პროგრამირების აბსტრაქციები

სემინარის ამოცანები #17

მატრიცის კლასის იმპლემენტაცია

დაწერეთ მატრიცის კლასი, რომელსაც შეეძლება ნებისმიერი ზომის მატრიცის შენახვა. ამ კლასს უნდა ჰქონდეს მატრიცის თითოეულ უჯრაში მონაცემების ჩაწერის და წაკითხვის საშუალება. ასევე უნდა დაწეროთ მატრიცების შეკრების და გამრავლების მეთოდები. იმისთვის რომ მეთოდებიდან პასუხად მატრიცის დააბრუნება შეძლოთ ასევე დაგჭირდებათ ე.წ. "copy constructor"-ის დაწერა.

header file

```
#ifndef MATRIX_CLASS_H_
#define MATRIX_CLASS_H_

class Matrix {
private:
    int w, h;
    double* data;
public:
    Matrix(int width, int height);
    Matrix(const Matrix& other);
    ~Matrix();
    double getAt(int row, int col);
    void setAt(int row, int col, double val);
    void addAt(int row, int col, double val);
    Matrix add(Matrix& other);
    Matrix times(Matrix& other);
};

#endif
```

source file

```
#include "MatrixClass.h"

Matrix::Matrix(int width, int height) {
    w = width; h = height;
    data = new double[w*h];
    for (int i = 0; i < w*h; i++)
        data[i] = 0.0;
}

Matrix::Matrix(const Matrix& other) {
    w = other.w; h = other.h;
    data = new double[w*h];
    for (int i = 0; i < w*h; i++)
        data[i] = other.data[i];
}

Matrix::~Matrix() {
    delete[] data;
}

double Matrix::getAt(int row, int col) {
    return data[row*w + col];
}

void Matrix::setAt(int row, int col, double val) {
    data[row*w + col] = val;
}

void Matrix::addAt(int row, int col, double val) {
    data[row*w + col] += val;
}

Matrix Matrix::add(Matrix& other) {
    Matrix result(w, h);
    for (int i = 0; i < w*h; i++)
        result.data[i] = data[i] + other.data[i];
    return result;
}

Matrix Matrix::times(Matrix& other) {
    Matrix result(other.w, h);
    for (int row = 0; row < h; row++) {
        for (int col = 0; col < other.w; col++) {
            for (int k = 0; k < w; k++)
                result.addAt(row, col, getAt(row, k) * other.getAt(k, col));
        }
    }
    return result;
}
```