CME111 Programming Languages I

Week 12
Character Array (String)

Assist. Prof. Dr. Caner ÖZCAN

String Definition

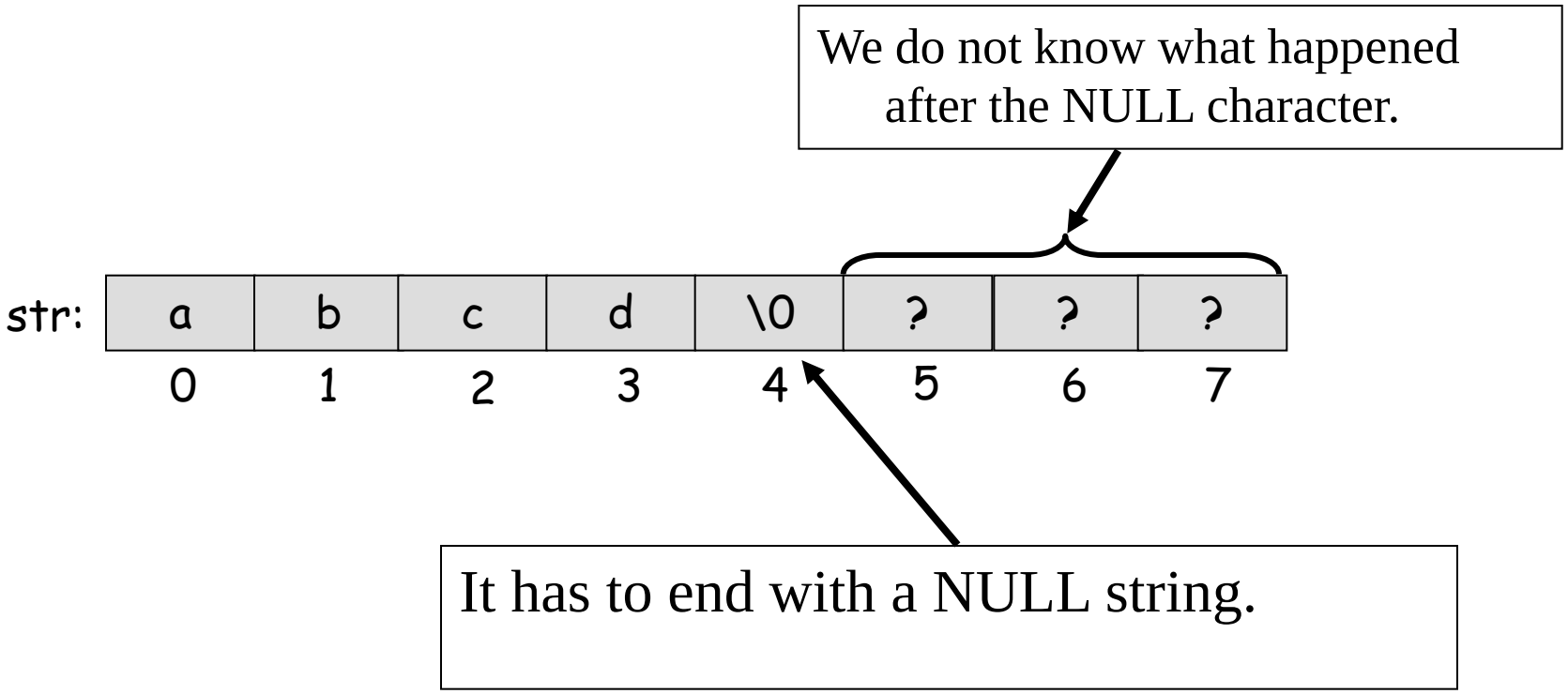
- We learned multidimensional arrays and arrays.
- String is actually an array of what we call in.
- Variable type char so that the characters arrays are called string.
- For example, an integer (int) array stores the total number integer; in string we store character (char).
- Names, addresses, user names, and phones for everything ... We use the character strings that can be expressed verbally.

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' .

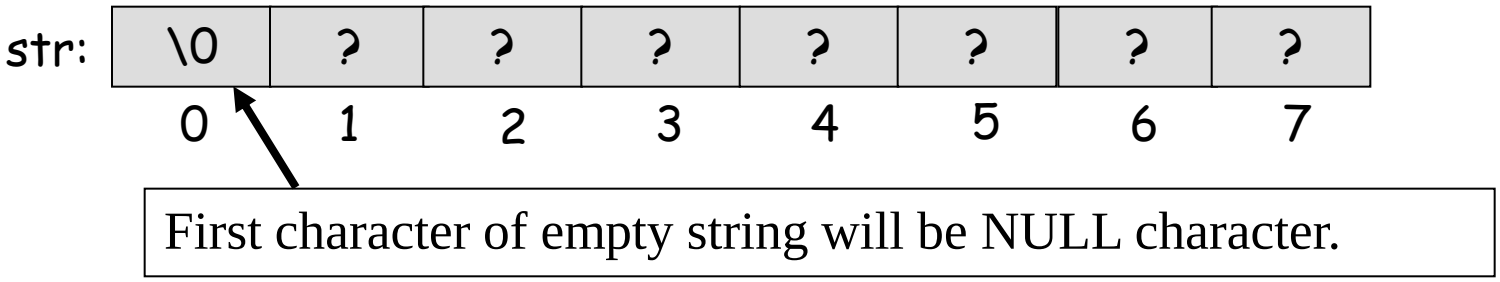
String Definition

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



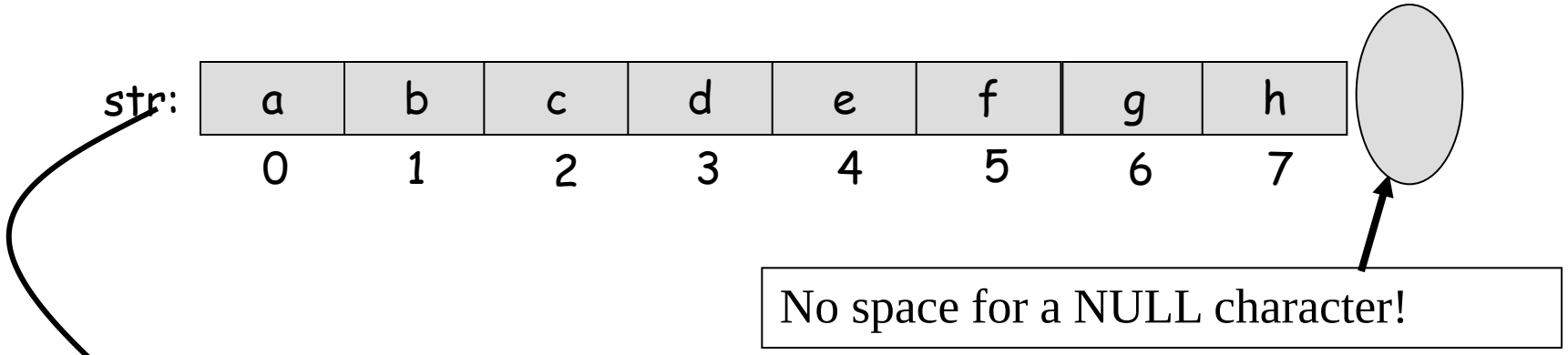
Empty String

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



String Maximum Length

- An 8 characters length string, such as "abcdefgh" can not be stored on str.



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

String: WARNING

- char str [8] as a declaration also emphasizes simply str could store up to 8 characters.

- At any point during the operation of the program, we may want to keep more than 8 characters on the str .

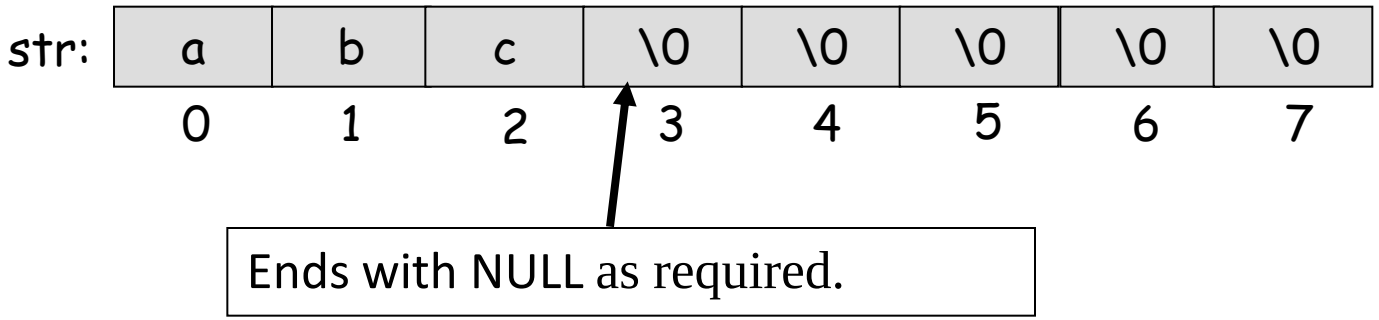
- But if "str" stores string, we store maximum 8-1 = 7 characters, and always have to end with a NULL.

String: Initialization

- As a string of instant identification in other sequences defined as follows.

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

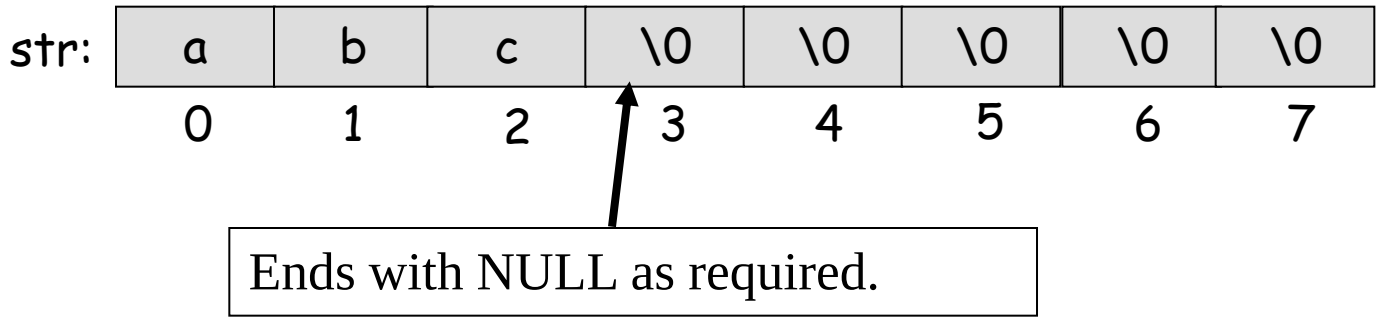
- Remember unspecified elements are filled with 0, NULL character that is happening.
  - Therefore, the above statement corresponds to the following string.



String: Initialization

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

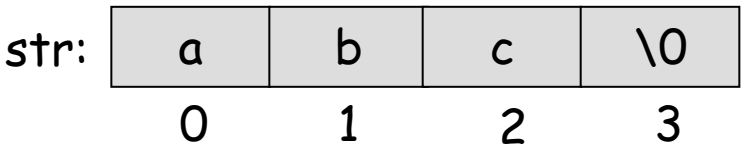
```
char str[8] = "abc"; /* same with previous */
```



String: Initialization

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

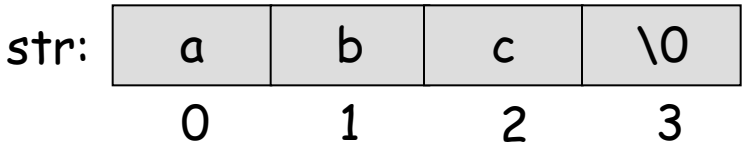
```
char str[] = "abc";
```



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];

/* 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", ...), this gap will pass and "45" to be read.

Reading a String

- If you read "Enter" to input until you enter, you can write your own reading function.

```
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 s contains many functions to manipulate strings.
  - To use this function, you need to add in the file of <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 function in near future.

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;
}
```

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