Experiment No. 5

Arrays and String in C Language

1. **Objective:** The experiment is designed to provide basic knowledge of Arrays and String in C Language. Students will be able to develop logics which will help them to create programs, applications in C for arrays and strings.

2. **Theoretical Background:** Arrays are data types found in most high level languages. The array allows the storage of a number of data values in one variable. The underlying concepts to arrays are derived from matrix algebra. Besides being used to represent matrices, arrays are also suited for a variety of other purposes because, in C, this data type represents a defined number of contiguous (i.e. consecutive) bytes in memory.

   An array is the grouping of repetitive data (of the same type) that shares the same name. Arrays are useful when manipulating lists of data. The data shares the same name but each element in the array can be accessed individually. An array can be of any data type. Each constituent value of an array is called an element. When an array is defined in C, the programmer specifies the number of elements in the array. The elements of an array can be ordered in different pattern schemes. A one-dimensional array can be conceptually viewed as a single string or row of a predefined number of values. For example: if we declare an array of 6 elements as

   int a[6];

   then what happens is shown below:

a[0] int1	a[1] int2	a[2] int3	a[3] int4	a[4] int5	a[5] int6

   Rules for Forming Arrays in C:

   1. An array name must be chosen according to the same rules as are used for naming any other variable.

   2. The name of the array cannot be the same as that of any other variable declared within the function.

   3. The size of the array (the number of elements) is specified using the subscript notation.

   4. The dimension used to declare an array must always be a positive integer constant, or an expression that can be evaluated to a constant when the program is compiled.

5. All elements of an array should be of the same type. In other words, if an array is declared to be type int, it cannot contain elements that are not of type int.

The following assignment statements are all valid in C:

```
x[0] = 14 ;

x[5] = x[14] -7;

x[7] = x [6] +8+x[5];

x[3] += 1;
```

Strings in C: Strings are defined as an array of characters. The difference between a character array and a string is that the string is terminated with a special character '\0'. Strings are actually a one-dimensional array of characters terminated by a null character '\0'. Thus a null-terminated string contains the characters that comprise the string followed by a null. When the compiler encounters a sequence of characters enclosed in the double quotation marks, it appends a null character \0 at the end. Before you can work with strings, you need to declare them first. Since string is an array of characters. You declare strings in a similar way like you do with arrays.

```
char a[6];
```

then what happens is shown below:

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]
char1	char 2	char3	char4	char5	char6

The following declaration and initialization will create a string consisting of the word "Hi". To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word "Hi."

```
char a[3] = {'H','i','\0'};
```

If you follow the rule of array initialization then you can write the above statement as follows –

```
char a[ ] = "Hi";
```

a[0]	a[1]	a [2]
H	i	\0

In the above example, you do not place the null character at the end of a string constant. The C compiler automatically places the '\0' at the end of the string when it initializes the array. You can initialize strings in a number of ways -

```
char c[ ] = "abcd";

char c[50] = "abcd";

char c[ ] = {'a', 'b', 'c', 'd', '\0'};

char c[5] = {'a', 'b', 'c', 'd', '\0'};
```

Rules for Forming Strings in C:

1. A string name must be chosen according to the same rules as are used for naming any other variable.

2. The name of the string cannot be the same as that of any other variable declared within the function.

3. The size of the string (the number of elements) is specified using the subscript notation which should also accommodate the null character.

4. The dimension used to declare a string must always be a positive integer constant, or an expression that can be evaluated to a constant when the program is compiled.

5. All elements of a string should be of the character type. In other words, it cannot contain elements that are not of type char.

3. Software requirements:

- Any IDE that supports a C compiler (preferably Code::Blocks; any other IDE or text editor with a C compiler installed will also be accepted.)

4. Example Problems:

a. Find the sum of all elements of an array

Code:

```
#include <stdio.h>
int main ()
{
    int midterm[10]={2, 2, 3, 3, 4};
    int i;
    int total = 0;
    for(i=0; i<5; i++)
    {
        total = midterm [i]+total;
    }
    printf("The Total Marks is %d", total);
    return 0;
```

```
}
```

b. Write a program to print the following numbers in reverse order
22 91 65 33 19 47 49 88 94 39

Code:

```
#include <stdio.h>
int main()
{
    int a[10]={22, 91, 65, 33, 19, 47, 49, 88, 94, 39};
    int i;
    //for printing current order
    for(i=0; i<10; i++)
    {
        printf("%d ", a[i]);
    }
    printf("\n");
    //for printing reverse order
    for(i=9; i>=0; i--)
    {
        printf("%d ", a[i]);
    }
}
```

c. Take a string as input from the user and print every single character of it into a separate line. If user gives “Programming”, output should look like the following

P
r
o
g
r
a
m
m
i
n
g

Code:

```
#include <stdio.h>
int main()
{
    char a[50];
    int index;
    printf("Enter The String Value\n");
    gets(a);
    for(index = 0; a[index] != '\0'; index++)
    {
        printf("%c", a[index]);
        printf("\n");
    }
    return 0;
}
```

d. Write a program in C to copy one string to another string

Code:

```
#include <stdio.h>
#include <string.h>
int main()
{
    char source[50];
    char destination[50];
    printf("Enter any string within 50 characters: ");
    gets(source);
    strcpy(destination, source);
     printf("Here  is  the  text  that  you  entered:  %s",
destination );
    return 0;
}
```

5. Results after running the example programs:

Screenshots of the results:

6. Lab report assignments (submit source code and screenshots of results):

a. Write a program in C using an array which finds the largest and smallest elements in a group of numbers.

b. Write a program in C to augment one string with another string. For example: String ‘a’ contains “foot”, string ‘b’ contains “ball”, now string ‘c’ should contain “football”.
c. Write a program to print a string in reverse order.
d. A palindrome is a word, number or other sequence of characters which reads the same backward as forward, such as ‘madam’. Write a C program which can determine if a word is palindrome or not.

7. Comment/Discussion on the obtained results and discrepancies (if any).