National University of Computer and Emerging Sciences

Lab Manual 03
Object Oriented Programming

Course Instructor	Mr. Bismillah Jan
Lab Instructor (s)	Mr. Saif Ali Mr. Dilawar Shabbir
Section	BCS-2E
Semester	Spring 2021

Department of Computer Science
FAST-NU, Lahore, Pakistan

1.1 Objectives

After performing this lab, students shall be able to:

- character pointers
- 2D character arrays
- Cstring functions

Note-> you can use ctring manipulators for display like setw().

TASK 1:

a) Write a function **char* inputFirstName(int& size)** that takes size of char array from user, allocates memory for the array and input Student first name from user and return its pointer.

b) Write a function **char* inputLastName(int& size)** that takes size of char array from user, allocates memory for the array and input Student last name from user and return its pointer.

c) Write a function **void searchStudentByFirstName(char* firstNameList, const int& size)** that takes input of first name to be search from user and print "Found/Not Found"on screen.

d) Write a function **char * returnFullNameList(char* firstNameList, char* lastNameList, const int& size)** that will concatenate first name and last name and store it in a new dynamic char array and return its pointer.

TASK 2:

a) Declare a string "- This is, my Object Oriented Programming Lab."

b) Tokenize the above string using strtok function and store it in a dynamic char array.

c) Print dynamic char array on screen.

TASK 3:

a) Write a function **char** AllocateMemory(int& rows, int& cols)** that takes size of matrix (rows and columns) from user, allocates memory for the matrix and return its pointer.

b) Write a function **void InputMatrix(char** matrix, const int rows, const int cols)** which takes input the values in matrix from user(console).

c) Write a function **void DisplayMatrix(char** matrix, const int& rows, const int& cols)** that displays the matrix in proper format.

d) Write a function that does the following:

- Creates three dynamic char arrays namely **alphabets, numbers, and specialchar.** (Define the sizes yourself).
- Iterate the 2D array and separate alphabet elements and save them in the alphabets array, separate number elements and save them in numbers array and separate special character elements and save them in the specialchar array.
- The function returns the three arrays **alphabets, numbers, and specialchar.**
- **Note:** The three arrays should not consume any extra space. Resize the arrays accordingly.

For example, if the **Sample Matrix** is

A	1	v	@
+	9	s	6
P	#	^	4

Your function will return the following arrays:

alphabets = A v s P
numbers = 1 9 6 4
specialchar = @ + # ^

e) Create a function **sortArray** that takes in input three arrays **alphabets, numbers, and specialchar and their sizes** and sort each of these arrays according to their ASCII values (ascending order) and print these sorted arrays on the console.

For the above example the sorted arrays are:

alphabets = A P s v
numbers = 1 4 6 9
specialchar = # + @ ^

f) Write a function **void DeallocateMemory**that deallocates all the memory.

```
void main()
{
        int rows, cols;

        //take input from user for rows and cols

        char** matrix = AllocateMemory (rows, cols);
        DisplayMatrix(matrix, rows, cols);
        InputMatrix(matrix, rows, cols);

        // additional function calls

        //deallocate memory

}
```