Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: Fall, Year: 2023, B.Sc. in CSE (Day)

LAB REPORT NO: 06

Course Title: Structured Programming Lab

Course Code: CSE 104 Section: 231

Lab Experiment Name: String processing

Student Details

Name	**ID**
Promod Chandra Das	231002005

Lab Date : _

Submission Date : _

Course Teacher's Name : _

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status

Marks: ………………………………… **Signature:.....................**

Comments:... **Date:..............................**

❖ **TITLE OF THE LAB EXPERIMENT: String processing**

❖ **OBJECTIVES/AIM**

• To attain knowledge on string, string declaration, string initialization, string library function
• To use string functions to process C-strings
• To implement programs using String

Strings are actually one-dimensional array of characters terminated by a null character 0\0
0

. Thus a null-
terminated string contains the characters that comprise the string followed by a null.

For example:
1 ch a r c [ ] = " c s t r i n g " ;

When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a
null character \0 at the end by default.

Let us now look at a sample program to get a clear understanding of declaring and initializing a string in C
and also how to print a string.

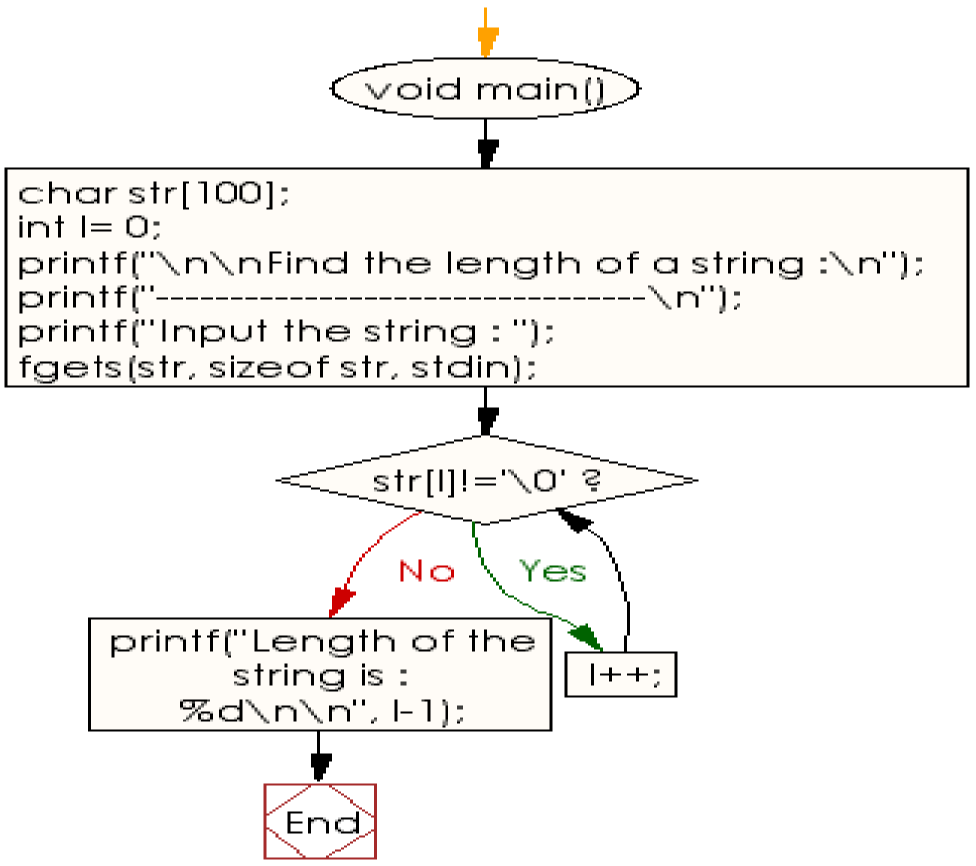
```
// C program t o i l l u s t r a t e s t r i n g s

 #i n cl u d e <s t d i o . h>
int  main ( )
{
// d e c l a r e and i n i t i a l i z e s t r i n g
 ch a r s t r [ ] = "Computer" ;
// p ri n t s t r i n g
 p r i n t f ( "%s " , s t r ) ;
r e t u r n 0 ;
 }
```
We can see in the above program that strings can be printed using normal printf statements just like we print
any other variable. Unlike arrays, we do not need to print a string, character by character. The C language
does not provide an inbuilt data type for strings but it has an access specifier “%s” which can be used to directly

print and read strings.

- **Exercise :**

Write a program in C to find the length of a string without using library function.:

1. **Flow chart:**



void main()
char str[100];
int l= 0;
printf("\n\nFind the length of a string :\n");
printf("-----------------------------------\n");
printf("Input the string : ");
fgets(str, sizeof str, stdin);
str[l]!='\0' ?
No
Yes
printf("Length of the string is : %d\n\n", l-1);
l++;
End

2. **Algorithm:-**

1. nitialize a variable `length` to 0.

2. Iterate through the characters of the string until the null character `\0` is encountered.
3. Increment the `length` variable for each character.
4. Return the final value of `length`

3.**Pseudocode:**

```
function findStringLength(str):
    length = 0
    while str[length] is not '\0':
        length = length + 1
    return length

// In the main program
inputString = get_user_input()
length = findStringLength(inputString)
display_result(length)
```

3. C program:-

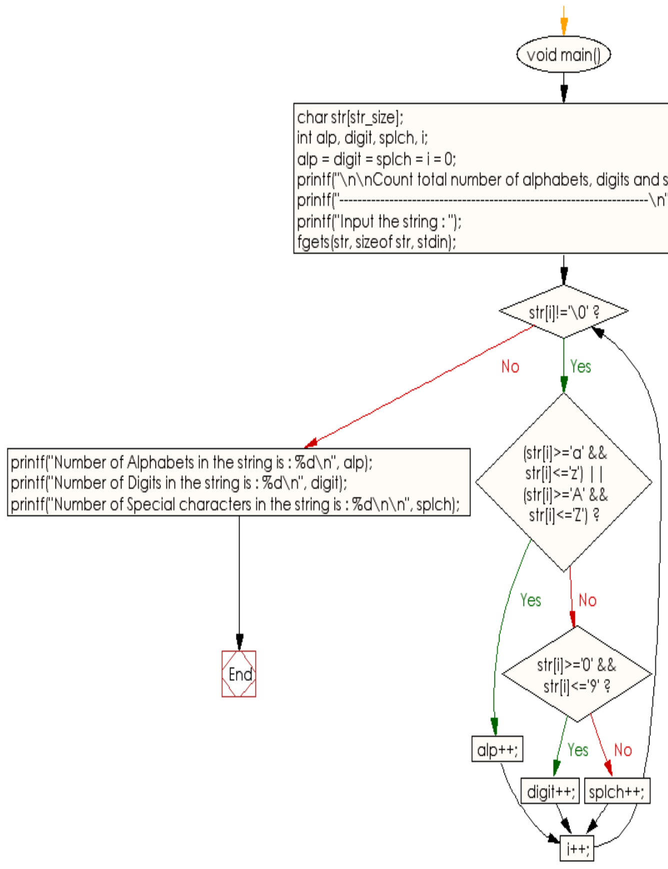
```c
#include <stdio.h>

// Function to calculate the length of a string
int findStringLength(char str[]) {
    int length = 0;

    // Iterate through the characters until the null character '\0' is encountered
    while (str[length] != '\0') {
        length++;
    }

    return length;
}

int main() {
    char inputString[100]; // You can change the size according to your needs
    // Get input from the user
    printf("Enter a string: ");
    scanf("%s", inputString);

    // Calculate the length of the string
    int length = findStringLength(inputString);

    // Display the result
    printf("Length of the string: %d\n", length);

    return 0;
}
```

4. **RESULT / OUTPUT**

```
Enter a string: Hello
Length of the string: 5
```

- **Write a program in C to count total number of alphabets, digits and special characters in a string.**

1. **Flow chart:**



```
char str[str_size];
int alp, digit, splch, i;
alp = digit = splch = i = 0;
printf("\n\nCount total number of alphabets, digits and special characters :\n");
printf("--------------------------------------------------------------------\n");
printf("Input the string : ");
fgets(str, sizeof str, stdin);
```

```
printf("Number of Alphabets in the string is : %d\n", alp);
printf("Number of Digits in the string is : %d\n", digit);
printf("Number of Special characters in the string is : %d\n\n", splch);
```

void main()
str[i]!='\0' ?
No
Yes
(str[i]>='a' && str[i]<='z') || (str[i]>='A' && str[i]<='Z') ?
Yes
No
str[i]>='0' && str[i]<='9' ?
Yes
No
alp++;
digit++;
splch++;
i++;
End

2. **<u>Algorithm:-</u>**

1. nitialize variables for counting alphabets, digits, and special characters to 0.
2. Iterate through the characters of the string until the null character '\0' is encountered.
3. Check each character and update the corresponding counter.
4. Display the counts of alphabets, digits, and special characters.

3. **<u>Pseudocode:</u>**

```
function countCharacters(str):
    alphabets = 0
    digits = 0
    specials = 0

    for i = 0 to length(str) - 1:
        if str[i] is an alphabet:
            alphabets = alphabets + 1
        else if str[i] is a digit:
            digits = digits + 1
        else:
            specials = specials + 1

    display "Alphabets: ", alphabets
    display "Digits: ", digits
    display "Special Characters: ", specials

// In the main program
inputString = get_user_input()
countCharacters(inputString)
```

4. **C program:-**

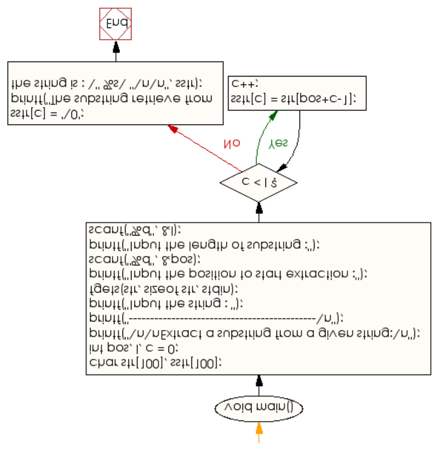
```
#include <stdio.h>

void countCharacters(char str[]) {
    int alphabets = 0, digits = 0, specialChars = 0;

    for (int i = 0; str[i] != '\0'; i++) {
        if ((str[i] >= 'A' && str[i] <= 'Z') || (str[i] >= 'a' && str[i] <= 'z')) {
            alphabets++;
        } else if (str[i] >= '0' && str[i] <= '9') {
            digits++;
        } else {
            specialChars++;
        }
    }

    printf("Alphabets: %d\n", alphabets);
    printf("Digits: %d\n", digits);
    printf("Special Characters: %d\n", specialChars);
}

int main() {
    char inputString[100];

    printf("Enter a string: ");
    fgets(inputString, sizeof(inputString), stdin);

    countCharacters(inputString);

    return 0;
}
```

5. **RESULT / OUTPUT**

```
Enter a string: Hello123! How are you?
Alphabets: 16
Digits: 3
Special Characters: 3
```

- **Write a program in C to extract a substring from a given string.**

1. **Flow chart:**


void main()
char str[100], sstr[100];
int pos, l, c = 0;
printf("\n\nExtract a substring from a given string:\n");
printf("--\n");
printf("Input the string : ");
fgets(str, sizeof str, stdin);
printf("Input the position to start extraction :");
scanf("%d", &pos);
printf("Input the length of substring :");
scanf("%d", &l);
c < l ?
Yes
sstr[c] = str[pos+c-1];
c++;
No
sstr[c] = '\0';
printf("The substring retrieve from
the string is : \" %s\ "\n\n", sstr);
End

2. Algorithm:-

1. Start
2. Input the main string.
3. Input the start index of the substring.
4. Input the length of the substring.
5. Extract the substring using the start index and length.
6. Display the extracted substring.
7. End.

3. Pseudocode:

```
1. Start
2. Input mainString
3. Input startIndex
4. Input length
5. substring = ExtractSubstring(mainString, startIndex, length)
6. Display "Extracted Substring: ", substring
7. End

Function ExtractSubstring(mainString, startIndex, length):
    return mainString[startIndex : startIndex + length]
```

4.C program:-

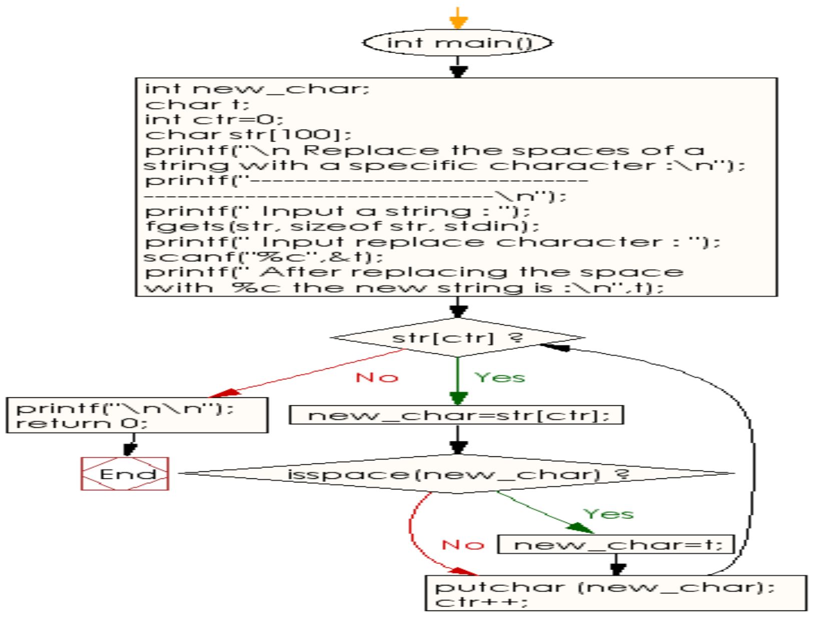
```
#include <stdio.h>
#include <string.h>

void extractSubstring(char mainString[], int startIndex, int length, char substring[]) {
    int i, j;

    for (i = startIndex, j = 0; j < length && mainString[i] != '\0'; i++, j++) {
        substring[j] = mainString[i];
    }

    substring[j] = '\0';
}

int main() {
    char mainString[100], substring[100];
    int startIndex, length;

    // Input
    printf("Enter the main string: ");
    fgets(mainString, sizeof(mainString), stdin);
    mainString[strcspn(mainString, "\n")] = '\0';  // Remove newline character

    printf("Enter the start index: ");
    scanf("%d", &startIndex);

    printf("Enter the length: ");
    scanf("%d", &length);

    // Extract Substring
    extractSubstring(mainString, startIndex, length, substring);

    // Output
    printf("Extracted Substring: %s\n", substring);

    return 0;
}
```

5. RESULT / OUTPUT

```
Enter the main string: Hello, World!
Enter the start index: 7
Enter the length: 5
Extracted Substring: World,
```

Write a program in C to replace the spaces of a string with a specific character.

1**. Flow chart:**



```
int main()
```

```
int new_char;
char t;
int ctr=0;
char str[100];
printf("\n Replace the spaces of a string with a specific character :\n");
printf("-----------------------------------------------------------\n");
printf(" Input a string : ");
fgets(str, sizeof str, stdin);
printf(" Input replace character : ");
scanf("%c",&t);
printf(" After replacing the space with  %c the new string is :\n",t);
```

```
str[ctr] ?
new_char=str[ctr];
isspace(new_char) ?
new_char=t;
putchar (new_char);
ctr++;
printf("\n\n");
return 0;
```

2. Algorithm:-

1. Accept the input string and the character to replace spaces with.
2. Iterate through each character of the input string.
3. If a space character is encountered, replace it with the specified character.
4. Print the modified string.

3. Pseudocode:

```
function replaceSpaces(inputString, replaceChar):
    for each character in inputString:
        if character is a space:
            replace the space with replaceChar
```

Input:

```
  Accept inputString and replaceChar from the user

Call replaceSpaces(inputString, replaceChar)

Output:
  Display the modified inputString
```

5. **C program:-**

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

void replaceSpaces(char str[], char replaceChar);

int main() {
    char inputString[100];
    char replaceChar;

    // Input
    printf("Enter a string: ");
    fgets(inputString, sizeof(inputString), stdin);

    printf("Enter the character to replace spaces with: ");
    scanf(" %c", &replaceChar);

    // Function call to replace spaces
    replaceSpaces(inputString, replaceChar);

    // Output
    printf("Modified string: %s\n", inputString);

    return 0;
}

// Function to replace spaces with a specific character
void replaceSpaces(char str[], char replaceChar) {
    int length = strlen(str);

    for (int i = 0; i < length; i++) {
        if (str[i] == ' ') {
            str[i] = replaceChar;
        }
    }
}
```

6. RESULT / OUTPUT

```
Enter a string: Hello World
Enter the character to replace spaces with: -
Modified string: Hello-World
```

❖ DISCUSSION:-

String processing is a fundamental aspect of programming that involves manipulating and analyzing sequences of characters, typically represented as strings. In most programming languages, strings are treated as arrays of characters, and various operations can be performed on them