Strings

Introduction
Reading and displaying strings
Passing strings to function
String handling functions

Introduction

- Strings are array of characters i.e. they are characters arranged one after another in memory. Thus, a character array is called string.
- Each character within the string is stored within one element of the array successively.
- A string is always terminated by a null character (i.e. slash zero \0).

Arrays and Strings…

- Operations performed on character strings include:
  - Reading and writing strings
  - Copying one string to another
  - Combining strings together
  - Comparing strings for equality
  - Extracting a portion of a string

- A string variable is declared as an array of characters.
- Syntax:

  char string_name[size];
- E.g. char name[20];
- When the compiler assigns a character string to a character array, it automatically supplies a *null character* (‘\0’) at the end of the string

Initializing String Variables

- Strings are initialized in either of the following two forms:

  char name[4]={'R','A','M', '\0'};

  char name[]={'R','A','M', '\0'};

OR

char name[4]="RAM";

char name[]="RAM";

R	A	M	\0
name[0]	name[1]	name[2]	name[3]

- When we initialize a character array by listing its elements, the null terminator or the size of the array must be provided explicitly.

Reading and displaying Strings

- It can be done manually.

```c
#include <stdio.h>
int main(){
     char name[30],ch;
     int i=0;
     printf("Enter name: ");
     while(ch!='\n')      // terminates if user hit enter
     {
          ch=getchar();
          name[i]=ch;
          i++;
     }
     name[i]='\0';          // inserting null character at end
     printf("Name: %s",name);
     return 0;
}
```

Using printf() and scanf()

```c
#include <stdio.h>
int main(){
    char name[20];
    printf("Enter name: ");
    scanf("%s",name);
    printf("Your name is %s.",name);
    return 0;
}
```

Using gets() and puts()

```c
int main(){
    char name[30];
    printf("Enter name: ");
    gets(name);      //Function to read string from user.
    printf("Name: ");
    puts(name);     //Function to display string.
    return 0;
}
```

Passing String to function

```c
#include <stdio.h>
void Display(char ch[]);
int main(){
    char c[50];
    printf("Enter string: ");
    gets(c);
    Display(c);       // Passing string c to function.
    return 0;
}
void Display(char ch[]){
    printf("String Output: ");
    puts(ch);
}
```

String handling functions

- Strings need to be manipulated by programmer.
- It can be done manually but is time consuming.

Counting length of the string

```
#include <stdio.h>
#include <conio.h>
void main()
{
char input_string[50];
int i=0, length=0;
clrscr();
printf("\nEnter your text:\t");
gets(input_string);
    while(input_string[i]!='\0')
            {
            length++;
            i++;
            }
printf("\nThe length of your text is: %d character(s)", length);
getch();
}
```

Copying one string to another

```
#include <stdio.h>
#include <conio.h>
void main()
{
char copy[50], paste[50];
int i;
clrscr();
printf("\nEnter your name (to copy):\t");
gets(copy);
    for(i=0;copy[i]!='\0';i++)
            {
            paste[i]=copy[i];
            }
            paste[i]='\0';
printf("\nThe name is (pasted as):\t");
puts(paste);
getch();
}
```

- There are various string handling functions define in string.h some of them are:

Function	Work of Function
strlen()	Calculates the length of string
strcpy()	Copies a string to another string
strcat()	Concatenates(joins) two strings
strcmp()	Compares two string
strlwr()	Converts string to lowercase
strupr()	Converts string to uppercase

```
void main()
{
char input_string[50];
int length;
clrscr();
printf("\nEnter your text:\t");
gets(input_string);
length=strlen(input_string);
printf("\nThe length of your text is: %d character(s)", length);
getch();
}
```

```
void main()
{
char copy[50], paste[50];
int i;
clrscr();
printf("\nEnter your name (to copy):\t");
gets(copy);
strcpy(paste, copy);
printf("\nThe name is (pasted as):\t");
puts(paste);
getch();
}
```

```
void main()
{
char first_name[30]=“College " ;
char middle_name[]=" of Applied";
char last_name[]=" Business";
clrscr();
        strcat(first_name,middle_name);
   puts(first_name);
        strcat(first_name,last_name);
   puts(first_name);
getch();
}
```

```
void main()
{
char str1[30],str2[40];
int diff;
clrscr();
printf("Enter first string:\t");
gets(str1);
printf("\nEnter second string:\t");
gets(str2);
diff=strcmp(str1, str2);
    if(diff>0)
    printf("\n%s is greater than %s by ASCII value difference %d", str1,
    str2, diff);
    else if(diff<0)
    printf("\n%s is smaller than %s by ASCII value difference %d", str1,
    str2, diff);
    else
    printf("\n%s is same as %s", str1, str2);
getch();
}
```

```
void main()
{
char string[25];
clrscr();
printf("\nInput string to be reversed:");
gets(string);
strrev(string);
printf("\nThe reversed string is: %s", string);
getch();
}
```

Arrays of Strings

- String is array of characters.
- Thus an array of string is 2-D array of characters.
- E.g.

```
char names[5][10];
```

- Here, names[5][10] means 5 names having 10 characters each.

Classwork

- WAP to read name of 5 persons using array of strings and display them
- WAP to sort name of 5 persons in alphabetical order

```
void main()
{
char names[5][10];
int i;
clrscr();
printf("\nEnter name of 5 persons:");
for(i=0;i<5;i++)
    scanf("%s", names[i]);

printf("\nThe names are:");
for(i=0;i<5;i++)
    printf("\n%s", names[i]);
getch();
}
```

```
void main()
{
char names[5][10],temp[10];
int i, j;
clrscr();
printf("\nEnter name of 5 persons:");
for(i=0;i<5;i++)
   gets(names[i]);

for(i=0;i<5;i++)
{
      for(j=i+1;j<5;j++)
      {
      if(strcmp(names[i], names[j])>0)
                  {
                  strcpy(temp, names[i]);
                  strcpy(names[i], names[j]);
                  strcpy(names[j], temp);
                  }
      }
}
printf("\nNames in ascending order:\n");
for(i=0;i<5;i++)
      puts(names[i]);
getch();
}
```