Characters and String in C Language

FUNCTIONS

- A **string** is a collection of characters, it may include letters, digits and various **special characters** such as #, $.
- **String literals**, or **string constants**, are written in double quotation
- A string is stored as an array of characters that ends with **null character ('\0')**.

MEMORY REPRESENTATION OF STRING

C	O	M	P	U	T	E	R	\0	

The values stored in an array of characters can be accessed using the index of the array. The user can input, process and displays the individual characters of string in the same way as an array.

EXAMPLES OF STRING LITERALS, OR STRING CONSTANTS

"F9-Islamabad" " House # 123"

STRING DECLARATION

C Language stores a string as an array of characters. The size of the string array should be long enough to store the null character. For example, the size of string array must be at least 16 if the string value has 15 characters.

Syntax:	**Example:**
char string-name[length];	char str[50];

A string is accessed via a *pointer* to the first character in the string. Thus, in C, it's appropriate to say that a **string is a pointer**—in fact, a pointer to the string's first character. In this sense, strings are like arrays, because an array is also a pointer to its first element.

STRING INITILIZATION

Syntax:	**Example:**
char string-name[length]= String-value;	char str[50] = "CMPC-201";

EXAMPLE PROGRAM

Creating String with 3 different methods in a Program.

```
#include<stdio.h>
int main()
{
    const char *st1="My name is Jamal ";
    const char st2[]={'C','L','A','S','S'};
    char st3[]="BS-CS";

    puts("output is::");
    printf("%s \n %s \n %s",st1,st2,st3);

}
```

```
output is::
My name is Jamal
 CLASS
 BS-CS
------------------------
```

Program-2:

Write a Program that input a string from the user and find it's length without using any built-in function.

```c
// input string and find its length
#include <stdio.h>

int main()
{
    char nm[20];
    int i=0;
    printf("Enter your name:");
    gets(nm);

    while(nm[i]!='\0')
    {
        i++;
    }
    printf("Length of my name is = %d character",i);
}
```

Output of program-2:

```
Enter your name:Khuram Shahzad
Length of my name is = 14 character
--------------------------------
```

Program-3:
Write a Program that input a string/sentence from the user and find all vowel characters in the string.

```c
#include <stdio.h>
int main()
{
    char str[20];
    int i=0,v=0;
    printf("Enter any text::");
    gets(str);

    while(str[i]!=NULL)
    {
        switch(str[i])
        {
            case 'a':
            case 'e':
            case 'i':
            case 'o':
            case 'u':
            case 'A':
            case 'E':
            case 'I':
            case 'O':
            case 'U':
            v++;
        }
        i++;
    }
    printf("TOTAL NUMBER of vowel is = %d",v);
}
```

Output of program-3:

```
Enter any text::programming class
TOTAL NUMBER of vowel is = 4
--------------------------------
```

CHARACTER-HANDLING LIBRARY

The **character-handling library** (<ctype.h>) contains several predefined functions to perform useful tests and character manipulations. Each function takes character input.

Following Table summarizes the functions of the character-handling library.

Function	Function Description
isdigit( )	Returns a true value if given character is a *digit* and 0 (false) otherwise.
isalpha( )	Returns a true value if given character is a *letter* and 0 otherwise.
isalnum( )	Returns a true value if given character is a *digit* or a *letter* and 0 otherwise.
islower( )	Returns a true value if given character is a lowercase letter and 0 otherwise.
isupper( )	Returns a true value if given characteris an uppercase letter and 0 otherwise.
isspace( )	Returns a true value if given character is a whitespace character—newline ('\n'), space (' '), form feed ('\f'), carriage return ('\r'), horizontal tab ('\t') or vertical tab ('\v')—and 0 otherwise.
tolower( )	If given character is an uppercase letter, tolower return its lowercase letter. Otherwise, tolower returns the argument unchanged.
toupper()	If given character is a lowercase letter, toupper returns its uppercase letter. Otherwise, toupper returns the argument unchanged.

Program-4:

Write a Program that input a sentence from the user and convert all lowercase characters to uppercase characters in the string.

```
#include <stdio.h>
#include<ctype.h>
int main()
{
    char str[25];
    int i=0;
    printf("Enter any text::");
    gets(str);

    while(str[i]!=NULL)
    {
        //str[i]=str[i]-32;
        str[i]=toupper(str[i]);
        i++;
    }
    printf("Text in Upper case is: %s",str);
}
```

Output of program-4:

```
Enter any text::programming class
Text in Upper case is: PROGRAMMING CLASS
--------------------------------
```

Program-5:

Write a Program that input a string from the user and convert all uppercase letters into lowercase letters and vice versa, and then display the resultant string.

```c
#include <stdio.h>
#include<ctype.h>
int main()
{
    char str[25];
    int i=0;
    printf("Enter any text::");
    gets(str);

    while(str[i]!='\0')
    {
        if(isupper(str[i]))
        str[i]=tolower(str[i]);
        else
        str[i]=toupper(str[i]);
        i++;
    }
    printf("Text in change case is: %s",str);
}
```

Output of program-5:

```
Enter any text::Information Technology
Text in change case is: iNFORMATION tECHNOLOGY
--------------------------------
```

Program-6:

Write a Program that input input a string and count the Uppercase Letters ,Lower case Letters, Upper case vowel and lower case vowels

```c
#include<stdio.h>
int main()
{
char str[50];
printf("Enter some text::");
gets(str);
int x=0;
int uc=0,lc=0,uv=0,lv=0;
while (str[x]!='\0')
{
if(str[x]=='A'||str[x]=='E'||str[x]=='I'||str[x]=='O'||str[x]=='U')
uv++;
if (str[x]=='a'||str[x]=='e'||str[x]=='i'||str[x]=='o'||str[x]=='u')
lv++;
if (str[x] >='A' && str[x]<='Z')
uc++;
if (str[x] >='a' && str[x]<='z')
lc++;

x++;
}
printf("UPPER CASE Letters :: %d\n",uc);
printf("Lower CASE Letters:: %d\n",lc);
printf("UPPER CASE VOwEL :: %d\n",uv);
printf("LOWER CASE Vowel :: %d\n",lv);
}
```

Output of program-6:

```
Enter some text::Information Technology
UPPER CASE Letters :: 2
Lower CASE Letters:: 19
UPPER CASE VOwEL :: 1
LOWER CASE Vowel :: 7

--------------------------------
```

Program-7:

Write a Program that input a string/sentence from the user and count total words in the string.

```
#include <stdio.h>

int main()
{
    char str[60];
    int i=0;
    int w=1;
    printf("Enter any text::");
    gets(str);

    while(str[i]!='\0')
    {
        if(str[i]==' ')
        w++;

        i++;
    }
    printf("total words in sentense are : %d",w);

}
```

Program-8:

Write a Program that input a string/sentence from the user and copy into another string and display both the strings.

```
#include <stdio.h>
int main()
{
    char st1[20],st2[20];
    int i=0;
    printf("Enter any text::");
    gets(st1);

    while(st1[i]!='\0')
    {
        st2[i]=st1[i];
        i++;
    }
    st2[i]='\0';
    printf("\nstring-1 is  =%s",st1);
    printf("\nstring-2 is  =%s",st2);
}
```

Program-9:

Write a Program that input a first name string ,last name string and combine both strings into fullname ,and display the content of fullname string.

```c
#include <stdio.h>
int main()
{
    char st1[20],st2[20],st3[40];
    int i=0,j=0;
    printf("Enter 1st Name:");
    gets(st1);

    printf("Enter 2nd Name:");
    gets(st2);

    while(st1[i]!='\0')
    {
        st3[i]=st1[i];
        i++;
    }
    st3[i++]=' ';

    while(st2[j]!='\0')
    {
        st3[i]=st2[j];
        i++;
        j++;
    }
    st3[i++]='\0';

    printf("Full name :%s",st3);
}
```

Program-10:

Write a Program that input a string from the user and reverse it without using any built-in function.

```c
#include<stdio.h>
int main() {
    char str[100], temp;
    int size = 0, i = 0;
    printf("Enter a string:");
    gets(str);
    while (str[size] != '\0')
    {
        size++;
    }
    size = size - 1;
    while (i < size) {
        temp = str[i];
        str[i] = str[size];
        str[size] = temp;
        size--;
        i++;
    }
    printf("string After reverse :");
    puts(str);
 }
```

Output of program-10:

```
Enter a string:Programming
string After reverse :gnimmargorP

--------------------------------
```

Function	Function Description
`fgets()`	• This Function reads characters until a newline character or the end-of-file indicator is encountered. • The arguments to fgets are an array of type char, the maximum number of characters that can be read and the stream from which to read. A null character ('\0') is appended to the array after reading terminates. • `fgets( sentence, SIZE, stdin );`
`puts()`	• Function takes a string (char *) as an argument and prints the string followed by a newline character.
`getchar()`	This Function reads a single character from the standard input and returns it as an integer. If the end-of-file indicator is encountered, getchar returns EOF.
`putchar()`	This Function prints its character argument.
`sscanf()`	This Function read formatted data from a string
`sprint()`	This Function print formatted data into an array of type char.

getchar() and putchar()

Write a Program that input a single character and create string, and display string by single character.

```c
//input single character and output single character
//getchar() and putchar()

#include<stdio.h>
int main()
{
    char st[50];
    char c;
    int i=0;
    printf("Enter any line of text:");

    while((c=getchar())!='\n')
    st[i++]=c;

    st[i]='\0';
    printf("you entered::");

    i=0;
    while(st[i]!=NULL)
    putchar(st[i++]);

}
```

fgets()

```c
#include <stdio.h>
int main()
{
    char str[50];

    printf("enter name:");
    //getting string from standard input
    fgets(str,50,stdin);
    printf("%s",str);

}
```

sscanf()

string scan
sscanf() function, which linearly reads the value from a char[] array and store each value into variables of matching data type by specifying the matching format specifier in sscanf() function,

```
#include<stdio.h>

int main()
{
char ar[20] = "Abrar M 19 3.65";

char nm[10];
char g;
int a;
float cg;

/* Calling sscanf() to read multiple values from a char[] array
 and store each value in matching variable */
sscanf(ar, "%s %c %d %f", nm, &g, &a, &cg);
printf("Name : %s \n", nm);
printf("Gender : %c \n", g);
printf("Age : %d \n", a);
printf("CGPA : %f ", cg);
}
```

sprintf()

string print
following example, use sprintf() function, that store different values in a char[] array , and display the string value.

```
#include<stdio.h>
int main()
{
char target[20];
char name[10] = "Abu Bakar";
char gender  = 'M';
int age = 25;
float height = 5.7;

/* Calling sprintf() function to read multiple variables
 and store their values in a char[] array i.e. string.*/

sprintf(target, "%s %c %d %f", name, gender, age, height);
printf("complete record is : %s ", target);
}
```

sprintf()

```
#include<stdio.h>
int main()
{
    int sal;
    char nm[30], ds[30], rec[60];

    printf("Enter your name: ");
    gets(nm);
    printf("Enter your designation: ");
    gets(ds);
    printf("Enter your salary: ");
    scanf("%d", &sal);
    sprintf(rec, "Welcome %s \nDesignation: %s\nSalary: %d",
        nm, ds, sal);

    printf("\n%s", rec);
}
```

The string-handling library (<string.h>) provides many useful functions for

1. String Manipulation functions
2. String comparison functions
3. Search functions
4. Memory Functions

1) Following Table summarizes the string-manipulation functions.

Function	Function Description
strcpy()	This function copy one string to other string including NULL character **strcpy(s1,s2) : copies the s2 into s1**
strncpy( )	Copies first **n** characters of one string to other **strncpy(s1,s2,n) copies n character of string s2 in string s1**
strcat( )	Appends one string at the end of another. **strcat(str1,str2) str2 is appended at the end of str1**
strncat( )	Appends first n characters of string at the end of another. **strncat(str1,str2,n) appends n characters of str2 at the end of str1**
strlwr( )	Convert all characters of a string to lower case
strupr( )	Convert all characters of a string to upper case

strcpy()

```c
#include <stdio.h>
#include<string.h>
int main()
{
    char str1[20]="Hello";
    char str2[20];

    strcpy(str2,str1);  //copy str1 to str2
    printf(" %s ",str2);

}
//output will be Hello
```

strncpy()

```c
#include <stdio.h>
#include<string.h>
int main()
{
    char str1[20]="Hello";
    char str2[20];

    strncpy(str2,str1,3);
    printf("%s",str2);

}
//output will be Hel
```

2) **Following Table summarizes the string-comparison functions.**

Function	Function Description
`strcmp()`	• This function compares two strings character by character. • Comparison is case sensitive • It returns integer value o **0**: if 1st string if equal to 2nd string o **>1** if 1st string is greater than 2nd o **<1** if 1st string is less or smaller than 2nd string ❖ **n =strcmp(s1,s2)** *n is of type int*
`strncmp( )`	Compare specified number of **n** characters in two strings **strncmp(s1,s2,n)**
`Sticmp( )`	Compaares two strings without determination between upper and lower case **stricmp(str1,str2)**

Program-11:

Write a Program that input a string and check if it is palindrome or not Palindrome is a string that reads the same backwords as forwards such as MADAM and MOM

```
#include<stdio.h>
#include<string.h>
int main() {

    char str[100], str1[100],temp;
    int size = 0, i = 0;

    printf("Enter a string:");
    gets(str);
    strcpy(str1,str);
    while (str[size] != '\0')
    {
        size++;
    }
    size = size - 1;
    while (i < size) {
        temp = str[i];
        str[i] = str[size];
        str[size] = temp;
        size--;
        i++;
    }
    printf("string After reverse :");
    puts(str);
    int n=strcmp(str,str1);
    if(n==0)
    printf("Palindrome");
    else
    printf("not a palindrome");
    return 0;
}
```

3) **Following Table summarizes the search functions.**

Function	Function Description
`strchr()`	• Find the first occurrence of a character in string and return a pointer to the this character, and NULL if character Not found. ❖ **strchr (str,ch)**
`strrchr( )`	• Find the last occurrence of a character in string and return a pointer to the this character, and NULL if character Not found. ❖ **strrchr (str,ch)**
`strstr( )`	• This function is used to Find the first occurrence of a 2nd string Within the 1st string and return a pointer. **strstr (str1,str2)**

`strchr()`

```c
#include <stdio.h>
#include <string.h>
int main ()
 {
   const char str[] = "This is just a String";
   char ch;
   char *p;
   puts(str);
   printf("any any character from above string:");
   ch=getchar();
   p = strchr(str, ch);
   printf("%s", p);

}
```

`strstr()`

```c
#include <stdio.h>
#include<string.h>
int main()
{
    char str1[50]="this is my online class";
    char str2[20];
    puts(str1);
    printf("Enter String:");
    gets(str2);
    if(strstr(str1,str2)!=NULL)
    printf("string found");
    else
    printf("string NOT found");

}
```

other functions

Function	Function Description
strlen()	Return the length of string.

Program-12:

Write a Program that input a string from the user and display it in triangle. for example if the user enter math ,the program display it like this

```
#include<stdio.h>
#include<string.h>
int main() {

    char str[20];
    int i,j,n;

    printf("Enter a string:");
    gets(str);
    n=strlen(str);
    for(i=0;i<n;i++)
    {
        for(j=0;j<=i;j++)
        printf("%c ",str[j]);
    printf("\n");
    }
}
```

```
m
ma
mat
math
```

```
Enter a string:math
m
m a
m a t
m a t h
```

Converting a string to uppercase using a pointer and user-defined function

```c
#include <stdio.h>
#include <ctype.h>
void convertToUppercase(char *sPtr );

int main( void )
{
char string[50];
printf( "Enter any String:");
gets(string);

convertToUppercase( string );
printf( "The string after conversion is: %s\n", string );
}

void convertToUppercase(char *sPtr )
{
    while (*sPtr!=NULL )
    {
    *sPtr = toupper( *sPtr );
    sPtr++;
    }
}
```

****

```c
/*
this program inputs a string in main(), pass it to
stringrev()function that reverse it character by character
after reversing, main() print it.
*/
#include <stdio.h>
void stringrev(char*);

int main()
{
   char st[50];
   printf("Input a string\n");
   gets(st);

   stringrev(st);    // calling our reverse function

   printf("Reversed String ::%s\n", st);

   }
void stringrev(char *str)
{
    char temp;
    int size = 0, i = 0;

    while (str[size] != '\0')
    {
        size++;
    }
    size = size - 1;
    while (i < size)
     {
        temp = str[i];
        str[i] = str[size];
        str[size] = temp;
        size--;
        i++;
    }

}
```

STRING-CONVERSION FUNCTIONS

<stdlib.h> contain different string-conversion functions.

Following table summarizes the string-conversion functions.

Function	Function Description
strtod()	This function convert string to double
strtol( )	This function convert string to long
strtoul( )	This function convert string to unsigned long

```
#include <stdio.h>
#include <stdlib.h>
int main( void )
{
const char *string = "51.2% Passed"; // initialize string
double d; // variable to hold converted sequence
char *stringPtr; // create char pointer
d = strtod( string, &stringPtr );

printf( "double value: %.2f and the string: %s", d, stringPtr );
} // end main
```

In the above program

```
d = strtod( string, &stringPtr );
```

Indicates that **d** is assigned the double value converted from string, and **stringPtr** is assigned the location of the first character after the converted value (51.2) in string.