Q.1 Read from a terminal using scanf function and print using printf function.

```
#include<stdio.h>

int main( )

{

      char str[40];

      printf("Enter text : \n");

      scanf("%s", str);

      printf("\n");

      printf("Entered text = %s", str);

  }
```

Q2. Read lines of text from a terminal using fgets function and print using puts function.

```
#include <stdio.h>

int main()

{

   int b;

   int size = 10;

   char *string;

   printf ("Please enter a string: ");

   string = (char *) malloc (size);

   b = getline (&string, &size, stdin); //stdin- standard input

   if (b == -1)

   {

      puts ("ERROR!");

   }

   else
```

```
   {
      puts ("You entered the following string:");
      puts (string);
   }
   return 0;
}
```

Q3. Convert

a. **Upper case to Lower case**
b. **Lower case to Upper case**
c. **Toggle case**
d. **Sentence case.**

a.

```
#include<stdio.h>
#include<string.h>
int main()
{
  char str[25];
  int i;
  printf("Enter the string: ");
  scanf("%s",str);
  for(i=0;i<=strlen(str);i++)
  {
    if(str[i]>=65&&str[i]<=90) // A-Z ASCII value(65-90)
      str[i]=str[i]+32; // Upper case+32= lower case
  }
  printf("\nLower Case String is: %s",str);
  return 0;
}
```

b.

```
#include<stdio.h>
#include<string.h>
int main()
{
  char str[25];
  int i;
  printf("Enter the string:");
  scanf("%s",str);
  for(i=0;i<=strlen(str);i++)
  {
    if(str[i]>=97&&str[i]<=122) //(a-z) ASCCI value 97-122
      str[i]=str[i]-32; //lower case-32 = Upper case
  }
  printf("\nUpper Case String is: %s",str);
  return 0;
}
```

c.

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

```
	char Str1[100];
	int i;
	printf("\n Please Enter any String to Toggle : ");
	gets(Str1);
	for (i = 0; Str1[i]!='\0'; i++)
	{
		if(Str1[i] >= 'a' && Str1[i] <= 'z')
		{
			Str1[i]=Str1[i]-32;
		}
		else if(Str1[i]>= 'A' && Str1[i]<= 'Z')
		{
			Str1[i]=Str1[i]+32;
		}
	}
	printf("\n The Given String after Toggling Case of all Characters = %s", Str1);
	return 0;
}
```

d.

```
#include <stdio.h>
#include <string.h>
int main()
{
	char str[50]={0};
	int length=0,i=0,j=0,k=0;
	printf("\nEnter the string : ");
```

```
gets(str);
length = strlen(str);
for(i=0;i<length;i++)
{
	if( (i==0) && (str[i]>='a' && str[i]<='z'))
	{
		str[i] = str[i] - 32;
	}
	else if(str[i]=='.')
	{
		if(string[i+1] == ' ')
		{
			if(str[i+2]>='a' && str[i+2]<='z')
			{
				str[i+2] = str[i+2] - 32;
			}
		}
		else
		{
			if(str[i+1]>='a' && str[i+1]<='z')
			{
				str[i+1] = str[i+1] - 32;
			}
		}
	}
}
printf("Final string is : %s",str);
```

```
}
```

Q4. Perform string Concatenation(With and without string handling functions).

Without:

```
#include <stdio.h>

int main()

{

  char str1[50], str2[50],i,j;

  printf("Enter first string: ");

  scanf("%s",str1);

  printf("Enter second string: ");

  scanf("%s",str2);

  for(i=0; str1[i]!='\0'; ++i);

  for(j=0; str2[j]!='\0'; ++j, ++i)

  {

    str1[i]=str2[j];

  }

  str1[i]='\0';

  printf("Output: %s",str1);

  return 0;

}
```

With:

```
#include <stdio.h>

#include<string.h>

int main()
```

```
{
  char s1[20];
  char s2[20];
    printf("Enter the first string : ");
    scanf("%s", s1);
    printf("\nEnter the second string :");
    scanf("%s",s2);
    strcat(s1,s2);
    printf("The concatenated string is : %s",s1);
    return 0;
}
```

Q5. Perform String Reversal (With and without string handling function)

Without:

```
#include<stdio.h>
#include<conio.h>
int main()
{
    int i, j, k;
    char str[100];
    char rev[100];
    printf("Enter a string:\t");
    scanf("%s",str);
    printf("The original string is %s\n", str);
    for(i = 0; str[i] != '\0'; i++);
    {
```

```
        k = i-1;
    }
    for(j = 0; j <= i-1; j++)
    {
        rev[j] = str[k];
        k--;
    }
    printf("The reverse string is %s\n", rev);
    return 0;
}
```

With:

```
#include<stdio.h>
#include<string.h>
int main()
{
   char name[30] = "Hello";
   printf("String before strrev: %s\n",name);
   printf("String after strrev: %s",strrev(name));
   return 0;
}
```

Q6. Perform Substring Extraction (With and Without String Handling Functions)

Without:

```
#include <stdio.h>
int main()
{
```

```
  char string[1000], sub[1000];
  int position, length, c = 0;
  printf("Input a string\n");
  gets(string);
  printf("Enter the position and length of substring\n");
  scanf("%d%d", &position, &length);
  while (c < length) {
    sub[c] = string[position+c-1];
    c++;
  }
  sub[c] = '\0';
  printf("Required substring is \"%s\"\n", sub);
  return 0;
}
```

With:

```
#include <stdio.h>
#include <string.h>
int main()
{
const char* lineConst = "abhijeet \"behu\" ria";
char line[256];
char *subString;
strcpy(line, lineConst);
subString = strtok(line,"\"");
subString=strtok(NULL,"\"");
printf("the thing in between quotes is '%s'\n", subString);
return 0;
```

```
}
```

```
the thing in between quotes is 'behu'
```

Q7. Copy one string into another and count the no of elements copied. (With and without string handling function).

Without:

```
#include <stdio.h>

int main()

{

   char s1[100], s2[100], i;

   int count;

   printf("Enter string s1: ");

   fgets(s1, sizeof(s1), stdin);

   for (i = 0; s1[i] != '\0'; ++i) {

      s2[i] = s1[i];

      count++;

   }

   s2[i] = '\0';

   printf("String s2: %s", s2);

   printf("Number of string copied:%d",count);

   return 0;

}
```

With:

```
#include<stdio.h>

#include<string.h>

int main()

{

   char c[100];
```

```
    char o[100];
    printf("\n\nEnter the string: ");
    gets(o);
    strcpy(c,o);
    printf("\n\nThe copied string is: %s\n\n", c);
    return 0;
}
```

Q8. Read a string and prints if it is a palindrome or not.

```
#include <stdio.h>
#include <string.h>
int main()
{
    char string1[20];
    int i, length;
    int flag = 0;
    printf("Enter a string:");
    scanf("%s", string1);
    length = strlen(string1);
    for(i=0;i < length ;i++)
    {
        if(string1[i] != string1[length-i-1])
        {
            flag = 1;
            break;
        }
    }
    if(flag)
```

```
    {
        printf("%s is not a palindrome", string1);
    }
    else
    {
        printf("%s is a palindrome", string1);
    }
    return 0;
}
```

Q9. Read a line of text and count all occurrences of particular word.

```
#include<stdio.h>
#include<string.h>
int main()
{
 int strln,wordln,i,j,k,flag,count=0;
 char str[200],word[20];
 printf("Enter line of text:n");
 gets(str);
 printf("Enter the word to count:n");
 scanf("%s",word);
 strln=strlen(str);
 wordln=strlen(word);
 for(i=0;i<strln;i++)
 {
  if(str[i]==word[0]&&((str[i-1]==' '||i==0)&&(str[i+wordln]==' '||str[i+wordln]==''))
  {
   for(flag=0,k=i+1,j=1;j<wordln;j++,k++)
```

```
  {
   if(str[k]==word[j])
   {
    flag++;
   }
  }
  if(flag==wordln-1)
  {
   count++;
  }
 }
}
 printf("Number of occurence of '%s' = %dn",word,count);
 return 0;
}
```

Q10. Read a string and rewrite it in the alphabetical order.

```
#include <stdio.h>
#include <string.h>
int main ()
{
        char string[100];
   printf("\n\t Enter the string : ");
        scanf(“%s”,string);
        char temp;
        int i, j;
        int n = strlen(string);
        for (i = 0; i < n-1; i++) {
```

```
                for (j = i+1; j < n; j++) {
                        if (string[i] > string[j]) {
                                temp = string[i];
                                string[i] = string[j];
                                string[j] = temp;
                        }
                }
        }
        printf(“The sorted string is : %s”, string);
        return 0;
}
```

Q11. Print the words ending with letter S.

```
#include <stdio.h>
#include <string.h>
char str[100];
int main()
{
   int i, t, j, len;
   printf("Enter a string : ");
   scanf("%[^\n]s", str);
   len = strlen(str);
   str[len] = ' ';
   for (t = 0, i = 0; i < strlen(str); i++)
   {
      if ((str[i] == ' ') && (str[i - 1] == 's'))
      {
         for (j = t; j < i; j++)
```

```
            printf("%c", str[j]);
        t = i + 1;
        printf("\n");
      }
      else
      {
        if (str[i] == ' ')
        {
          t = i + 1;
        }
      }
   }
   return 0;
}
```

Q12. Delete all repeated words in the line of text.

```
#include <stdio.h>
#include <string.h>
#define SIZE 500
void duplicateRemover(char *, const int);
int main(void)
{
   char someString[SIZE];
   puts("Enter text: ");
   fgets(someString, SIZE, stdin);
   someString[strcspn(someString, "\n")] = 0;
   printf("\n%s", "Text without repeated words: ");
   duplicateRemover(someString, SIZE);
```

```
}
void duplicateRemover(char *arrayPtr, const int sizeP)
{
   char wordTable[sizeP][sizeP], *tokPtr;
   size_t i, j, k, l;
   tokPtr = strtok(arrayPtr, " ");
   strcpy(wordTable[0], tokPtr);
   for(i = 1; (tokPtr = strtok(NULL, " ")) != NULL; i++)
      strcpy(wordTable[i], tokPtr);
   for(j = 0; j <= i; j++)
      for(k = j + 1; k <= i; k++)
         if(strcmp(wordTable[j], wordTable[k]) == 0)
         {
            for(l = k; l < i; l++)
               strcpy(wordTable[l], wordTable[l + 1]);
         k = j;
         i--;
         }
   for(l = 0; l <= i; l++)
      printf("%s ", wordTable[l]);
}
```