C-Strings (Character Arrays)

STRING: It is an array of type char.

Syntax for declaration

char <array/string name> [max. number of characters to be stored +1];

The number of elements that can be stored in a string is always n-1, if the size of the array specified is n. This is because 1 byte is reserved for the NULL character '\0' i.e. backslash zero. A string is always terminated with the NULL character.

Example:

```
char str[80];
```

In the above example, str can be used to store a string with 79 characters.

Initializing a string

A string can be initialized to a constant value when it is declared.

```
char str[ ] = "Good";
```

Or

```
char str[]={'G','o','o','d','\0'};
```

Here. 'G' will be stored in str[0], 'o' in str[1] and so on.

Note: When the value is assigned to the complete string at once, the computer automatically inserts the NULL character at the end of the string. But, if it is done character by character, then we have to insert it at the end of the string.

Reading strings with/without embedded blanks

To read a string without blanks cin can be used

```
cin>>str;
```

To read a string with blanks cin.getline() or gets() can be used.

```
cin.getline(str,80);
```

-Or-

```
gets(str);
```

Printing strings

cout and puts() can be used to print a string.

```
cout<<str:
```

Or

```
puts(str);
```

Note: For gets( ) and puts(), the header file <cstdio> (formally stdio.h) has to be included. puts() can be used to display only strings. It takes a line feed after printing the string.

cin	**gets()**
It can be used to take input of a value of any data type.	It can be used to take input of a string.
It takes the white space i.e. a blank, a tab, or a new line character as a string terminator.	It does not take the white space i.e. a blank, a tab, or a new line character, as a string terminator.
It requires header file <iostream>	It requires the header file <cstdio>
Example: char S[80]; cout << "Enter a string:”; cin>>S;	**Example:** char S[80]; cout << "Enter a string:"; gets(S);

cout	puts()
It can be used to display the value of any data type.	It can be used to display the value of a string.
It does not take a line feed after displaying the string.	It takes a line feed after displaying the string.
It requires the header file iostream	It requires the header file cstdio
Example: char S[80] = "Computers"; cout<<S<<S; **Output:** ComputersComputers	**Example:** char S[80] = "Computers"; puts(S); puts(S); **Output:** Computers Computers

Counting the number of characters in a string and printing it backwards

```
#include<iostream>
using namespace std;

int main( )
{
  char str[80];
  cout<<"Enter a string:";
  cin.getline(str,80);
  for(int l=0; str[l]!='\0';l++);  //Loop to find length
    cout<<"The length of the string is : "<<l<<endl ;
  for(int i=l-1;i>=0;i--)    //Loop to display the string backwards
    cout<<str[i];
  return 0;
}
```

Function to count the number of words in a string

```
void count(char S[])
{
  int words=0;
  for(int i=0;S[i]!='\0';i++)
  {
    if (S[i]==' ')
      words++;                    //Checking for spaces
  }
  cout<<"The number of words="<<words+1<<endl;
}
```

Function to find the length of a string

```
int length(char S[ ])
{
  for(int i=0;S[i]!='\0';i++);
     return i;
}
```

Function to copy the contents of string S2 to S1

```
void copy(char S1[ ], char S2[ ])
{
   for(int i=0;S2[i]!='\0';i++)
      S1[i]=S2[i];
   S1[i]='\0';
}
```

Function to concatenate the contents of string S2 to S1

```
void concat(char S1[ ], char S2[ ])
{
   for(int l=0;S1[l]!='\0';l++);
     for(int i=0;S2[i]!='\0';i++)
         S1[l++]=S2[i];
     S1[l]='\0';
}
```

Function to compare strings STR1 to STR2.

The function returns a value>0 if //STR1>STR2, a value<0 if STR1<STR2, and value 0 if STR1=STR2

```
int compare(char STR1[ ],char STR2[])
{
  for(int I=0;STR1[I]==STR2[I] && STR1[I]!='\0'&&STR2[I]!='\0'; I++);
     return STR1[I]-STR2[I];
}
```

To reverse the contents of string S and store it in string Rev

```
void Reverse(char S[], char Rev[])
{
   for(int C1=0; S[C1]!='\0'; C1++);
      C1--;
   for(int C2=0;C1>=0;C2++,C1--)
      Rev[C2]=S[C1];
   Rev[C2]='\0';
}
```

Function to check whether a string S is a palindrome or not

```
int Palin(char S[])
{
   for(int L=0;S[L]!='\0';L++);    //To find length
      for(int C=0;(C<L/2) && (S[C]==S[L-C-1]);C++);
         return (C==L/2)?1:0; //Returns 1 if Palindrome else 0
}
```

Function to change the case of string S to uppercase

```
void Upper(char S[])
{
   for(int i=0;S[i]!='\0';i++)
      S[i] = (S[i]>='a' && S[i]<='z')?(S[i]-32):S[i];
}
```

Function to change the case of string S to lower case

```
void Lower(char S[])
{
   for(int i=0;S[i]!='\0';i++)
       S[i] = (S[i]>='A' && S[i]<='Z')?(S[i]+32):S[i];
}
```

Function to extract n characters from left side of the string and store it in a different string.

Example: 4 characters from ENVIRONMENT=ENVI

```
int SLeft(char S[ ], int n, char result[ ])
{
   for(int l=0;S[l]!='\0';l++);
     if(n<=I)    //characters extracted should be <=length
     {
        for(int i=0;i<n;i++)
            result[i]=S[i];
        result[i]='\0';
        return 1;
     }
     else
        return 0;
}
```

Function to extract n characters from right side of the string and store it in a different string.

Example: 4 characters from ENVIRONMENT=MENT

```
int SRight(char S[ ], int n, char result[ ])
{
   for(int l=0;S[l]!='\0';l++);
     if(n<=I)     //characters extracted should be <=length
     {
        for(int j=0;i=l-n;S[i]!=’/0’;i++,j++)
            result[j]=S[i];
        result[j]='\0';
        return 1;
     }
     else
        return 0;
}
```