STRINGS in C

- string: array of characters
  terminated by NULL character
- string i/o statement:
  scanf("%s",S) or gets(S);
  printf("%s",S) or puts(S);
- string.h :
  collection of functions for string manipulation
- no standard operators for string assignment and comparisons!
  (remember: strings are arrays!)
- string manipulation is based on lexicographical analysis which
  means lexical order or alphabetical order or dictionary order.

String functions in C with examples

- **What are strings?**

  In simple language STRING'S are nothing but the character array.

  The declaration of string (character array) is much similar to normal array declaration.

  Each string is terminated by '\0' as indication of string termination.So obviously you will

  require an extra byte of memory for storing '\0'. '\0' is single character whose ASCII

  (American Standard Code for Information Interchange) value is 0.

  How to initialize strings (char array)?
  char site[]={'g','o',o','g','l','e'.};
  char Site[]="google";

  How to take Input as string and print Output that string?
  Method:1

-
```
void main()
{
   char name[30];
   printf("Enter your name");
   scanf("%s",name);       //format specifier
   printf("%s",name);
}
```
Method:2
```
void main()
{
    char name[30];
    printf("Enter your name");
    gets(name);         //in-built function
    puts(name);
}
```

Difference between both methods is that the 1st method use's format specifier and second method uses in-built function.

In 1st method if you Enter something like Bill gates it will take only Bill as string and any thing after (space)

will be discarded.

In 2nd method if you enter any space it will be accepted. Output of second method would be Bill gates.

Similarity between both methods is that you don't have to add '\0' character.

scanf(%s) and gets() converts newline character '\n' into '\0'.

Standard C String library functions

1) strlen() :Determines the length of the string
Syntax: int strlen(char array)

Example: length = strlen(s1)

2) strcpy(); Copies a string from a source to destination
Syntax: strcpy (Destination string,source string)
Example:char ds[] = "MIT" ; char sr[] "VIT";
strcpy(ds,sr)
Output: ds = VIT sr = VIT.

3) strncpy(); Copies character of a string to another string upto the specified lenghth
Syntax: strncpy(Destination string, source string, n) where n is the number of character to be copies from source to desitnation
Syntax: strncpy (Destination string, Source string, 2)
Example:char ds[] = "MIT" ; char sr[] "VIT";
strcpy(ds,sr)
Output: ds = VI sr = VIT.

4) strcmp(): Compares characters of two strings (Function discriminates between small and capital letters.
Syntax: strcmp (Destination string, Source string)
Example:char ds[] = "MIT" ; char sr[] "VIT";
Int c = strcmp(ds, sr)
C returns a +ve value indicates ds > sr.
0 value indicates ds and sr are equal
-ve value indicates ds < sr.
Output, c = -1

5) stricmp(): Compares two strings (Function doesnot discriminate between small and capital letters.
Syntax: stricmp (Destination string, Source string)
Example:char ds[] = "vIT" ; char sr[] "VIT";

Int c = stricmp(ds, sr)
C returns a +ve value indicates ds > sr.
0 value indicates ds and sr are equal
-ve value indicates ds < sr.
Output , c= 1

6) strncmp(); Compares between two strings upto the specified length.
Syntax: strncmp (Destination string, Source string, n)
Example:char ds[] = "MIT" ; char sr[] "VIT";
Int c = strncmp(ds, sr,2) //where sr = VI is compared with dr = MI
C returns a +ve value indicates ds > sr.
0 value indicates ds and sr are equal
-ve value indicates ds < sr.
Output, c = -1

7) strnicmp(); Compares between two strings upto the speicified length (ignores the case)
Syntax: strnicmp (Destination string, Source string, n)
Example:char ds[] = "MIT" ; char sr[] "mIT";
Int c = strncmp(ds, sr,2) //where sr = mI is compared with dr = MI
C returns a +ve value indicates ds > sr.
0 value indicates ds and sr are equal
-ve value indicates ds < sr.
Output, c = -1

8) strlwr(): Converts uppercase characters of string to a lower case
Syntax: strlwr(s1)
Example: char s1[] = "ABCDE"
strlwr(s1)
Output, s1 = abcde

9) strupr(): Converts lowercase characters of a string to upper case
Syntax: strupr(s1)
Example: char s1[] = "abcde"
strupr(s1)
Output, s1 = ABCDE

10) strdup():Duplicates a string
Syntax : strdup(s1)
Example : char s1[] = "abcde"
s2 = strdup(s1);
Output, s2 = abcde

11) strchr(); Determines the first occurrence of a given character and returns the remaining string starting from that position
Syntax: strchr(string, character)
Example: char s1[] = "abcde" //char cc = 'c'
strchr(s1,'')
Output, s1 = cde

12) strrchr(): Determinines the last occurrence of a given character in a string.

```
#include <stdio.h>

int main() {
```

```
  char *s;
  char buf [] = "This is a testing";

  s = strrchr (buf, 't');

  if (s != NULL)
    printf ("found a 't' at %s\n", s);

  return 0;
}
```

13) strstr(): Determinines the first occurrence of a given string in a another string

14) strcat(): Appends source string to destination string
Syntax: strccat (Destination string, Source string)
Example:char ds[] = “MIT” ; char sr[] “VIT”;
Char result[];
Result = strcat(ds,sr)
Output, result = MITVIT

15) strncat(): Appends source string of specified length to destination string.
Syntax: strncat (Destination string, Source string,n)
Example:char ds[] = “MIT” ; char sr[] “VIT”;
Char result[];
Result = strncat(ds,sr,1)
Output, result = MITV

16) strrev(): Reverses all characters of a string
Syntax: strrev(s1)

17) strset(): Sets all characters of strings with a given argument or symbol
Syntax: strset(s1,char);
Example: strset(s1,’*’); s1 = abcd
Output, s1 = ****

18) strnset(): Sets specified number of characters of string with a given argument or symbol
Syntax: strnset(s1,char);
Example: strnset(s1,’*’,2); s1 = abcd
Output, s1 = **cd

19) strspn(): Finds upto what length two stings are identical
Strspn(“aaabcde”, “aaamnop”);
Output = 3

20) strpbrk(): Searches the first occurrence of the character in a given string and then it displays the string starting from that character
Char *ptr;
Ptr = Strpbrk(“india is great”, ‘g’);
Output, ptr = great

String functions examples

1) int strlen(char array): This function accepts string as parameter and

return integer i.e the length of String passed to it.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char string[]="spark";
  int len;
  len=strlen(string);
  printf("length of %s is %d\t",string,len);
}
```

Output::length of spark is 5.
Did you notice that strlen() does not include '\n' in string length or else length would be 6.

2) strcpy (Destination string,source string): This function accepts 2 strings as parameter,1st one is destination string and 2nd is source string.This function copies source string to destination string.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char src[]="spark",dest[15];
  strcpy(dest,src);
  printf("%s is copied to dest string\t",dest);
}
```

Output:spark is copied to dest string.

3) strcat (Destination string,source string): This function accepts two,strings source string is appended to the destination string.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char src[]="spark",dest[]="programming";
  strcat(dest,src);
  printf("concatenated string is %s",dest);
}
```

Output:concatenated string is programmingspark

4. Strncat(dest,src,n)

The strncat function concatenates or appends first n characters from src to dest. All characters from src are copied including the terminating null character.

Return Value

The strncat function returns dest.

Example

```
#include <stdio.h>

int main() {
  char string1[20];
  char string2[20];

  strcpy(string1, "Hello");
  strcpy(string2, "Vatsalooo");

  printf("Returned String : %s\n", strncat( string1, string2, 6 ));
  printf("Concatenated String : %s\n", string1 );

  return 0;
}
```

It will proiduce following result:

```
Returned String : HelloVatsal
Concatenated String : HelloHell
```

5**) strrev (string):**This function accepts single string as parameter and reverse that string.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
   char string[]="spark";
   strrev(string);
   printf("reverse string is %s",string);
}
```

Output: reverse string is kraps.

6)int strcmp (string 1, string2): This function compares two strings passed as parameters and returns either +ve number,0,-ve number.

+ve value indicates string1 > string2.
0 indicates string1 and string2 are equal
-ve value indicates string1 < string2.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char string1[]="spark",string2[]="programming";
  int cmp;
  cmp=strcmp(string1,string2);
  if(cmp>0)
    printf("%s > %s",string1,string2);
  else
  {
  if(cmp<0)
     printf("%s < %s",string1,string2);
  else
 printf("%s = %s",string1,string2);
  }
}
```

Output: spark > programming.
.this is because alphabetically p comes first then s.so the string compare function returns difference between ascii of s and p which would be +ve.

7) strcmpi (string 1, string2): This function is similar to strcmp().The onlyy difference is that it ignores the case.example SparK and spark both are same.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char string1[]="spark",string2[]="SPArk";
  int cmp;
  cmp=strcmpi(string1,string2);
  if(cmp>0)
    printf("%s > %s",string1,string2);
  else
  {
  if(cmp<0)
     printf("%s < %s",string1,string2);
  else
 printf("%s = %s",string1,string2);
```

```
  }
}
```

Output: spark = SPArk.

8. strncmp(dest,scr,n)

The strncmp function compares first n characters of string1 and string2 and returns a value indicating their relationship.

Return Value

- if Return value if < 0 then it indicates string1 is less than string2
- if Return value if > 0 then it indicates string2 is less than string1
- if Return value if = 0 then it indicates string1 is equal to string1

Example

```
#include <stdio.h>

int main() {
  char string1[20];
  char string2[20];

  strcpy(string1, "Hello");
  strcpy(string2, "Hellooo");
  printf("Return Value is : %d\n", strncmp( string1, string2, 4));

  strcpy(string1, "Helloooo");
  strcpy(string2, "Hellooo");
  printf("Return Value is : %d\n", strncmp( string1, string2, 10));

  strcpy(string1, "Hellooo");
  strcpy(string2, "Hellooo");
  printf("Return Value is : %d\n", strncmp( string1, string2, 20));

  return 0;
}
```

It will proiduce following result:

```
Return Value is : 0
Return Value is : 111
Return Value is : 0
```

- **9) strlwr (string):**:This function accepts single string that can be in any case(lower or upper).It converts the string in lower case.

  Example

```
#include <stdio.h>
#include <string.h>
void main(void)
```

```
{
  char string1[]="SPArk";
  strlwr(string1);
  printf("%s is in lower case",string1);

}
```
Output: spark is in lower case.

10) strupr (string)::This function accepts single string that can be in any case(lower or upper).It converts the string in upper case.

Example

```
#include <stdio.h>
#include <string.h>
void main(void)
{
  char string1[]="SPArk";
  strupr(string1);
  printf("%s is in upper case",string1);

}
```

Output: SPARK is in upper case.

11)char* strstr (main string,substring): This function accepts two strings i.e main string and substring.
It searches for the first occurrence substring in main string and returns the character pointer to the first char.

Example

```
#include <stdio.h>
#include <string.h>
void main()
{
      char str1[]="programmingspark",str2[]="ming",*ptr;
      ptr=strstr(str1, str2);
      printf("substring is: %s",ptr);
      getch();
}
```

Output : substring is mingspark

Character conversions and testing: ctype.h

We conclude this chapter with a related library #include <ctype.h> which contains many useful functions to convert and test *single* characters. The common functions are prototypes as follows:

Character testing:

int isalnum(int c) -- True if c is alphanumeric.
int isalpha(int c) -- True if c is a letter.
int isascii(int c) -- True if c is ASCII .

int iscntrl(int c) -- True if c is a control character.
int isdigit(int c) -- True if c is a decimal digit
int isgraph(int c) -- True if c is a graphical character.
int islower(int c) -- True if c is a lowercase letter
int isprint(int c) -- True if c is a printable character
int ispunct (int c) -- True if c is a punctuation character.
int isspace(int c) -- True if c is a space character.
int isupper(int c) -- True if c is an uppercase letter.
int isxdigit(int c) -- True if c is a hexadecimal digit

Character Conversion:

int toascii(int c) -- Convert c to ASCII .
tolower(int c) -- Convert c to lowercase.
int toupper(int c) -- Convert c to uppercase.

The use of these functions is straightforward and we do not give examples here.

Find Length of String Using Library Function

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str[100];
int len;

printf("\nEnter the String : ");
gets(str);

len = strlen(str);

printf("\nLength of Given String : %d",len);
getch();
}
```

Program to Find Length of String Without using Library Function

```
#include<stdio.h>
#include<conio.h>
void main()
{
 char str[100];
 int length;

 printf("\nEnter the String : ");
 gets(str);

 length = 0;  // Initial Length
 i=0;

 while(str[i]!='\0')
{
length=length+1;
i++;
}

 printf("\nLength of the String is : %d",length);
 getch();
}
```

Concat Two Strings Using Library Function

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str1[100];
char str2[100];
char str3[100];
int len;

printf("\nEnter the String 1 : ");
gets(str1);

printf("\nEnter the String 2 : ");
gets(str2);

strcpy(str3,str1);
strcat(str3,str2);

printf("\nConcated String : %s",str3);
getch();
}
```

Program : C Program to Concat Two Strings without Using Library Function

Method:1
CONCATINATE TWO STRINGS

```
#include<stdio.h>
#include<conio.h>
void main()
{
char a[25],b[25],c[25];
int i.j;
clrscr();
printf("Enter first string:");
scanf("%s",a);//VIT
printf("Enter second string:");
scanf("%s",b);//SCOPE
for(i=0;a[i]!='\0';i++) I = 3
c[i]=a[i];

for(j=0; b[j]!='\0'; j++) j = 5
c[i+j]=b[j];c=VITSCOPE

c[i+j]='\0';
printf("The concatenated string is:\n%s,c");
 getch();
}
```

OUTPUT :

```
Enter first string:  Net
Enter second string: World

The concatenated string is:  NetWorld
```

Method:2

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

void concat(char[],char[]);

void main()
{
 char s1[50],s2[30];
 printf("n Enter two strings :");
 gets(s1);
 gets(s2);
 concat(s1,s2);
 printf("nConcated string is :%s",s1);
 getch();
}

void concat(char s1[],char s2[])
{
```

```
 int i,j;
 i=strlen(s1);

 for(j=0;s2[j]!='\0';i++,j++)
  s1[i]=s2[j];

 s1[i]='\0';
}
```

Copy One String Into Other Using Library Function

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char str1[100];
char str2[100];

printf("\nEnter the String 1 : ");
gets(str1);

strcpy(str2,str1);
printf("\nCopied String : %s",str2);
getch();
}
```

Program : C Program to Copy One String into Other Without Using Library Function.

```
#include<stdio.h>
#include<conio.h>
void main()
{
 char s1[100],s2[100];
 int i;
 //reading a string and finding its length
 printf("\nEnter the string :");
 gets(s1);
 i=0;

 while(s1[i]!='\0')
 {
  s2[i]=s1[i];
  i++;
 }
 //since the '\0' is not copied

 s2[i]='\0';

 printf("\nCopied String is %s ",s2);
 getch();
}
```

/*Program to accept names of 5 students and display them in alphabetic order*/

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{

char name[5][20]; // to store 5 names each having a max 19 characters


char temp[20]; // to help in exchange of name

int i,j;
clrscr();

for (i=0;i<5; i++)
{
printf ("enter name %d\t",i+1);
scanf ("%s",name[i]);
}

printf("before sorting\n");
for (i=0;i<5; i++)
printf ("%s\n",name[i]);

// sorting namewise


for (i=0; i<4;i++)
for (j=i+1;j<5;j++)
if( stricmp(name[i],name[j])>0)  name[0] sam >  name[1] bala
{
strcpy(temp,name[i]);
strcpy(name[i],name[j]);
strcpy(name[j],temp);
}

printf("after sorting\n");
for (i=0;i<5;i++)
printf("%s\n",name[i]);
}
```

Write a c program to find the occurance of a given character in a given string

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,occ=0;
char str[100],ch;
printf("\n enter string");
```

```
scanf("%s",str);
printf("\n enter character");
scanf("%s",ch);
for(i=0;i[str]!='\0';i++)
{
if(str[i]==ch)
{
occ++;
}
}
printf("\n occurance of %c in %s is %d",ch str occ);
getch();
}
```

What is the difference between a string and an array?

The following are the differences:

- String can hold only char data. Where as an array can hold any data type.
- An array size can not be changed. Where as a string size can be changed if it is a char pointer
- The last element of an array is an element of the specific type. The last character of a string is a null – '\0' character.
- The length of an array is to specified in [] at the time of declaration (except char[]). The length of the string is the number of characters + one (null character).