7: STRINGS

Strings are actually one-dimensional array of characters terminated by a **null** character '\0'. Thus a null-terminated string contains the characters that comprise the string followed by a **null**.

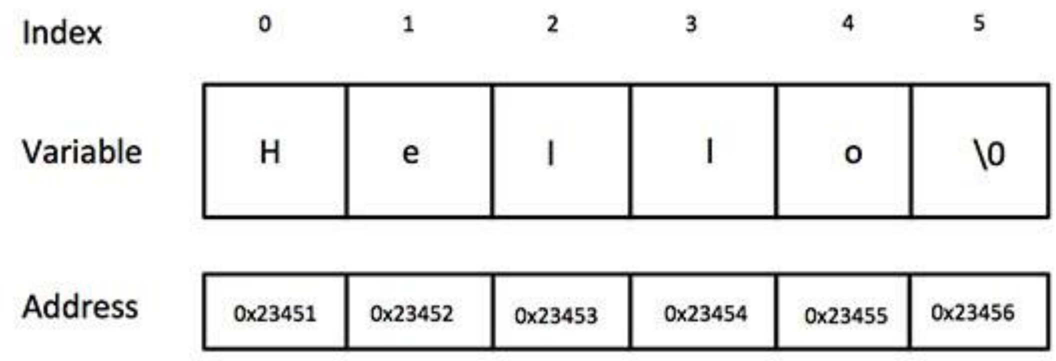
The following declaration and initialization create a string consisting of the word "Hello". To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word "Hello."

```
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
```

If you follow the rule of array initialization then you can write the above statement as follows −

```
char greeting[] = "Hello";
```

Following is the memory presentation of the above defined string in C/C++ −



Actually, you do not place the *null* character at the end of a string constant. The C compiler automatically places the '\0' at the end of the string when it initializes the array. Let us try to print the above mentioned string −

```c
#include <stdio.h>


int main () {


   char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
   printf("Greeting message: %s\n", greeting );
   return 0;
}
```

When the above code is compiled and executed, it produces the following result –

```
Greeting message: Hello
```

1Declaring and initializing strings

Declaring And Initializing String Variables

C does not support string as a data type. However, it allows us to represent strings as character arrays. In C therefore, a string variable is any valid C variable name and is always declares as an array of characters. The general form of declaration of a string variable is:

char string_name[size];

The size determine the number of characters in the string_name.Example:

char girl[10];

char name[30];

When the compilier assigns a character string to a character array, it automatically supplies a null character('\0') at the end of the string. Therefore, the size should be equal to the maximum number of characters in the string plus one.

Like numeric arrays, character arrays may be initialized when they are declared. C permits a character array to be initialized in either of the following two forms:

char city[9] = "NEW YORK";

char city[9] = {'N', 'E', 'W', ' ', 'Y', 'O', 'R', 'K', '\o'};

Following are some of the useful string handling functions supported by C.

1. strlen()
2. strcpy()
3. strncpy()
4. strcat()
5. strncat()
6. strcmp()
7. strncmp()
8. strcmpi()
9. strncmpi()

[1] strlen()
strlen() function returns the length of the string. strlen() function returns integer value

```
char *str = "Learn C Online";
int strLength;
strLength = strlen(str);    //strLength contains the length of the
string i.e. 14
```

[2] strcpy()
strcpy() function is used to copy one string to another. The Destination_String should be a variable and Source_String can either be a string constant or a variable.
Syntax:
strcpy(Destination_String,Source_String);
Example:

```
char *Destination_String;
char *Source_String = "Learn C Online";
strcpy(Destination_String,Source_String);
printf("%s", Destination_String);
```

Output:

```
Learn C Online
```

[3] strncpy()
strncpy() is used to copy only the left most n characters from source to destination.
The Destination_String should be a variable and Source_String can either be a string constant or a variable.
Syntax:
strncpy(Destination_String, Source_String,no_of_characters);

[4] strcat()
strcat() is used to concatenate two strings.
The Destination_String should be a variable and Source_String can either be a string constant or a variable.
Syntax:
strcat(Destination_String, Source_String);
Example:

```
char *Destination_String ="Learn ";
char *Source_String = "C Online";
strcat(Destination_String, Source_String);
puts( Destination_String);
```

Output:

```
Learn C Online
```

[5] strncat()
strncat() is used to concatenate only the leftmost n characters from source with the destination string.
The Destination_String should be a variable and Source_String can either be a string constant or a variable.

Syntax:
strncat(Destination_String, Source_String,no_of_characters);
Example:

```
char *Destination_String="Visit ";
char *Source_String = "Learn C Online is a great site";
strncat(Destination_String, Source_String,14);
puts( Destination_String);
```

Output:

```
Visit Learn C Online
```

[6] strcmp()
strcmp() function is use two compare two strings. strcmp() function does a case sensitive comparison between two strings. The Destination_String and Source_String can either be a string constant or a variable.
Syntax:
int strcmp(string1, string2);
This function returns integer value after comparison.
Value returned is 0 if two strings are equal.
If the first string is alphabetically greater than the second string then, it returns a positive value.
If the first string is alphabetically less than the second string then, it returns a negative value

```
char *string1 = "Learn C Online";
char *string2 = "Learn C Online";
int ret;
ret=strcmp(string1, string2);
printf("%d",ret);
```

Output:

```
0
```

[7] strncmp()
strncmp() is used to compare only left most ‘n’ characters from the strings.
Syntax:
int strncmp(string1, string2,no_of_chars);
This function returns integer value after comparison.
Value returned is 0 if left most ‘n’ characters of two strings are equal.
If the left most ‘n’ characters of first string is alphabetically greater than the left most ‘n’ characters of second string then, it returns a positive value.
If the left most ‘n’ characters of first string is alphabetically less than the left most ‘n’ characters of second string then, it returns a negative value
Example:

```
char *string1 = "Learn C Online is a great site";
char *string2 = "Learn C Online";
int ret;
ret=strncmp(string1, string2,7);
printf("%d",ret);
```

Output:

```
0
```

[8] strcmpi()
strcmpi() function is use two compare two strings. strcmp() function does a case insensitive comparison between two strings. The Destination_String and Source_String can either be a string constant or a variable.
Syntax:
int strcmpi(string1, string2);
This function returns integer value after comparison.
Example:
```
char *string1 = “Learn C Online”;
char *string2 = “LEARN C ONLINE”;
int ret;
ret=strcmpi(string1, string2);
printf("%d",ret);
```
Output:
```
0
```

[9] strncmpi()
strncmpi() is used to compare only left most ‘n’ characters from the strings. strncmpi() function does a case insensitive comparison.
Syntax:
int strncmpi(string1, string2,no_of_chars);
This function returns integer value after comparison.
Example:
```
char *string1 = "Learn C Online is a great site";
char *string2 = "LEARN C ONLINE";
int ret;
ret=strncmpi(string1, string2,7);
printf("%d",ret);
```

Output:
```
0
```

Give the syntax of strcat ( ) string function.

Ans: Syntax:

strcat(str1,str2);

Concatenates str1 and str2 and resultant string is stored in str1.

Write a program to copy one string into another and count the number of character copied.

Ans:

```
 #include <stdio.h>
#include <string.h>
//#include<conio.h>
void main()
{
char src[25],dest[25];
int count_char=0,int i;
clrscr();
printf("\nEnter the String which is to be copied to another String:");
gets(src);
strcpy(dest,src);
printf("\nCopied String is: %s",dest);
for(i=0; dest[i]!=‟\0‟; i++)
{
count_char++;
}
printf("\nNumber of characters in string : %d",count_char);
getch();
}
```

Output

```
Enter the String which is to be copied to another String: abc
Copied String is: abc
Number of characters in string : 3
```

Explain the use of the following function with syntax:

i) **strcmp ( ) ii) strlen ( )**

i) strcmp():-

This function compares two strings identified by arguments and returns zero if both strings are equal, otherwise it returns the difference between ASCII values of first non matching character pair from the strings.

Syntax:

strcmp (string1, string2);

strcmp ("there", "their");

a (difference between „r„ & „e„)

ii) strlen():-

It can be used to get the length of the string. It uses string argument which is name of character array. It returns an integer showing no. of characters from the string excluding last„\0„ (null character).

Syntax:

strlen(string1);

Example:

int n; n=strlen("abc");

then n=3

Give the syntax of strcat ( ) string function.

Ans: Syntax:

strcat(str1,str2);

Concatenates str1 and str2 and resultant string is stored in str1.

State the use and syntax of strcpy( ) function.

strcpy() function:

use :

It is a built in string library function which is used to copy one string
to another.

Syntax:

strcpy(str1,str2);

it copies contents of str2 to str1.

Explain strlen() and strcmp() string functions with examples.

strlen()-This library function is used to count the length of the string i.e. number of characters including blank spaces.

Syntax : strlen(string1);

Example :

int i;

char string1[]="abc";

i=strlen(string1);

strlen function counts number of characters from string1 and stores
the count in the variable i.

strcmp( ): This library function is used to compare two strings which are passed as arguments to it. If the strings are equal then function returns value as 0 and if they are not equal then the function returns ASCII value difference of the first mismatched characters from the string.

Syntax: strcmp(string1,string2);

Example:

Consider str1=”abc” and str2=”abc”

i=strcmp(str1,str2)

strcmp function compares characters from str1 and str2 and returns 0 as both the strings are same.

Write a program in “c‟ to copy one string into other without using built in function.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i;
char str[20];
char dest[20];
clrscr();
printf("Enter a string");
scanf("%s",&str);
for(i=0;str[i]!='\0';i++)
{
dest[i]=str[i];
}
dest[i]='\0';
printf("The source string is %s",str);
printf("\nThe copied string is %s",dest);
getch();
}
```