Strings

SESSION 10

Objectives

- Explain string variables and constants
- Explain pointers to strings
- Perform string input/output operations
- Explain the various string functions
- Explain how arrays can be passed as arguments to functions
- Describe how strings can be used as function arguments

String variables

- Strings are arrays of characters terminated by the NULL ('\0') character.
- String variables can be assigned string constants.
- A string constant is a sequence of characters surrounded by double quotes.
- The '\0' null character is automatically added in the internal representation of a string.
- While declaring a string variable, allow one extra element space for the null terminator.

Declaring string variables

- A typical string variable declaration is:.

```
char s[10];
```

- **s** is a **character array** variable that can hold a maximum of 10 characters including the null terminator.

String I/O operations -1/2

- String I/O operations are carried out using functions from the standard I/O library called **stdio.h**
- **gets()** function is the simplest method of accepting a string through standard input:
  - Characters are accepted till the Enter key is pressed.
  - The terminating '\n' new line character is replaced with the '\0' character

  Syntax : **gets(str);**
- **puts()** function is used to display a string on the standard output device.

  Syntax : **puts(str);**

String I/O operations -2/2

- The **scanf()** and **printf()** functions are used to accept and display mixed data types with a single statement.

- The syntax to accept a string is as follows:

```
scanf("%s", str);
```

- The syntax to display a string is as follows:

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

String functions

- Functions for handling strings are found in the standard header file **string.h**.
- Few of the operations performed by these functions are:
  - Concatenating strings
  - Comparing strings
  - Locating a character in a string
  - Copying one string to another
  - Calculating the length of a string

strcat() function

- Joins two string values into one.
- Syntax:

```
strcat(str1, str2);
```

- Concatenates the str2 at the end of str1
- The function returns str1

strcmp() function

- Compares two strings and returns an integer value based on the results of the comparison.

- Syntax:

```
strcmp(str1, str2);
```

- The function returns a value:
  - Less than zero if str1<str2
  - Zero if str1 is same as str2
  - Greater than zero if str1>str2

strchr() function

- Determines the occurrence of a character in a string.
- Syntax:

strchr (str, chr);

- The function returns a value of pointer:
  - Point to the first occurrence of the character (**chr**) in the string, **star**
  - NULL if it is not present

strcpy() function

- Copies the value in one string onto another
- Syntax:

```
strcpy(str1, str2);
```

- The value of str2 is copied onto str1
- The function returns **str1**

strlen() function

- Determines the length of a string
- Syntax:

```
strlen(str);
```

- The function returns an integer value for the length of **str**

Passing array to function -1/2

- When an array is passed as an argument to a function, only the address of the array is passed

- The array name without the subscripts refers to the address of the array

```
void main()   {
   int ary[10];
   fn_ary(ary);
    .
    .
}
```

Passing array to function -

```
void  main()  {
    int num[5];
    int sum_arr(int num_arr[]);    /* Funct
    for(int i=0; i<5 ; i++)  {     /* input numbers into array */
        printf("\nEnter number %d:", i+1);
        scanf("%d", &num[i]);
    }

    printf("\nThe sum of the array is %d", sum_arr(num) );
}

int sum_arr(int num_arr[]){    /* Function definition */
    int total;
    for(int i=0,total=0;i<5;i++)
        total+=num_arr[i];
    return total;
}
```

Sample output :

```
Enter number 1: 5
Enter number 2: 10
Enter number 3: 13
Enter number 4: 26
Enter number 5: 21
The sum of the array is 75
```

Passing Strings to Functions – 1/3

```
int longest(char lines_arr[][20]);

void main()     {
  char lines[5][20];
  int i, longctr=0;

  /* Accepts string into the array */
  for(i=0; i<5; i++)              {
     printf("\nEnter string %d: ", i+1);
     scanf("%s", lines[i]);
  }

  longctr = longest(lines);
  printf("\nThe longest string is %s", lines[longctr]);
}
```

Passing Strings to Functions – 2/3

```
int longest(char lines_arr[][20])        /* Function definition */
{
   int index =0 , prev_len, new_len;
   prev_len = strlen(lines_arr[0]);  //length of the first element

   for(int i=1 ; i<5 ; i++) {
       new_len = strlen(lines_arr[i]); //length of next element
       if(new_len > prev_len){
           /* Stores the subscript of the longer string */
           index = i;
           prev_len = new_len;
       }
   }
   /* Returns the index of the longest string */
   return index;
}
```

Passing Strings to Functions – 3/3

Sample output of the program

Enter string 1: The

Enter string 2: Sigma

Enter string 3: Protocol

Enter string 4: Robert

Enter string 5: Ludlum

The longest string is Protocol

Passing Strings to Functions – 3/3

Sample output of the program

```
Enter string 1: The

Enter string 2: Sigma

Enter string 3: Protocol

Enter string 4: Robert

Enter string 5: Ludlum

The longest string is Protocol
```