National University of Computer and Emerging Sciences

Lab Manual 03
Object Oriented Programming

Course Instructor	Mr. Usama Hassan
Lab Instructor (s)	Ms. Fariha Maqbool Mr. Sohaib Ahmad
Section	BSE-2C
Semester	Spring 2023

Department of Computer Science
FAST-NU, Lahore, Pakistan

Objectives

After performing this lab, students will practice:

- ✓ 2D dynamic integer arrays
- ✓ 1D dynamic character arrays

TASK-1:

a) Write a function **void InputMatrix(int**& matrix, const int rows, const int cols)** which allocates the memory for the matrix and takes input the values in matrix from user(console)

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

c) Write a function **returnLarger** that takes as parameters a pointer to a 2D array, its dimensions and a threshold value. This function should store all the elements greater than the threshold value into a 1D array and return it. The last column of 1D array should contain -1.
For Example, if the threshold value is **4** and **Sample Matrix** is

1	4	3
9	2	6
5	7	8

Your function will return array containing all the elements greater than 4 in all the columns i.e.
9, 5, 7, 6, 8

d) Write a function called **maxCol** that takes as parameters a pointer to a 2D array and its dimensions. It should return the largest element in each column of the array. Since there is more than one column in 2D array, you have to return a 1D dynamic array that contains largest of each column. The last column of 1D array should contain -1.
For example, if the **Sample Matrix** is

1	4	8
9	1	6
5	7	2

Your function will return array containing maximum elements of all the columns i.e.
9, 7, 8

e) **Bonus Task**: Implement the above mentioned tasks with different number of columns in each row

f) **(Concatinate tables)** It takes two 2D arrays and returns a new 2D array that is concatenated on y or x axis.

int ** concat_table (int **table1, int **table2, int row1, int col1, int row2, int col2, bool axis); //axis 0 mean x axis 1 means Y axis.

Array 1 1,2,3 4,4,5 3,4,6	**Array 2** 5,8,9 4,9,4 4,6,0
If axis = 0 **(x-axis)**	If axis = 1 **(y-axis)**
Output 1,2,3,5,8,9 4,4,5,4,9,4 3,4,6,4,6,0	**Output** 1,2,3 4,4,5 3,4,6 5,8,9 4,9,4 4,6,0

g) Write a function **void DeallocateMemory(int** matrix, const int& rows)** that deallocates all the memory.

Example:

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

	//take input from user for rows and cols

	int ** matrix = nullptr;
	InputMatrix(matrix, rows, cols);
	DisplayMatrix(matrix, rows, cols);

	int * largerValues = returnLarger(matrix,rows, cols);
	for(int i=0; i<cols; i++)
		cout<< *( largerValues + i) <<",";
	cout<<endl;

	int * maxColValues = maxCol(matrix,rows, cols);
	for(int i=0; i<cols; i++)
		cout<< *(maxColValues + i) <<",";
	cout<<endl;

	int row1, col1;
	//take input from user for rows and cols
	int ** table1= AllocateMemory(row1,col1);
	InputMatrix(table1, row1, col1);

	int row2, col2;
	//take input from user for rows and cols
	int ** table2= AllocateMemory(row2,col2);
	InputMatrix(table2, row2, col2);
	bool axis;
	cin>>axis;
	/ / before calling concat_table check some conditions of rows and cols
	int ** result = concat_table (table1, table2, row1, col1, row2, col2, axis);
	/ /display result
	//deallocation
}
```

TASK-2:

Write a program to concatenate the data of two-character arrays. Your task is to prompt the user to enter the string (you can take the strings from user in a variable of string data type). Now store this data from the variable string data type to a 1d dynamic character array having the size one greater than the string length (you can use size function of string to get the string length).

Now create a function **StringConcat** that should receive 2-character arrays namely **c_arr1** and **c_arr2** from the main function like (**char * c_arr2**). Remember the return type of this function is void. The main task of this function is to **concatenate c_arr1 and c_arr2.** The concatenated string should be stored in **c_arr1**. It is better to create a **temp 1d dynamic character array** large enough to store the concatenated data.

Once you are done with the concatenation, then point the address of c_arr1 to the temp character array. You need to pass c_arr1 by reference so that the data of 1d dynamic character array of main function should also be updated. **Display the contents of c_arr1 inside the main function**. The output of this statement should be the concatenated data as provided in the sample run. There should not be any memory leak or dangling pointer.

Sample Run:

Input:
c_arr1: Hello Class
c_Arr2: Object Oriented Programming
Output:
c_arr1 after concatenation is:
Hello Class Object Oriented Programming