Strings in C language are an array of characters ended with null characters (‘\0’). The null character at the end of a string indicates its end and the strings are always enclosed by double quotes. In C language characters are enclosed by single quotes. Some examples of both of them are shown below.
String functions are usually used to manipulate the strings.

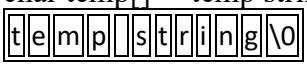
Example or Representation of C Characters and Strings

- char string [10] = {‘s’,’d’,’f’,’d’,’t’,’j’,’a’,’\0’};
- char string [10] =” fresher”;
- char string [] =” fresher”;

There is a minor difference between the declarations of the strings in both of the above statements. Like when we declare char as string [10], 10 bytes of memory space gets allocated to hold the 10 values of string, while when we declare it like string [] then memory gets allocated at the time of execution of the program.
To find the length of any string, manual programming can be done but it could be a time-consuming task, rather one can use the string functions directly to save time and effort.

String Declaration and Initialization

In C programming, the strings can be declared in two ways as shown above. In C programming, a string is a sequence of characters that are terminated with a null or ‘\0’ character. An example of the same is given below:
char temp[]=” temp string”;
When a string of character is declared of char type that is enclosed in a double quotation mark, then \0 is automatically appended at the end of the string. For example –
char temp[]=” temp string”;

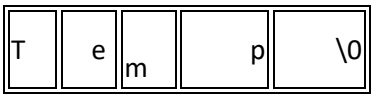


String Declaration
A string in language C is declared in the following manner:
char temp[5];

In this way, we can initialize a string of length 5.
To declare a string in C, a data array should be used because C does not support string as a data type. While declaring a C string, the size of a variable must be defined for it to calculate the number of characters going to be stored inside the string variable in C.

String Initialization

String initialization can be done in many ways and some of them are given below:
char t[]=” temp string”;
char t[10]=” temp string”;
char t[]={‘t’,’e’,’m’, ‘p’,’\0’};
char t[5]={‘t’,’e’,’m’, ‘p’,’\0’};



t[0] t[1] t[2] t[3] t[4]

In the above type of declaration, we can only store the strings that have only four characters. While if you want to store five characters in such string or array, then you may need the character array of more length.

C language allows initialisation of string variables without defining the string array. It can be done in the following way –

char color_name [ ] = “BLUE”;

Assigning Values to Strings

Arrays and strings do not support the assignment operators. Once the strings are declared you cannot assign the values to string-type variables. For example in C language we cannot write and assign the values in the following way:

char t[100];

t=” temp value”;

Strings are copied by value, not reference. String assignment happens using the =operator followed by the copied actual bytes from the operand up source. The new type string variable can be created by assigning it an expression of the type string.

String Handling in C

Now we are going to enlist some of the popular string functions in C that make string handling quite easier. Multiple operations on the string like reading the String, Copying or Reversing the String and many other operations can be easily performed on the strings by using these functions. String operators or string functions can be used directly to manipulate the strings. Here, in this article we will explain the library functions like gets(), puts(), strlen(), strcopy(), and many others to explain string handling in C.

Sometimes programmers have to write the string functions to manipulate them as per the required problem. However, string manipulation can be done manually, but this can make the programming quite complex and large.

There are predefined string functions in the C language, namely string handling functions. There is a header file defined for these functions named as string.h. While string handling it is important to use a header file of string.h while doing string handling C.

List of some Common String Handling Functions in C

Function	Description
strlen()	Can compute the length of the string
Strcpy()	Can copy the content of a string to another
Strcat()	Is used to concatenate or join two strings
Strcmp()	Can compare two strings
Strlwr()	Can convert the string to lowercase
Strupr()	Is used to convert the letters of string to uppercase
Strrev()	Is used to reverse the string

When you have to use any of the string handling functions in your program, the functions are not limited only to these many. There are many other string functions as well. So, let's discuss them:

1) puts() and gets()

The two popular functions of string header file gets and puts are used to take the input from the user and display the string respectively.To understand briefly the working of the string handling functions in c of puts and gets, the gets() function, allows the ensure to enter characters followed by enter key. And it also enables the user to add spaced separated strings. Whereas, the puts() function is also one of the types of strings in C, is used for writing a line for the output screen. It is similar to the printf() function

Both of these functions are defined in string.h file. Let's see one example of these functions:

```
#include main()
Int main()
{
char temp[20];
printf("Enter your Name");
gets(temp);
printf("My Name is: ");
puts(temp);
return 0;
}
```

2) strcat()

For the cases when one string has to be appended at the end of another string, this function is being used. Function strcat can append a copy of the source string at the end of the destination string. The strcat() is one of the string operations in c which concatenates two strings, meaning it joins the character strings end-to-end. In the strcat() operation, the destination string's null character will be overwritten by the source string's first character, and the previous null character would now be added at the end of the new destination string which is a result of stcrat() operation.

The user has to pass two arguments that are described below:

i) src

ii) dest

Here at the place of "src" string is specified, while at the place of 'dest' the destination string in which we have to append the source string is specified.

Example

```
#include<string.h>
int main()
{
char src[20]= " before";
char dest[20]= "after ";
strcat(dest, src);
puts(dest);
return 0;
}
```

The output will be: after before

3) Function strlen()

One more function of string header file that can be directly used for the strings is strlen(). You can use the function strlen(), the **string function in C**, when you have to find out the length of any string. The strlen() string functions in c basically calculate the length of a given string. However, one can also write a program manually to find out the length of any string, but the use of this direct function can save your time and the example is given below:

```
#include<stdio.h>
int main()
{
int length;
char s[20] = “We are Here”;
length=strlen(s);
printf(“\Length of the string is = %d \n”, length);
return 0;
}
```

Length of the string is = 11

4) Function strcpy()

If you have to copy the content of one string to another string, then this function is being used. Even the null characters are copied in the process. Syntax of the function is strcpy(dest,source). The function can copy the content of one string to another. One example of the function is given below:

```
#include<string.h>
int main()
{
char src[20]= “ Destination”;
char dest[20]= “”;
printf(“\n source string is = %s”, src);
printf(“\n destination string is = %s”, dest);
strcpy(dest, src);
printf (“\ntarget string after strcpy() = %s”, dest);
return 0;
}
```

Output

Source string is = Destination

Target string is =

Target string after strcpy() = Destination

5) Function strcmp()

To compare two strings to know whether they are same or not we can use strcmp() function.This string functions in c, compares two strings. While comparing the strings takes two parameters into account namely –

1. str1
2. str2

On comparing the return value be determined basis the strings setup as shown below.
The function returns a definite value that may be either 0, >0, or <0. In this function, the two values passed are treated as case sensitive means 'A' and 'a' are treated as different letters. The values returned by the function are used as:
i) 0 is returned when two strings are the same
ii) If str1<str2 then a negative value is returned
iii) If str1>str2 then a positive value is returned

Example:

```
#include<stdio.h>
#include<string.h>
int main()
{
char str1[]=”copy”;
char str2[]=”Trophy”;
int I,j,k;
i=strcmp(str1, “copy”);
j=strcmp(str1, str2);
k-strcmp(str1, “f”);
printf(“\n %d %d %d”,I,j,k);
return 0;
}
```

Output: 0 -1 1

6) Functions strlwr() / strupr()

Sometimes you may need to convert the lowercase letters of any string to the uppercase or vice-versa. As it can be understood the lwr stands for lowercase and upr stands for uppercase. For this purpose there are two direct **string functions in C**, they can be used to perform the conversions either from upper to lower case or vice-versa. Here, we have explained an example of the same:

```
#include<stdio.h>
#include<string.h>
int main()
{
char str[]=”CONVERT me To the Lower Case”;
printf(“%s\n”, strlwr(str));
return 0;
}
```

Output: convert me to the lower case

Similarly, if we will use the strupr function in place of strlwr, then all the content will be converted to the upper-case letters. We can use the strupr function, defined in the string header file. Through this function, all letters of the string are converted, that too without any long manual procedure.

7) Function strrev()

If you want to reverse any string without writing any huge or extensive program manually, then you can use this function. The rev in the strrev() stands for reverse and it is used to reverse the given string. Function strrev() is used to reverse the content of the string. Strrev function is used to check the nature of the string, whether the given string is a palindrome or not. Several other uses and applications are also present in the string reverse function. One of its uses is given below:

```
#include<stdio.h>
#include<string.h>
int main()
{
char temp[20]=”Reverse”;
printf(“String before reversing is : %s\n”, temp);
printf(“String after strrev() :%s”, strrev(temp));
return 0;
}
```