (HOMEWORK)

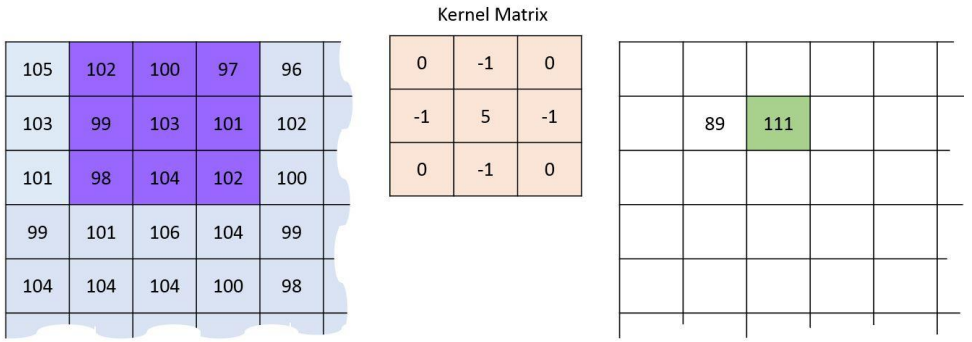
Soru 1: Ödevin 15 puanlık bölümünü oluşturmaktadır. (**Question 1:** It consists of 15 points of the homework.)

Aşağıdaki gibi f isimli iki boyutlu MxN boyutlu bir matrisi $0-255$ arasında rasgele değerler alacak şekilde oluşturunuz. Hazırlanan bu f matrisi, $3x3$ boyutlu ve değerleri dışarıdan girilecek bir v filtresi ile iki boyutlu filtre işlemine tabi tutulacaktır. Elde edilen sonuçlar iki boyutlu g matrisinde saklanacaktır. Filtre işlemi f matrisinin kenar elemanları hariç tüm elemanları üzerine uygulanacaktır. Örnek bir hesaplama aşağıda verilmektedir. Bu işlemleri gerçekleştiren C kodunu yazınız.

(Create a two-dimensional MxN-sized matrix named f, which takes random values from $0-255$. This matrix f will be subjected to a two-dimensional filter process with a filter v whose $3x3$ dimensional values will be entered from keyboard. The results will be stored in a two-dimensional g matrix. The filter process will be applied to all elements of the matrix f except the edge elements. An example calculation is given below. Write the C code that performs these operations.)



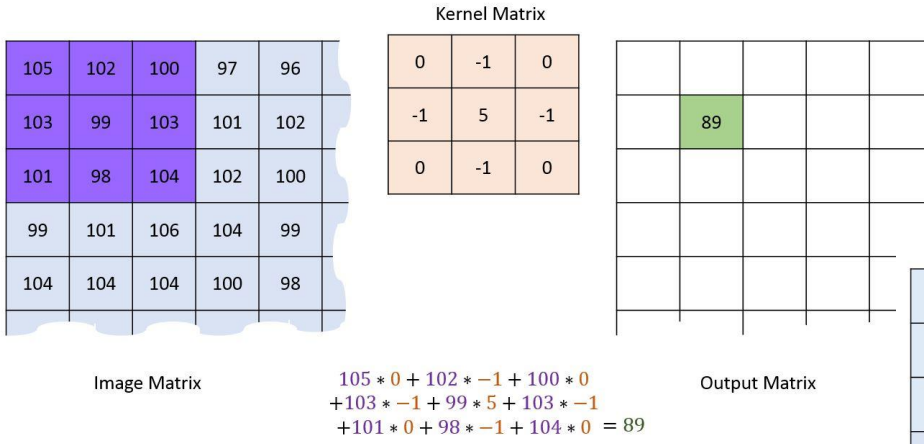
$$105*0+102*-1+100*0+103*-1+99*5+103*-1+101*0+98*-1+104*0=89$$



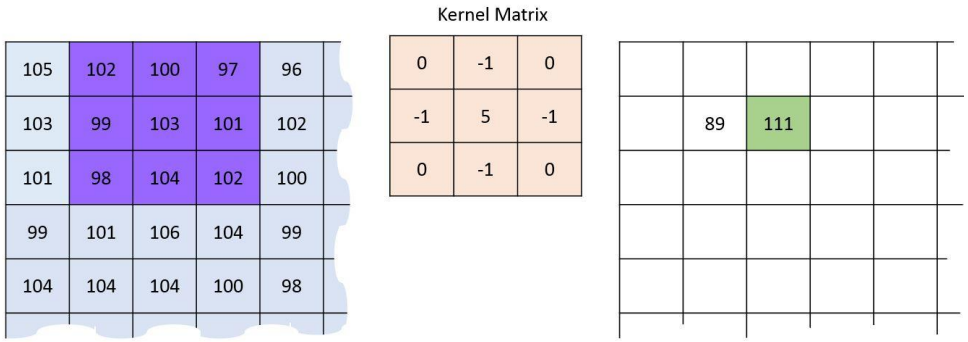
$$102*0+100*-1+97*0+99*-1+103*5+101*-1+98*0+104*-1+102*0=111$$

Soru 2: Ödevin 10 puanlık bölümünü oluşturmaktadır. (**Question 2:** It consists of 10 points of the homework.)

Birinci soruda hazırlanan algoritma kenar elemanlarına uygulanmamaktaydı. Bu adımda v filtresini f matrisinin kenar elemanlarına da uygulayabilecek bir çözüm geliştiriniz. Çözümlerinizde tamamen özgünlük aranacak olup değerlendirmeler buna göre yapılacaktır.
(The algorithm prepared in the first question was not applied to the edge elements. In this step, develop a solution that can also apply the filter v to the edge elements of the matrix f. Completely originality will be expected in your solutions and evaluations will be made accordingly.)

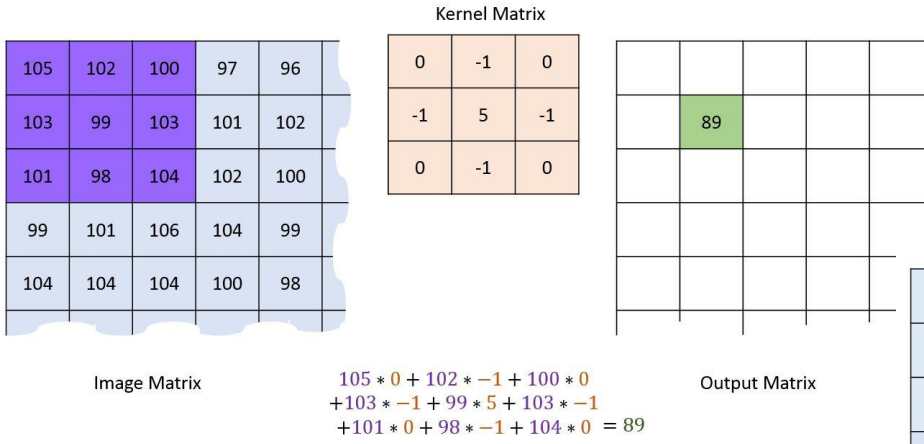


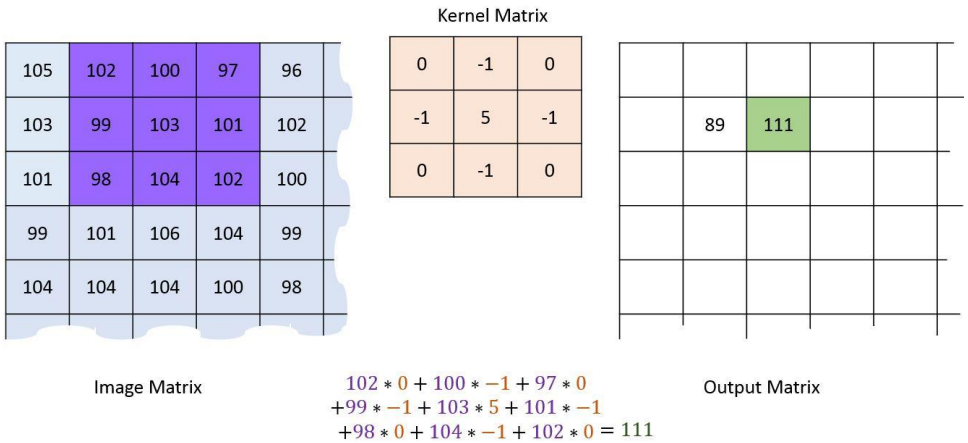
$$105*0+102*-1+100*0+103*-1+99*5+103*-1+101*0+98*-1+104*0=89$$



$$102*0+100*-1+97*0+99*-1+103*5+101*-1+98*0+104*-1+102*0=111$$

Not: Ödevin gösterim tarihi 16-20 Aralık tarihlerinde laboratuvar dersleridir. Ödev e-mail veya başka bir şekilde gönderilmeyecek, yüz yüze bilgisayar başında anlatılacaktır. Bütün öğrenciler kayıt oldukları laboratuvar saatinde ödevini gösterecektir. Bunun haricinde ödev kontrolü yapılmayacaktır. Ödev 25 puandır. Ödevden alınan not final ve bütünleme sınav notuna etki edecektir. **Kopya ödevler -15 puan olarak değerlendirilecektir.**
(**Note:** The deadline for the assignment is laboratory lectures on December 16-20. The assignment will not be sent by 3-mail or in any other way. The homework will be explained face to face at the computer. All students will show their homework during the laboratory hours. Homework control will not be done by another way or another time. The assignment is 25 points. The grade from the assignment will affect the final and make-up exam grade. **Copy assignments will be evaluated as -15 points.**)





References

- 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