National University of Computer and Emerging Sciences, Lahore Campus

Course:	Object Oriented Programming	Course Code:	CS-1004
Program:	BS (Software Engineering)	Semester:	Spring 2023
Deadline:	22 Feb 22 (11:59 PM)	Total Marks:	20
Section:	BSE-2C		
	Homework-2		

Instruction/Notes:

1: Late submissions, even after a fraction of second will not be accepted.

2: Any kind of plagiarism will result into 0 marks in all the sessional activities (whether you are provider or receiver)

Task#1: Write a program that fetch only specific information from the original 2D array and store that fetched information in another dynamic 2D array. Declare the following in the main function and perform the tasks accordingly:

1. Double pointer to integer (for original 2D dynamic integer array)
2. Double pointer to integer (for fetched data 2D array)
3. Two Integer variables (rows and columns for original 2D dynamic array)
4. One integer variable (fd_row for the rows count of fetched data 2D array).

In the first task you are going to create a 2D dynamic array in which number of columns in each row will be the same but the number of columns in the fetched data array will not be the same.

- Create a function **allocateMemory** and call this function from the main function. allocateMemory function should receive all the parameters by-reference i.e., by-reference double pointer to integer, by-reference integer variables for rows and columns. Return type of this function should be void and the main task of this function is to get the rows and columns only once from the user then allocate the required memory dynamically to the 2D array.
- Create a function **getData** and call this function from the main function. This function should also receive by-reference parameters but the nature of parameters in this function will be slightly different. You need to receive by-reference const pointer to const pointer to integer (by-reference will make sure that there will be only single copy of double pointer in the main function and in the getData function and remaining part i.e., const pointer to const pointer will make it sure that this double pointer can't point to any other memory location). Similarly, you need to receive by-reference const integers for rows and columns. This function should be used to get the input from user and accept only +ve integers greater than equal 10 as a valid input.
  (int * const * const &p) => 'p' is a by-reference const pointer to const pointer to integer.

Q: why we are not writing const before the very first 'int' in the expression? what will happen if we also write const before the very first 'int' in the expression?

- Now create another function **fetchData** and call this function from the main function. This function should receive all by-reference parameters i.e., by-reference const pointer to const pointer to const integer, by-reference const integers for rows and columns and a by-reference integer (fd_rows). This function should return a dynamic 2D array through return statement, so you need to write appropriate return type of this function. This function should perform the following tasks.
  - Create another 2D dynamic integer array namely (fd_Arr i.e., fetched data array) having 3 rows and varying length of columns in each row so you can initialize the fd_rows with 3 (fd_rows will be basically holding the information about the rows in fd_Arr). Since you have no clue about the columns in each row so skip the memory allocation for columns at this stage.

- This newly created 2D dynamic array (fd_Arr) should **store all the "even values" in the "odd columns" of original 2D array in the very first row of fd_Arr.** Similarly **store all the "odd values" in the even columns of the original 2D array in the second row of fd_Arr**. You need to **store all the prime numbers of the original array in the third row of fd_Arr**.
- Hint! First you need to find out the count of required data so that you can allocate the required amount of memory in the corresponding row. Don't forget to allocate one extra column in each row to store the sentinel value (-1) which will indicate the end of list. Once you allocate the required memory you need to iterate again to store the data and store '-1' in the last column.
- Since the function is returning a dynamic 2D array so you need to receive a double pointer in the main function.

- Now create a function **printData** and call this function from the main function. Make sure that all the parameters of this function should be by-reference const. This function should receive original 2D array, rows and columns and print the data on console.
- Create another function **displayData** and call this function from the main function. This function should receive the information of fetched data i.e., number of rows in the fetched data array and a double pointer pointing to fetched data array. Print the data of fetched data array row wise. We know that we have stored even values in the first row, odd values in second row and prime values in the third column.
- Create a function **releaseResources** and pass all the pointers pointing to dynamically allocated memory. This function should deallocate the memory and relocate the pointers to NULL.

Task#2:

ICC (International cricket council) has announced a cash prize of US ($2000) for all those players who have achieved a milestone of 300 runs i.e., scored more than 300 runs in a recently conducted ICC event. They have already shortlisted the teams in which there exist at-least one players who has achieved the milestone like the sample data shown in the figure 1. There might be a possibility that many players of a single side achieved that milestone so it is not practically possible to reward everyone so they have decided that a maximum of 5 top scorers of each shortlisted team will be considered that is why there are exactly 5 entries in each team. You can see that in the figure1 that there are few scores less than the threshold value so ICC is interested to represent the information in a more meaningful way so that they can only display the scores of only qualifying players i.e., (discard the scores that are failed to meet the required milestone and keep only those scores that are greater than or equal to the threshold value). Your task is to fetch and store only the scores fulfilling the required criteria from each team in 2D dynamic array where each row should store the scores of the players from a particular country like the sample data shown in the figure 2. You need to implement a function that should receive a 2D integer array namely "**Scores**" (already filled with the data), rows and columns of the existing 2D array, and by reference double pointer to integer namely "**Achievers**" to store the required data. This function should allocate the memory and store the data in a 2D array (**Achievers**) from the given 2D array of **Scores**. You are supposed to allocate exactly the required amount of memory to the **Achievers** as shown in the figure 2. The data provided in the figure is just to demonstrate the problem you are required to provide generic implementation of the function. The function prototype is provided for the reference.

- Declare a double pointer to integer (**scores**), two integer variables (**row, and column**), and a double pointer to integer (**achievers**)
- You can initialize the memory and fill the array in the main function. You don't need to get the values of rows, columns, and data from user. you can assign any hard code value to rows and columns according to the number records and read the data from a text file (scores.txt) rather getting the input from user. To test the functionality of your program you can copy the exact data in the text file and make sure that once

a 2D array (**scores)** filled with the data the sequence of values should be look like the same as shown in the figure. Number of rows in both the arrays i.e., (**scores** and **achievers**) will be the same.

- Call the function **getAchieversList** from the main function and pass the required parameters. Implement the required functionalities in the function. Function prototype is provided for your reference.
  void getAchieversList(const int ** scores, const int rows, const int cols, int **& achievers)
- Create two functions (**print** and **display**). Print should receive (**scores, rows, and columns)** and print the data on console whereas display should receive (**achievers**, and **rows**) and print the data on console.

Visual representation of the task

Scores (A 2D dynamic array of 'r' no of rows, and 'c' no of columns already filled with the data)

Figure 1: representing the data in **scores.**

Australia	England	India	Pakistan	New Zealand
198	352	240	286	332
362	281	290	392	218
249	408	374	346	341
426	451	160	132	412
186	195	248	482	486

Achievers (A 2D dynamic array of 'r' no of rows and varying length of column in each row where a sentinel value "-1" indicating the end of list in each row).

Figure 2: representing the data in **achievers** fetched from **scores.**

Australia	362	426	-1		
England	352	408	451	-1	
India	340	-1			
Pakistan	392	346	482	-1	
New Zealand	332	341	412	486	-1

You need to fetch the information from **scores** into **achievers in the following way.** Such that every row of achievers will store the data of a specific team players. You can see that the order of arrangement of data in **scores** and **achievers** is different. In **scores** data of specific team players was stored in columns whereas in achievers the data of specific team player is stored in the rows.

Hint! You need to find out the amount of memory required to be allocated in each row of achievers first, allocate the memory and then copy the data in each row.