STRINGS

String Manipulation Functions

- ✓ All string manipulation functions were defined under string.h header file.
- ✓ The various string manipulation functions are:

1. **strlen:**	Determines length of the string
2. **strcpy:**	Copies content of one to another
3. **strcmp:**	Compares one string to another
4. **strcat:**	Combines two strings
5. **strncpy:**	Copies block of n characters from one to another
6. **strncmp:**	Compare block of n characters from one to another
7. **strncat:**	Specified n characters are appended to the end of another string.
8. **strchr:**	Search specific character in a string
9. **strlwr:**	Converts character to lower case
10. **strupr:**	Converts character to upper case
11. **strrev:**	Reverse the given string

1. strlen

- ✓ The string length function “strlen” is used to find the length of the string in bytes. (i.e., Total number of characters in the string).

Syntax: **int_ variable = strlen(char str[20]);**

- ➢ *The below program shows how the strlen works.*

```
#include<stdio.h>
#include<string.h>
     void main)
     {
          char name[15]=”attitude”;
          int len;
          len=strlen(name);
          printf(“\n The string length is %d”,len);
     }
```

Write a program to find length of string without using library function.

```
#include<stdio.h>
    void main()
    {
          char name[50];
          int i;
          printf("\n Enter string");
          scanf("%s",name);
          i=0;
          while(name[i]!='\0')
                i++;
          printf("The length of the string is %d",i);
    }
```

Note:

- ✓ **sizeof() -**

  The sizeof operator determines the size of a string declared.

  Ex: char name[10]= "gitam"

  sizeof(name)=10

  strlen(name)=4

  Generally the sizeof operator has its use while performing string copy operation.

2. strcpy:

- ✓ Assigning of a string to another string is not possible, so we use function "strcpy" to assign value to a string. This function copies content from source string to destination string.

  Syntax: **strcpy(destination, source);**

- ✓ The strcpy has two parameters, both are pointers to strings.

  It copies characters from source to destination till it finds and copies null character.

  Ex:

```
#include<stdio.h>
#include<string.h>
      void main()
        {
            char src[10]="mother";
            char dest[10];
            strcpy(dest,src);
            printf("destination string=%s",dest);
        }
```

Write a C program to perform strcpy without using library function

```
#include<stdio.h>
      void main()
      {
        char src[10],dest[20];
        int i;
            printf("Enter source string");
            scanf("%s",src);
             i=0;
             while(src[i]!='\0')
             {
             dest[i]=src[i];
             i++;
             }
             dest[i]='\0';
            printf("\n Destination string is %s",dest);
      }
```

3. strncpy:

- ✓ This function copies a substring of one string to another. It copies n characters of source string to destination.

Syntax: **strncpy(dest, source, num_chars);**

Ex: char src[10]="krishna";

char dest[10];

strncpy(dest,src,5);

Write a C program to perform strncpy function

```
#include<stdio.h>
#include<string.h>
    void main()
    {
     char src[20],dest[20];
     int n;
          printf("enter source string");
          scanf("%d";&n);
               strncpy(dest,src,n);
          printf("destination string is %s",dest);
    }
```

4. strcat

- ✓ This function concatenates or joins two strings and the resulting string has only one null character at the end

  Syntax: strcat (first, second) ;

- ✓ The content of second string is appended at the end of first string and at last null inserted

  Ex: char first[20]="kumar";

  char second[20]="ram";

  strcat (first,second);

Write a C program to perform strcat function

```
#include<stdio.h>
#include<string.h>
void main()
{
char first[20],second[20];
   printf("enter first string");
   scanf("%s",first);
   printf("enter second string");
   scanf("%s",second);
     strcat(first,second);
   printf("The string is %s",first);
}
```

Write a C program to perform strcat without using library function.

```
#include<stdio.h>
    void main()
    {
       int i,j;
       char first[50],sec[50];
             printf(“Enter first string”);
             scanf(“%s”,first);
             printf(“Enter second string”);
             scanf(“%s”,sec);
                   i=0;
                   while(first[i]!=’\0’)
                   i++;
                   j=0;
                   while(sec[j]!=’\0’)
                   first[i++]=sec[j++];
                   first[i++]=’\0’;
             printf("The string is %s",first);
    }
```

5. strncat

- ✓ This function copies n characters of second string and joins at the end of first string.

  Syntax: strncat(first,second,n);

  It consists of 3 parameters two strings and an integer value.

Ex:

char first[20] = "god";
char sec[20] = "owner";
strncat(first,sec,3);

Before strncat	***After strncat***
first god	first godown
sec owner	sec owner

6. strcmp

- ✓ String comparison is an extension of character comparison. Here the ASCII value of characters are compared.

  Syntax: int_variable = strcmp(char s1[],char s2[]);

  where,
  s1 is a first string
  s2 is a second string

- ✓ On string comparison strcmp() returns three values:
  - 0 – if both strings are equal.
  - Positive value, if s1>s2.
  - Negative value, if s1<s2.

Write a C program to perform strcmp function

```
#include<stdio.h>
#include<string.h>
void main()
{
char s1[20],s2[20];
int res;
   printf("enter first string");
   scanf("%s",s1);
   printf("enter second string");
   scanf("%s",s2);
      res= strcmp(s1,s2);
   if(res==0)
          printf("%s=%s",s1,s2);
   else if(res>0)
          printf("%s>%s",s1,s2);
   else
          printf("%s<%s",s1,s2);
 }
```

Write a C program to perform string comparison without using library function.

```
#include<stdio.h>
  void main()
      {
      char str1[20],str2[20];
      int i,res;
       printf(“Enter string 1”);
       scanf(”%s”,str1);
       printf(“Enter string 2”);
       scanf(“%s”,str2);
            i=0;
            while(str1[i]==str2[i])
            {
            if(str1[i]==’\0’)
            break;
            i++;
            }
             res = str1[i]-str2[i];
       if(res==0)
       printf(“strings are equal”);
       else if(res>0)
       printf(”first string is greater than second”);
       else
       printf”first string is less than second”);
}
```

7. strncmp

- ✓ This function compares up to n number of characters from the two strings. As in strcmp it returns the integer value representing the relationship between compared string.

Syntax: int_variable = strncmp(char s1[],char s2[],int n);

where,

s1 is a first string

s2 is a second string

n is a number of characters to be compared.

Ex:

```
#include<stdio.h>
   void main()
     {
     char first [20]=”bhima”;
     char second [20]=”bheema”;
          if(strncmp(first,second,4)==0)
               printf(“four characters are same”);
          else if(strncmp(first,second,4)<0)
               printf(“four characters of first string is lesser”);
          else
               printf(“first four characters of first string is
          greater”);
     }
```

8. strrev

- ✓ This function reverses all characters except null characters. The original string will be lost.

Syntax: void strrev (char str[ ]);

Ex:

```
#include<stdio.h>
#include<string.h>
     void main()
     {
          char str[]=”malayalam”;
          strrev(str);
          printf(“string=%s”,str);
     }
```

Write a C program to implement string reverse without using strrev function.

```
#include<stdio.h>
#include<string.h>
   void main()
   {
     char src[20],dest[20];
     int n,i;
         printf(“enter string”;
         scanf(“%s”,src);
          n=strlen(src);
          for(i=0;i<n;i++)
          dest[n-1-i]=src[i]’
          dest[n]=’\0’;
     printf(“\n original string is %s”,src);
     printf(“\n reversed string is %s”,dest);
   }
```

Other Programming Examples

1. Write a C Program to count vowels and consonants in a string.

```
#include<stdio.h>
#include<ctype.h>
#include<string.h>
        void main()
        {
        char str[40], ch;
        int i, vow_count, con_count;
        printf("\n Enter the sentence");
        gets(str);
        vow_count=0;
        con_count_=0;
              for(i=0;i<strlen(str);i++)
              {
                 if(isalpha(str[i]))
                 {
                 ch=tolower(str[i]);
                 if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u')
                 vow_count++;
                 else
                 con_count++;
                 }
              }
        printf("\n The no. of vowels is %d",vow_count);
        printf("\n The no. of consonants is %d",con_count);
       }
```

2. Write a C program to check whether the string is palindrome or not

```
        #include<string.h>
          void main()
              {
              int flag;
              char str[20],rev[str];
              printf("\n Enter the string");
              scanf("%s",str);
                    strcpy(rev,str);
                    strrev(rev);
                    flag=strcmp(str,rev);
              if(flag==0)
              printf("The string is palindrome\n");
              else
              printf("The string is not a palindrome\n");
              }
```

3. Write a C program to convert lower to upper and upper to lower

```
#include<stdio.h>
#include<ctype.h>
#include<string.h>
    void main()
    {
     char str[30],dest[30];
     int i;
     printf(“Enter the sentence”);
     gets(str);
          strcpy(dest,str);
          for(i=0;i<strlen(dest);i++)
          {
                 if(dest[i]>=’a’ && dest[i]<=’z’ )
                 dest[i]=tolower(dest[i]);
                 else
                 dest[i]=toupper(dest[i]);
          }
        printf(“The input sentence is \n”);
        puts(str);
        printf(“The case converted sentence is \n”);
        puts(dest);
    }
```

4. Write a C program to count a character in given string

```
#include<stdio.h>
   void main
   {
    char str[30],ch;
    int i,count;
    printf(“enter string”);
    gets(str);
    printf(“enter key to be search”);
    ch=getchar();
         i=0;
         count=0;
         while(str[i]!=’\0’)
         {
         if(str[i]==ch)
         count++;
         i++;
         }
    printf(“number of chars=%d\n”, count);
   }
```

5. Write a C program to perform a binary search for array of strings

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
  void main()
    {
         int low,mid,high,res,i,n;
         char name[10][20];
         printf("\n Enter no of names\n ");
         scanf("%d",&n);
         printf("\n Enter the  names ");
              for(i=0;i<n;i++)
              gets(name[i]);
         low=0;
         high=n-1;
              while(low<=high)
              {
                   mid=(low+high)/2;
                   res=strcmp(key,name[mid]);
                   if(res==0)
                   {
                   printf("Successful search \n");
                   getch();
                   exit(0);
                   }
                   if(res>0)
                           low=mid+1;
                   else
                           high=mid-1;
              }
         printf("unsuccessful search");
         getch();
    }
```