STRINGS :

Definition :

→ In C language the group of characters, digits and symbols enclosed within quotation marks are called as string.

→ The string is always declared as character arrays. In other words character arrays are called strings.

→ To manipulate text such as words and sentences normally strings are used.

→ Every string is terminated with '\0' (NULL) character. The NULL character is a byte with all bits at logic zero. Hence, its decimal value is zero.

▨ Why we need a terminating null character ?

→ As we know, a string is not a data type in C, but it is considered a data structure stored in an array.

→ The string is a variable length structure and is stored in a fixed-length array.

→ The array size is not always the size of the string and most often it is much larger than the string stored in it.

→ Therefore, the last element of the array need not represent the end of the string.

→ we need some way to determine the end of the string data and the null character serves as the "end-of-string" marker.

Declaration of String :

→ 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 declared as an array of characters.

→ The general form of declaration of a string variable is ;

```
char string-name [size] ;
```

Here the size determines the no. of characters in the string name.

→ Example : char name[10] ;

when the compiler 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.

Initialization of String :

→ Like, numeric arrays, character array may be initialized when they are declared.

→ General form of initialization of a string variable is ;

```
   char string-name [size] = " list of characters" ;
or char string-name[size] = { list of characters } ;
```

→ Example : char city [9] = "NEW YORK" ;
char city[9] = {'N', 'E', 'W', 'Y', 'O', 'R', 'K', '\0'} ;

→ C also permits to initialize a character array without specifying the number of elements. In such cases, the size of the array will be determined automatically, based on the no. of elements initialized.

For example, the statement

`Char string [] = {'G', 'O', 'O', 'D', '\0'};` defines the array string as a five element array.

→ we can also declare the size much larger than the string size in the initializer. i.e. the statement

`char string [10] = "GOOD";` is permitted.

In this case, the computer creates a character array of size 10, places the value "GOOD" in it, terminates with the null character, and initializes all other elements to NULL.

G	O	O	D	\0	\0	\0	\0	\0	\0

▨ Char name [6] = {'M', 'A', 'M', 'A', 'T', 'A'};

In this case the output will not be MAMATA, but it contains some garbage value like MAMATA ôŭ*, because arguments in this examples are initialized with [6], which is exactly equal to the number of characters inside the braces.

The NULL character must be included and hence the argument must be [7] instead of [6] like;

Char name[7] = {'M', 'A', 'M', 'A', 'T', 'A'};

Reading the strings :

→ The scanf() function can be used with %S format to read a String.

→ For example : Char name[11] ;
  Scanf ("%S", name) ;

→ However, the scanf() has got a limitation that it terminates its input on the first white| blank space it encounters. In above example, if the name is "C programming" then only C will be read into the array name, since C is followed by a blank space In such case two character arrays must before reading the entire text C programming as ;

```
Scanf ( "%S%S", fname, lname) ;
```

→ Here the string C is assigned to fname, and programming is assigned to lname.

→ To overcome this difficulty, a function similar to scanf() called gets() is used. Since gets() is a function, it requires a set of parentheses as ;

```
Char a[80] ;
gets(a) ;
```

when executed, the gets(a) accepts a string length of 80 characters. The gets() function can be terminated by pressing a return key.

→ There is one more function similar to scanf() called getchar(), which will read only a single character from the keyboard unlike gets() which can read the entire String until terminated by a return key.

For example :
```
Char a ;
a = getchar() ;
```

Here getchar() accepts a character from the keyboard and is assigned to the variable a.

use of gets() Function :

```
#include <stdio.h>
#include <conio.h>
void main()
{
  Char name[50];
   Clrscr();
   printf("Enter name :\n");
    gets (name);
    printf(" Entered name is : %s\n", name);
} getch();
```

output : Enter name : Mamata
Entered name is : Mamata.

Writing the Strings :

→ The printf() function is used with %s format to print the strings to the screen.

For example : Char name [11];
printf("%s", name);

This printf() function will display the entire contents of the array name.

→ similar to the scanf() function, the printf() function too has its own limitations and so we have two functions putchar() and puts() to write characters. There are similar to getchar() and gets().

use of putchar() function :

```
#include <stdio.h>
# include <conio.h>
void main()
{
  char ch ;
  clrscr();
  printf("Enter a character:\n"),
  ch = getchar();
  printf("Entered character is :\n");
   putchar (ch);
 }
```

<u>output</u>:
```
Enter a character : a
Entered character is : a
```

The function <u>putchar()</u> will <u>display</u> the character on a screen i.e. <u>only one character</u>

use of puts() function :

→ The function <u>puts()</u> is used to <u>display the entire string</u>.

→ The function puts() is an extension of printf() function i.e. it is a <u>combination of</u> <u>printf()</u> with a <u>new line character</u>. In printf() we use new line character '\n to skip to the next line whereas in puts() it will automatically skip to the next line after printing the message on the screen.

```
#include <stdio.h>
# include <conio.h>
void main()
{
   char name[80],
   clrscr(),
   puts("Enter a string:"),
gets (name);
puts(" Entered string is :");
puts (name);
```

output :
```
Enter a string :
Bhubaneswar
Entered string is :
Bhubaneswar
```

/* write a program to find the length of the string */

```
# include <stdio.h>
# include <conio.h>
void main()
 {
   char name[80]; int i=0;
   clrscr();
   printf(" Enter the string :\n");
   gets (name);
   while ( name[i] != '\0')
         i++;
   printf(" The length of the string is : %d", i);
   getch();
 }
```

output :
```
Enter the string :
Bhubaneswar
The length of the string is :
11
```

```
/* write a program to calculate Concatenate two strings */
#include <stdio.h>
# include <conio.h>
void main()
 {
    char name1[90], name[80] ;
    int i, j ;
    clrscr();
   printf ( " Enter First string : \n");
    gets (name);
   printf (" Enter second string :\n");
   gets (name1).
   i = 0;
  j = 0;
  while ( name[i] ! = '\0')
        i++ ;
  while ( name1[j] ! = '\0')
       name [i++] = name1[j++] ;
          name [i] = '\0' ;
  printf (" Concatenated string is : %s\n", name);
  getch();
 }
```

output :

```
        Enter First String : Bhubaneswar
        Enter second string : Cuttack
      Concatenated string is : BhubaneswarCuttack
```

```
/* write a program to reverse a string */
#include <stdio.h>
#include <conio.h>
void main()
{   Char name[80], name1[80];
    int i, j;
    clrscr();
    printf("Enter a string :\n");
    gets (name);
     i = 0;
     j = 0;
  while ( name[i] != '\0')
       i++ ;
  while (--i >= 0)
       name1[j++] = name[i];
    name1[j] = '\0' ;
printf(" The Reversed string is : %s", name1);
 getch();
}
```

output :
```
Enter a string :
      Bhubaneswar
   The Reversed String is :
      rawsenabuhB
```

```
/* write a program to check whether a given string is palindrome
   or not */

#include <stdio.h>
# include <conio.h>
void main()
{
   char string[15], flag ;
   int i, j ;
   clrscr();
   printf (" Enter the string :\n");
   gets (string);
   flag = 'Y' ;
   printf ( " The given string \"");
  for ( i=0 ; string[i] ! = '\0' ; i++)
        printf (" %c", string[i]),
   printf (" \"" );
  i = i-1 ;
for ( j=0 ; i>j , j++ , i--)
   {   if ( string[i] ! = string[j])
           {
              flag= 'n' ;
              break ;
           }
     }
 if ( flag = = 'Y')
        printf ( " is a palindrome \n") ;
 else   printf ( " is not a palindrome \n") ,
    getch() ,
}
```

output : Enter the string : Madom
The given string "Madam" is a palindrome .

/* write a program to copy one string to another string */

```
#include <stdio.h>
# include <conio.h>
void main()
{
  char name[80], name1[80];
   int i ;
   clrscr();
   printf(" Enter a string to copy :\n");
    gets(name);
    i=0 ;
  while ( name[i] != '\0')
       name1 [ i] = name[i++] ;
           name1 [i] = '\0' ;
  printf ( " Output String is : %s \n", name1);
  getch();
}
```

output : Enter a string to copy :
Bhubaneswar

Output String is :
Bhubaneswar

/* write a program to compare two strings */

```
#include <stdio.h>
# include <conio.h>
void main()
{
   char name[80], name1[80] ;
    int i ;
   clrscr();
  printf(" Enter the First string\n");
    gets (name);
  printf(" Enter the second string\n");
    gets (name1);
    i=0;
 while( name[i] = = name1[i] ?? name1[i] ! = '\0')
      i++ ;
if ( name[i] = = name1[i])
        printf(' Two strings are equal\n"),
 else   printf(' Two strings are not equal");
 getch();
}
```

output :

```
Enter the First string
      Bhubaneswar
Enter the second string
       Bhubaneswar
Two strings are equal
```

String Handling using Library Functions:

The string library functions are used to handle string operations.

Function	Description
Strlen (S1)	Returns the length of S1
Strcat (S1, S2)	Concatenates S2 onto the end of S1.
Strrev (St1)	Converts string to reverse.
Strcpy(S1, S2)	Copies S2 to S1.
Strcmp(S1, S2)	Returns 0(false) if S1 = S2 Returns less than 0 if S1 < S2 Returns greater than 0 if S1 > S2
Strlwr(S1)	Converts string to lowercase
Strupr(S1)	Converts string to uppercase

Strlen (S1) :

→ The length of the string can be calculated using strlen() library function

→ This function counts and returns the number of character in a string.

→ For example : $\boxed{a = \text{Strlen}(S);}$

a is integer variable that receives the length of the string. The counting ends of the first null character.

/* program to calculate the length of the string using strlen() function */

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
    char name[10];
    int length;
    clrscr();
 printf("Enter the string\n");
    gets(name);
 length = strlen(name);
printf("The length of the string is = %d\n", length);
 getch();
}
```

<u>output</u>:

```
Enter the string  Bhubaneswar
The length of the string is = 11
```

/* <u>Strcat(S1,S2)</u>:

→ By concatenation we mean to add two or more strings and place them in the first string.

→ The function strcat(s1,s2) will append the string s2 to string s1. This is done by removing the null character at the end of s1 and placing s2 from there. The string value s2 remains unchanged.

→ Example: n_1 = "Ram"
n_2 = "Vihar"

Now strcat(n1, n2). will give the output n_1 = Ram Vihar

→ It is also possible to have nesting of strcat() functions the statement ;

```
strcat ( strcat (s1,s2), s3) ;
```

joins all the three strings s1, s2 and s3 together and the result is stored in s1 .

```
/* program for concatenation of two strings using strcat()
   function */
# include <stdio.h>
# include <conio.h>
# include <string.h>
void main( )
{
  Char  source[] = "Garanayak" ;
  Char  dest [] = " Mamata" ;
  clrscr( );
 strcat ( dest, source) ;
 printf (" After string concatenation = %s\n", dest) ;
 getch( );
}
```

<u>output</u> : After string concatenation = Mamata Garanayak

<u>Strrev (st1) :</u>

→ This function is used to check whether the given string is palindrome or not .

→ In this function we calculate the length of the first string. Then last character of the first string is copied to first character of the second string.

→ The Statement strrev(s11) will reverse the string s11.

→ For example if s11 = "mamala" then upon executing the function the resultant string in s11 is "alamam".

```
/* program to check whether a string is palindrome or
   not palindrome */

#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
    Char s1[50], s2[50];
    Clrscr();
  printf("Enter the string:\n");
   gets(s1);
  strcpy(s2, s1);
   strrev(s1);
 if ( strcmp(s2, s1) ==0)
       printf("The string is palindrome\n");
 else   printf("The string is not palindrome");
 getch();
}
```

<u>output</u>:

```
Enter the string : Madam
The string is palindrome.
```

Strcpy (S_1, S_2) :

→ This function is used to copy a string to another string.

→ After getting the string from the user, the input string is copied character by character to the destination string till the first encounters the null character.

→ For example : The statement strcpy(S_1, S_2); will assign content of string variable S_2 to S_1 and the array S_1 should have the size to hold the contents of S_2 also.

```
/* program to copy source string to destination string by
   using strcpy() function */

#include <stdio.h>
# include <conio.h>
# include < string.h>
void main()
{
   Char   source[] = "password" ;
   Char  dest[];
   Clrscr();
   strcpy ( dest , source) ;
  printf ("\nSource string = %s", source);
  printf (" \n Destination string = %s\n", dest);
  getch();
}
```

Output :

```
Source string = password
Destination string = password
```

Strcmp(S1, S2) :

→ This function is used to check whether the given two strings are equal or not .

→ The statement Strcmp(S_1 , S_2) will return 0 if $S_1 = S_2$, positive if $S_1 > S_2$ and negative if $S_1 < S_2$.

→ For example : S_1 = " Mamata" and S_2 = "Sarita" and S_1 and S_2 will compared using strcmp( ) function . Then it will return a negative value . That is for M , ASCII value is 77 & for S , ASCII value is 83 . So $S_1 < S_2$, & hence the function strcmp( ) returns a negative value .

```
/* program to illustrate string comparison */
#include <stdio.h>
# include <conio.h>
# include < string.h>
void main( )
{   static char string1[ ] ="Mamata";
    static char string2[ ] = "Badal";

   int a, b, c ;
   clrscr( );
  a = strcmp ( string1 , "Sarita" ) ;
  b = strcmp ( string1 , String2 ) ;
  c = strcmp ( string2 , " Badal" ) ;
 printf ( " a = %d  b = %d  c = %d\n", a, b, c ) ;
 getch( ) ;
}
```

output :

```
        a =       b =      c = 0
```

```
/* write a program to compare two strings. */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
 {
    Char s1[], s2[];
     clrscr();
   printf ("Enter the first string:");
    gets(s1);
  printf (" Enter the second String:\n");
   gets(s2);
 if ( strcmp (s1,s2)==0)
       printf (" Two strings are equal\n");
 else if ( strcmp(s1,s2)<0)
           printf (" s1 is less than s2\n");
    else
         printf (" s1 is greater than s2\n");
 getch();
}
```

<u>output</u>:

```
Enter the first string:
 Bhubaneswar
Enter the second string:
 Bhubaneswar
Two strings are equal.
```

Strlwr() :

→ This is used to convert string to lowercase.

→ The statement strlwr(S1) will convert uppercase characters to lowercase.

→ For example if S1 = "SURAJ" then upon executing the function the resultant string in S1 is "suraj".

```
/* program to converts string into lower case */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
  Char string[] = "MAMATA GARANAYAK"; clrscr();
   printf ("String in lowercase = %s\n", strlwr(string));
 getch();
}
```

output : String in lowercase = mamata garanayak

Strupr() :

→ This function is used to convert string to uppercase.

→ The statement strupr(S1) will Convert ~~uppercase~~ lowercase Characters to uppercase.

→ For example if S1 = "suraj" then upon executing the function the resultant string in S1 is "SURAJ".

```
/* program to convert String into Uppercase */
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
```

```
char string[] = "mamata";
 clrscr();
 printf(" String in uppercase = %s\n", strupr(string));
getch();
}
```

Output : String in uppercase = MAMATA