CHARACTER ARRAYS AND STRINGS

Introduction:

A string is a sequence of characters that is treated as a single data item. Any group of characters defined between double quotation marks is a string constant.

Character strings are often used to build meaningful and readable programs. The common operations performed on character string include:

- Reading and writing strings
- Combining strings together
- Copying one string to another
- Comparing strings for equality
- Extracting a portion of a string

Declaring and initializing String Variables:

C does not support strings as a datatype. It allows to represent strings as a character arrays. In C, therefore a string variable is any valid C variable name and is always declared as an array of characters.

Syntax:

```
char string_name[size];
```

Example:

```
Char str1[3]=”GOOD”;
```

Program :

```
#include <stdio.h>
void main()
{
        char str1[3]= “ GOOD”;
        clrscr();
        printf(‘The value of string is %s”,str1);
        getch();
}
```

Output:

```
The value of string is GOOD
```

Reading Strings from Terminal:

Using Scanf Function:

The familiar input function scanf can be used with %s format specification to read in a string of character.

Example:

```
char address [10];
scanf(“%s”, address);
```

The problem with the scanf function is that it terminates its input on the first white space it finds. A white space includes blanks, tabs, carriage returns, form feeds and new lines.

The scanf function automatically terminates the string that is read with a null character and therefore the character array should be large enough to hold the input string plus the null character.

Program :

```
#include<stdio.h>
void main()
{
        char word1[20], word2[20], word3[20],word4[20];
        printf( “Enter text :\n”);
        scanf( “ %s  %s ”,&word1, &word2);
        scanf( “ %s “ , word3);
        scanf( “ %s “, word4);
        printf( “ \n ” );
        printf( “ Word1 = %s\n Word2 = %s\n”, word1 , word2);
        printf( “ Word3 = %s\n Word4 = %s\n”, word3 , word4);
       getch();
}
```

Output :

```
Enter text :
Namakkal Salem Chennai Coimbatore
Word = Namakkal
Word = Salem
Word = Chennai
Word = Coimbatore
```

We can also specify the field width using the form %ws in the scanf statement for reading a specified number of characters from the input string.

```
scanf( “ %ws “ , name);
```

Here two things may happen:

1.The width w is equal to or greater than the number of characters typed in. The entire string will be stored in the string variable.
2.The width w is less than the number of characters in the string. The excess characters will be truncated and left unread.

Example:

```
scanf(“ %5s “, &name);
```

Program :

```
#include<stdio.h.
void main();
```

```
{
    char name[50];
    clrscr();
    scanf( " %5s ", &name);
    printf( " %s ", name);
    getch();
}
```

Output :

```
Keerthana
Keert
```

Writing Strings to Screen:

Using printf Function:

We already used that printf function with %s format to print strings to the screen . The format %s can be used to display an array of characters that is terminated by the null character.

Example:

```
printf(" %s " , name);
```

Program :

```
#include<stdio.h>
void main()
{
    char  count[10] = " United Kingdom " ;
    printf ( " *12345678910* \n " );
    printf( " %5s \n " , country );
    printf ( " %.3s\n ", country );
    printf ( " %s \n ", country );
    getch();
}
```

Output :

```
*12345678910*
Unite
Uni
United Kingdom
```

String Manipulation:

In C language the group of character, digits and symbols enclosed within quotation marks are called as string otherwise strings are array of characters. Null character ('\0') is used to mark the end of the string.

Example:

```
Char name [ ] ={'b','a','b','u','\0'};
```

Strings Standard Functions:

The 'C' compiler provides the following string handling functions.

strlen () -Used to find length of the string
strcpy() - Used to copy one string to another
strcat() - Used to combine two strings
strcmp() - Used to compare characters of two strings
strlwr() - Used to convert strings into lower case
strupr() - Used to convert strings into upper case
strdup() - Used to duplicate a string
strrev() - Used to reverse a string
strncpy() - Used to copy first 'n' characters of one string into another
strncmp() - Used to compare first 'n' characters of two strings
strcmpi() - Used to compare two strings without regarding the case
strnicmp() - Used to compare first 'n' characters of two string without regarding the case.
stricmp() - Compares two string
strchr() - Determines first occurrence of a given character in a string
strrchr() - Determines last occurrence of a given character in a string
strstr() - Determines first occurrence of a given string in another string
strncat() - Appends source string to destination string upto specified length.
strnset() - Sets specified number of characters of string with a given argument or symbol.
strspn() - Finds upto what length two strings are identical.
strpbrk() - Searches the first occurrence of the character in a given string and then it displays the string starting from that character.

The commonly used string manipulation functions are follows:

The strlen() fuction:

This function is used to count and return the number of characters present in a string.

Syntax:

```
var = strlen(string);
```

Program:

```
#include<stdio.h>
void main()
{
        char name[]="RAJA";
         int len1 , len2;
         len1= strlen(name);
         len2= strlen("LAK" );
         printf(" \n string length of %s is %d",name , len1);
         printf(" \n string length of %s is %d ", "LAK",len2);
         getch();
}
```

Output:

```
String length of RAJA is 4
String length of LAK is 3
```

The strcpy() function:
This function is used to copy the contents of one string to another and almost works like string assignment operator.
Syntax:
strcpy(string1 , string2);

Program :

```
#include<stdio.h>
void main()
{
        char source = “RAJA”;
        char target[10];
        strcpy(target , source );
        printf(“\n Source string is %s”, source);
        printf(“\n Target string is %s “,target);
        getch();
 }
```

Output :

```
Source string is RAJA
Target string is RAJA
```

The strcat() function:
The strcat() function is used to concatenate or combine , two strings together and forms new concatenated string.
Syntax:
strcat(string1 , string2);

Program :

```
#include<stdio.h>
void main()
 {
       char source[] = “Ramesh”;
       char target[10]=”Babu”;
       strcat( source , target)
       printf(“\n Source string is %d”, source);
       printf(“\n Target string is %s”, target);
       getch();
 }
```

Output :

```
Source string is Ramesh Babu
Target string is Babu
```

The strcmp () function:
This function compare two strings to find out whether they are same or different. The two strings are compared character by character until the end of one of the string is reached. If the two strings are identical strcmp() returns a value zero. If they are not equal, it returns the numeric difference between the first non-matching characters.
Syntax:
Strcmp(string1, string2);

Program :

```
#include<stdio.h>
#include<conio.h>
void main()
{
        char name[]="Kalai";
        char name1[]="Malai";
        int i , j , k;
         i= strcmp(name , "Kalai");
         j= strcmp(name1,name);
         k=strcmp(name ,"Kalai mani");
         printf("\n %d %d %d",i,j,k);
         getch();
}
```

Output :
1 1 6

The strrev() function :
The strrev() function is used to reverse a string . This function takes only one argument and return one argument.

Syntax:
strrev(string);

Program :

```
#include<stdio.h>
void main()
{
            char y[30];
             printf("Enter the string :");
             gets(y);
             printf("The string reversed is : %s ", strrev(y));
             getch();
 }
```

Output :

```
Enter the string : book
The string reversed is : koob
```