COMPUTER PROGRAMMING - I

Basics of strings

- String is a one-dimensional array of characters which is terminated by a **null** character '\0'.
- char name[ ] = { 'H', ‘E', ‘L', ‘L', ‘O’, '\0' } ;
- Each character in the array occupies one byte of memory and the last character is always ‘\0’.
- ‘\0’ is called null character..
- The terminating null (‘\0’) is important, because it is the only way the functions that work with a string can know where the string ends.

Basics of strings

- For example, the string used above can also be initialized as,
- char name[ ] = "HELLO" ;
- In this declaration ‘\0’ is not necessary. C inserts the null character automatically.

Basics of strings

- int main( )
- {
- char name[25] ;
- printf ("Enter your name ") ;
- gets(name);
- printf ("%s", name);
- }

Basics of strings

- Enter your name Radhika Chapaneri
- Radhika Chapaneri

Library functions for strings

- strlen – returns the length of a string
- strcpy - copy one string into another
- strcat - append one string onto the right side of the other
- strcmp – compare alphabetic order of two strings

- Include:
- #include <string.h>

strlen

- strlen(str) returns length of string excluding null character
- strlen(“tttt”) = 4 not 5 since \0 not counted

Length of string with library function

```
#include <stdio.h>
#include <string.h>
int main()
{
    char str[100];
    int n;
    printf("Enter a string: ");
    gets(str);
     n = strlen(str);
    printf("Length of string: %d",n);
    return 0;
}
```

Length of string without library function

```
//Calculating length of string
int main()
{
    char str[100];
    int i,count=0;
    printf("Enter a string: ");
    gets(str);
for(i=0; str[i]!='\0'; i++)
    {
        count++;
    }
    printf("Length of string: %d",count);
    return 0;
}
```

Length of string without library function

```
Enter a string: Hello friends
Hello friends
Length of string: 13
```

Example with strlen

```
//Count number of t in = "tommy tucket took a tiny ticket"
  int main()
     {
      int i, count,n;
    char x[ ] = "tommy tucket took a tiny ticket";
       count = 0;
       n= strlen(x);
      for (i = 0; i < n;i++)
         {
          if (x[i] == 't')
             count++;
         }
     printf("The number of t in %s is %d \n ", x,count);

    }
```

Vowels Example with strlen

```
#include <stdio.h>
#include <string.h>
  main()
     {
      int i, count;
    char x[] = “tommy tucket took a tiny ticket ”;
      count = 0;
      for (i = 0; i < strlen(x);i++)
         {
          if ((x[i] == ‘a’)||(x[i]==‘e’)||(x[i]==‘I’)||(x[i]==‘o’)||(x[i]==‘u’))
  count++;
         }
     printf(“The number of vowels’s in   %s is %d \n “, x,count);

    }
```

No of Words Example with strlen

```
#include <stdio.h>
#include <string.h>
  main()
     {
      int i, count;
    char x[] = “tommy tucket took a tiny ticket ”;
      count = 0;
      for (i = 0; i < strlen(x);i++)
          {
           if ((x[i] == ‘ ‘) count++;
          }
     printf(“The number of words’s in   %s is %d \n “, x,count+1);

    }
```

strcpy

- strcpy(destinationstring, sourcestring)
- Copies source string into destination string
- For example
- strcpy(str, “hello world”); assigns “hello world” to the string str

Example with strcpy

```
#include <stdio.h>
#include <string.h>
 int main()
{
    char source[ ] = "Hello World";
    char dest[25];
    strcpy(dest,source);
    printf("The string in source is %s \n", source);
    printf("The string in destination is %s \n",dest);
}
```

Copying a string without library function

```
// Copy one string into another and count number of characters copied
int main()
{
    char string1[20],string2[20];
    int i;
    printf("Enter a string \n");
    gets(string1);
    for(i=0; string1[i] !='\0';i++)
    {
        string2[i] = string1[i];
    }
    string2[i] = '\0';
    printf("Copied string is %s \n",string2);
    printf("Number of character in copied string = %d",i);

}
```

Copying a string

- Enter a string
- hello world
- Copied string is hello world
- Number of character in copied string = 11

strcat

- strcat function joins two strings together.
- strcat(string1, string2)

- string2 is appended to string 1
- It does by removing the null character at the end of the string1 and placing string2 from there
- The string at string2 remains unchanged

- For example if string1 =“Very ”
- string2 =“good ”
- So string1 will be “Very Good”

Example with strcat

```
/Concantenation of string using strcat
int main()
   {
      char string1[ ] = "Very";
      char string2[ ] = "Good";
      printf("Before Concatenation \n");
       printf("The string in  first array  is %s \n ",string1);
        printf("The string in  second array  is %s \n ",string2);
       strcat(string1,string2);
       printf("\n");
        printf("After Concatenation \n");
         printf("The string in  first array  is %s \n ",string1);
         printf("The string in  second array  is %s \n ",string2);
   }
```

```
Before Concatenation
The string in  first array  is Very
 The string in  second array  is Good

After Concatenation
The string in  first array  is VeryGood
 The string in  second array  is Good
```

- .

Concatenation without using string library functions

```
#include <stdio.h>
void concatenate(char [], char []);
int main()
{
  char p[100], q[100];
  printf("Input a string\n");
  gets(p);
  printf("Input a string to concatenate\n");
  gets(q);
  concatenate(p, q);
  printf("String obtained on concatenation is \"%s\"\n", p);
  return 0;
}
```

Concatenation without using string library functions

```
void concatenate(char p[], char q[]) {
  int c, d;
  c = 0;
  while (p[c] != '\0') {
    c++;
  }
  d = 0;
  while (q[d] != '\0') {
    p[c] = q[d];
    d++;
    c++;
  }
  p[c] = '\0';
}
```

strcmp

- strcmp(string1, string2)

- Compares string1 and string2 alphabetically
- Returns 0 if they are equal
- If they are not it has the numeric difference between the first non matching characters in the string
- Returns a negative value if string1 precedes string2 alphabetically
- Returns a positive value if string2 precedes string1 alphabetically
- Note lowercase characters are greater than Uppercase

strcmp

```
#include<stdio.h>
#include<string.h>
int main()
{
   char a[100], b[100];
   printf("Enter the first string\n");
   gets(a);
   printf("Enter the second string\n");
   gets(b);
   if( strcmp(a,b) == 0 )
      printf("Entered strings are equal.\n");
   else if(strcmp(a,b)<0)
      printf("a precedes b.\n");
      else
         printf("b precedes a.\n");
      return 0;
}
```

String compare without library function

```
#include<stdio.h>
int main() {
   char str1[30], str2[30];
   int i;
   printf("\nEnter two strings :");
   gets(str1);
   gets(str2);
   i = 0;
   while (str1[i] == str2[i] && str1[i] != '\0')
      i++;
   if (str1[i] > str2[i])
      printf("str1 > str2");
   else if (str1[i] < str2[i])
      printf("str1 < str2");
   else
      printf("str1 = str2");
   return (0);
}
```

Compare strings

```
//Comparison of two strings
int main() {
   char str1[30], str2[30];
   int i;
   printf("\nEnter two strings :");
   gets(str1);
   gets(str2);
   i = 0;
   while (str1[i] == str2[i] && str1[i] != '\0' && str2[i] != '\0')
      i++;
   if (str1[i] == '\0' && str2[i] == '\0')
      printf("Strings are equal");
      else
      printf("Strings are not equal");
}
```

Compare strings

- Enter two strings :hello world
- hello
- Strings are not equal

Program to check with string is Palindrome

```
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;
              }
           }
```

Program to check with string is Palindrome

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

Program to reverse a string

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

int main()
{
   char s[100], r[100];
   int n, c, d;

   printf("Input a string\n");
   gets(s);

   n = strlen(s);
```

Program to reverse a string

```
   for (c = n - 1, d = 0; c >= 0; c--, d++)
      r[d] = s[c];

   r[d] = '\0';

   printf("%s\n", r);

   return 0;
}
```

Program to reverse a string

```
int main() {
   char str[100], temp;
   int i, j = 0;

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

   i = 0;
   j = strlen(str) - 1;
```

Program to reverse a string

```
    while (i < j) {
       temp = str[i];
       str[i] = str[j];
       str[j] = temp;
       i++;
       j--;
    }

    printf("\nReverse string is :%s", str);
    return (0);
}
```

Program to count vowels, consonants, digit and spaces

```
int main()
{
    char line[150];
    int i, vowels, consonants, digits, spaces;

    vowels =  consonants = digits = spaces = 0;

    printf("Enter a line of string: ");
    gets(line);

    for(i=0; line[i]!='\0'; ++i)
    {
        if(line[i]=='a' || line[i]=='e' || line[i]=='i' ||
           line[i]=='o' || line[i]=='u' || line[i]=='A' ||
           line[i]=='E' || line[i]=='I' || line[i]=='O' ||
           line[i]=='U')
        {
            vowels++;
        }
```

Program to count vowels, consonants, digit and spaces

```
else if((line[i]>='a'&& line[i]<='z') || (line[i]>='A'&& line[i]<='Z'))
        {
            consonants++;
        }
        else if(line[i]>='0' && line[i]<='9')
        {
            digits++;
        }
        else if (line[i]==' ')
        {
            spaces++;
        }
    }

    printf("Vowels: %d",vowels);
    printf("\nConsonants: %d",consonants);
    printf("\nDigits: %d",digits);
    printf("\nWhite spaces: %d", spaces);

    return 0;
}
```