Graded lab 3B - 18:15-19:45 (8 points)

In this task you will be implementing classes for representing mathematical vectors and matrices. You are provided with header files for both of those classes. Each part contains a description of what needs to be implemented as well as example usages of this particular functionality in the main.cpp file. The main.cpp file cannot be modified (apart from uncommenting the code) and the functions you implement should fit the example usages in it. Your code should produce the same output as provided in the output.txt file.

After finishing the task make sure to upload your solution to Teams as a single .zip file. The .zip file should contain only .h and .cpp files.

IMPORTANT: The main topic of this task is dynamic allocations so make sure that there are no memory leaks!

Part 1 (1.5 point)

Implement the following functions for the Vector class:

- Vector(int size, int* elements = nullptr) - constructs a vector of given size filled with passed elements or zeroes if elements == nullptr
- Destructor of the Vector class
- Output stream operator matching the output from output.txt
- Bracket operators allowing for getting and modifying specific values inside the vector (assume the index is correct)

Part 2 (1 point)

Implement both the copy constructor and the copy assignment operator for the Vector class.

Part 3 (1 point)

Implement the postfix decrement operator and the prefix increment operator for the Vector class. The decrement operator should make the Vector shorter by one element (all the other values need to be preserved). The increment operator should make the Vector longer by one element (the new element has a value of 0).

Part 4 (1.5 points)

Implement the following functions for the Matrix class:

- Matrix(int rows_count, int columns_count) - constructs a matrix of given size filled with zeroes
- Destructor of the Matrix class
- Output stream operator matching the output from output.txt
- Bracket operators allowing for getting and modifying specific values inside the matrix (assume the index is correct)

Part 5 (1 point)

Implement both the copy constructor and the copy assignment operator for the Matrix class.

Part 6 (1 point)

Implement the prefix decrement operator for the Matrix class. The operator should remove the last row of the matrix (all the other elements are preserved).

Part 7 (1 point)

Implement the multiplication operator for the Matrix class. It should perform the standard matrix multiplication operation on the arguments and return the result as a pointer to a newly allocated matrix. If matrix sizes are not compatible return a nullptr.