МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

Дисципліна «Автоматизоване тестування»

Лабораторна робота № 3: «Тестові дублі»

Виконав студент гр. 8.1214

Козловський Іван

Перевірив: Викладач

Лебідь Максим Володимирович

Запоріжжя

2024

## ЗАВДАННЯ

1. Ознайомитись з програмною реалізацією алгоритму розв’язання СЛАР https://github.com/LiannaAsatryan/gauss\_js
2. Реалізувати mock методи класу Matrix.
3. Створити github репозиторій з тестами.

Посилання на github: <https://github.com/IvanK5565/AutoTest/tree/master/lab3>

Спосіб використання бібліотеки gauss\_js показана у файлі main.js. Почерзі викликаються методи read\_input, read\_golden, gauss. Клас Matrix використовується у кожному з них. Для цих методів створено тести з mock методами класу Matrix:

describe("Test input", function () {

input\_file = "3\n3 6 9 15\n8 8 40 48\n7 35 49 56\n"

golden\_file = '1 2 3'

var fs\_stub;

beforeEach(() => {

fs\_stub = sinon.stub(fs, "readFileSync");

});

afterEach(() => {

fs\_stub.restore();

});

//read\_input

it("test read\_input", () => {

fs\_stub.withArgs('input.txt', 'utf8').returns(input\_file);

m = my\_lib.read\_input();

assert.equal(m.get\_cols(),4)

assert.equal(m.get\_rows(),3)

})

//read\_golden

it("test read\_golden", () => {

fs\_stub.withArgs('golden.txt', 'utf8').returns(golden\_file);

golden = my\_lib.read\_golden();

assert.deepEqual(['1','2','3'],golden)

})

});

describe("Gauss tests", function() {

it("test gauss\_forward", function() {

matrix\_instance = new matrix(2)

const matrix\_mock = sinon.mock(matrix\_instance);

matrix\_mock.expects('get\_rows').once().returns(2);

matrix\_mock.expects('get\_cols').once().returns(3);

matrix\_mock.expects('get').withArgs(0, 0).atLeast(1).returns(3);

matrix\_mock.expects('get').withArgs(1, 0).returns(8);

matrix\_mock.expects('mull\_add').withArgs(1,0,-8/3).once();

matrix\_mock.expects('swap\_with\_nonzero\_row').never();

my\_lib.gauss\_forward(matrix\_instance);

matrix\_mock.verify();

})

//gauss\_backward

it("test gauss\_backward", function() {

matrix\_instance = new matrix(2)

const matrix\_mock = sinon.mock(matrix\_instance);

matrix\_mock.expects('get\_rows').once().returns(2);

matrix\_mock.expects('get\_cols').once().returns(3);

matrix\_mock.expects('get').withArgs(1, 2).once().returns(40);

matrix\_mock.expects('get').withArgs(1, 1).once().returns(8);

matrix\_mock.expects('get').withArgs(0, 1).once().returns(6);

matrix\_mock.expects('get').withArgs(0, 2).once().returns(9);

matrix\_mock.expects('get').withArgs(0, 0).once().returns(3);

my\_lib.gauss\_backward(matrix\_instance);

matrix\_mock.verify();

})

//gauss

it("test gauss", () => {

matrix\_instance = new matrix(2)

const matrix\_mock = sinon.mock(matrix\_instance);

result = my\_lib.gauss(matrix\_instance);

assert.ok(result==null);

matrix\_mock.verify();

});

});

