Strings

Definition:

- A string is an array of characters stored in consecutive memory location.
- In strings, the ending character (termination of string) is always the null character '\0'.

Declaration:

Syntax:

char stringname[length];

Example:

char str[10];

This example declares an array str which can store at the most 10 characters.

Initialization:

Character array can be initialized in two ways

1. as individual character
2. as single string.

Example:

char str[10]={'h','e','l','l','o'};

char str[10]="hello";

char str[]hello"; //this declaration automatically assigns size of array as 6
// including '\0'

A string of n elements can hold (n-1) characters , as nth character is always '\0'.

The compiler automatically stores null character at the end of the string.

Memory Representation:

Str[0]	Str[1]	Str[2]	Str[3]	Str[4]	Str[5]
H	E	L	L	O	\0
100	101	102	103	104	105

str = & str[0];

Name of array is equivalent to address of 1st character in that array.

String Input and Output:

```
char str[10];
scanf("%c",&str[0]);          // reading character
scanf("%s",str);              // reading string , termination character is space
printf("%c",str[0]);          // printing character
printf("%s",str);             // printing string
```

void gets(string name) and void puts(string name) are two library functions available for reading and writing string respectively.

Example:

```
char  str[10];
gets(str);          //reading string , termination character is null character i.e. '\0'
puts(str);          // writing or printing string.
```

Various operations on characters:

Example	Output	Meaning
X='a'; printf("%d",X);	97	Ascii value of a in integer form
X='z'-1;printf("%c",x);	y	Ascii value of z=122-1=121=y
Ch>='A' && ch <= 'Z'	-	Tests whether character is upper case or not.

Character related Functions: (available in ctype.h)

Below are the few standard library functions that deal with only single characters. It takes a single character as argument. If condition evaluates to be true, it returns a non zero value else it returns a zero value.

Sr. No.	Function	Meaning
1.	isalphanum(c)	Is the character alphanumeric?
2.	isalpha(c)	Is the character alphabet?
3.	isascii (c)	Is the character an ascii character?
4.	isdigit(c)	Is the character a digit?
5.	islower(c)	Is the character a lower case letter?
6.	isupper(c)	Is the character a upper case letter?
7.	isspace(c)	Is the character a space?
8.	isprint(c)	Is the character is printable?
9.	ispunct(c)	Is the character a punctuation?
10.	tolower(c)	It converts upper case character into lower case character.
11.	toupper(c)	It converts lower case character into upper case character.

Various Operation possible on strings

There are various operations that can be performed on strings.

1. To find length of string.
2. To copy content of one string to another.
3. To change the case of the letters in the string.
4. To do encryption and decryption operation with string.(for security purpose)
5. To find out tokens from the string.(for compiler purpose)
6. To do some string manipulations and rearranging the strings.(for networking)
7. To search match (for editing as well as searching purpose).

C Language does not provide direct operations on string. Like str3=str1+str2 ; // not allowed in C.It provides standard library functions for the same, which are defined in the header files string.h , ctype.h, stdio.h

Sr No.	**Function**	**Meaning**
1.	strlen(s1)	Returns length of string s1 excluding '\0'
2.	strlwr(s1)	Converts string s1 to lower case
3.	strcat(s1,s2)	Appends the contents of string s2 to the end of string s1 and terminates s1 with null and return s1.
4.	strncat(s1,s2,n)	Appends 1st n character from string s2 to the end of string s1 and terminates s1 with null and return s1.
5.	strcmp(s1,s2)	Compares s1 and s2 and returns zero on success , -ve if s1<s2 and +ve if s1>s2
6.	strcpy(s1,s2)	Copies contents of string s2 to string s1.
7.	strncpy(s1,s2,n)	Copies 1st n characters from s2 to s1.
8.	strcmpi(s1,s2)	Compares s1 and s2 ignoring the case and returns similar result as strcmp
9.	strncmp(s1,s2)	Compares 1st n characters of strings s1 and s2 and returns same result as strcmp
10.	strupr(s1)	Converts the string s1 to upper case
11.	strchr(s1,ch)	Returns a pointer to the first occurrence of the character ch in s1.
12.	strrchr(s1,c)	Returns a pointer to the last occurrence of the character c in s1.
13.	strstr(s1,s2)	Returns a pointer to 1st occurrence of string s2 in string s1.

14.	strrev(s1)	Returns reverse string of s1.
15.	strtok(s1,s2)	Searches s1 for tokens that are separated by delimiters specified by s2. Returns the pointer to 1st character of 1st token in s1.
16.	strset(s1,c)	Sets all characters in string s1 to character c and quits when null character is detected.
17.	strnset(s1,c,n)	Sets 1st n characters in string s1 to character c and quits when null character is detected. If n>length of s1, then length of s1 replaces n.
18.	sscanf(buff,control string,parameters)	Divide lines into tokens.

Array of Strings:

Table of strings is called as array of strings.

Example:

Char name[3][10]; //Two dimensional array of strings.

name is an array of 3 strings , each containing 10 characters. The total memory required for name is 30 bytes.

Examples:

1. Write a program to accept a string and print it.

Solution:-

```
#include<stdio.h>
#include<conio.h>
void main()
{
  char   s[80];
  clrscr();
  printf("\n Enter a string ");
  gets(s);
  printf("\nEntered string is =  ");
  puts(s);
}
```

2. Write a program to accept a string and print its length using string.h.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
  int len;
  char   str[80];
  clrscr();
  printf("\n Enter a string ");
```

```
  gets(str);
  len = strlen(str);
  printf("\nLength of string is =  %d",len);
}
```

3. Write a program to accept a string and print its length without using string.h.

```
#include<stdio.h>
#include<conio.h>

int length(char *s)
{
    int len=0;
   while(*s  != '\0')
   {
  len++;
  s++;
    }
     return(len);
}
void main()
{
  int len;
  char   str[80];
  clrscr();
  printf("\n Enter a string ");
  gets(str);
  len = length(str);
  printf("\nLength of string is =  %d",len);
}
```