National University of Computer and Emerging Sciences



Lab Manual 04

**Section BCS 2B**

Object Oriented Programming

|  |  |
| --- | --- |
| Course Instructor | Miss Abeeda Akram |
| Lab Instructor (s) National University of Computer and Emerging Sciences    Lab Manual 02  Object Oriented Programming  Department of Computer Science  FAST-NU, Lahore, Pakistan  1.1 Objectives  After performing this lab, students shall be able to:   Have an improved understanding of pointers.   Create and manipulate 1D dynamic array.   Allocation and de-allocation of 1D array.   Passing dynamic arrays into functions. | Ms. Abiha Aftab  Mr. Dilawar Shabbir |
| Section | BCS – 2A |
| Semester | Spring 2021 |

Department of Computer Science

FAST-NU, Lahore, Pakistan

## Objectives

After performing this lab, students shall be able to:

* Have an improved understanding of pointers.
* Character array.
* Multiple indirection with functions.
* Sorting in arrays

**TASK 1:**

Write and test a function to reverse a string in place (not use another array), without any duplication of characters. Use this prototype.

void reverse(char \* s)

**TASK 2:**

**Exercise 1:**

Write a function **int GetCharacterCount(char\* myString, char c)** that takes a character ***ch*** and a c-string ***myString*** and returns total number ofoccurrences of *ch* in *myString*.

**Sample Output:**

**myString:** Pakistan

**ch:** a

**Total No of Occurances:** 2

**Exercise 2:**

Update the program written in Exercise 2 to accept a sentence in myString and test it. You need to replace **“cin>>myString;”** with **“cin.getline(myString,50);”**

**Sample Output:**

**myString:** I am Pakistani

**ch:** a

**Total No of Occurances:** 3

**TASK 3:**

Write the following function that indirectly sorts the floats pointed to by the first n pointers in the array p by rearranging the pointers not data:

void sort(float\* p[], int n);

Use the following main as driver and also write code of print function.

void main (){

float a[81] = (44.4, 77.7, 22.2, 88.8, 66.6, 33.3, 99.9, 55.5);

float\* p[8];

for (int i = 0; i c 8; i++)

p[i] = new float (a[i]);// p[i] points to a[i]

print(p, 8);

sort(p, 8);

print(p, 8);

}

**TASK 4:**

1. 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.
2. Write a function **void InputMatrix(char\*\* matrix, const int rows, const int cols)** which takes input the values in matrix from user(console).
3. Write a function **void DisplayMatrix(char\*\* matrix, const int& rows, const int& cols)** that displays the matrix in proper format.
4. 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.

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 = @ + # ^

**TASK 5:**

First find the number of tokens that can be formed from data. This will be the number of **rows** for **char \*\*list\_tokens**. Each row will have different number of columns. e.g. If string is **my,name** and delimiter is ‘,’ then following shall be the result.

**0 1 2 3 4**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | m | y | ‘\0’ |
| **1** | n | a | m | e | ‘\0’ |

**Don’t allocate extra memory**. Release memory before exiting program.

**Sample:**

**Input**: my,name,is,Mr,Faheem

**Delimiter**: ,

**Tokens are**:

my

name

is

Mr

Faheem