МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

Дисципліна «Автоматизоване тестування»

Лабораторна робота №1:

Виконав студент гр. 8.1214

Ніценко Вадим Андрійович

Перевірив Викладач

Лебідь Максим Володимирович

Запоріжжя

2024

Завдання

1. Ознайомитись з бібліотекою операцій над матрицями <https://github.com/zhufuge/Mtrx>
2. Виконати тестування основних функцій бібліотеки засобами бібліотек Mocha Chai.
3. Створити github репозиторій з тестами.

Хід роботи

Встановлюємо бібліотеки Mocha та Chai

npm install --save-dev mocha

npm install chai

Використовував версію Chai 4.5.0



Версія Mocha 10.7.3

Встановлюємо бібліотеку яку будемо тестувати

npm install mtrx

Додаємо залежності

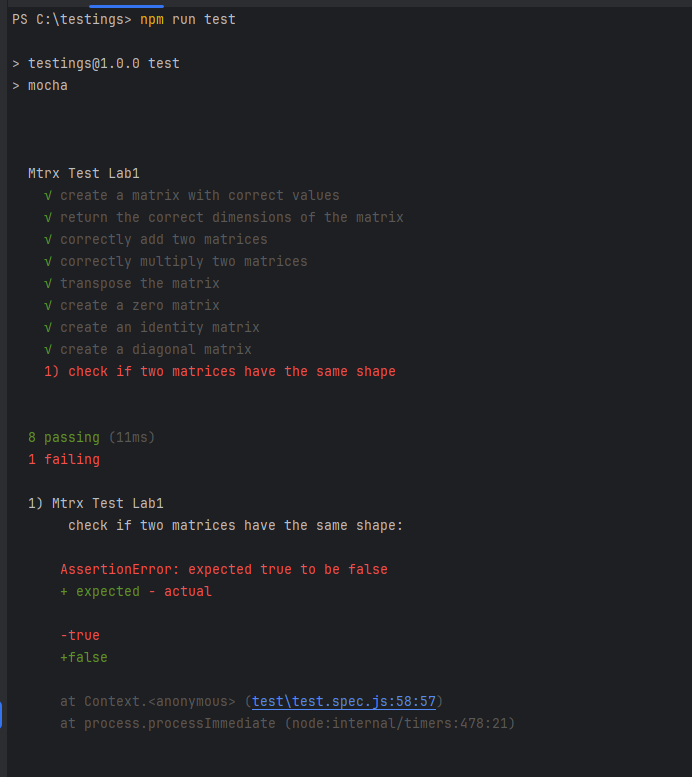
const { expect } = require('chai');

const Mtrx = require('mtrx');

Код тестів

const { *expect* } = require('chai');  
const *Mtrx* = require('mtrx');  
  
  
*describe*('Mtrx Test Lab1', function () {  
  
 *it*('create a matrix with correct values', () => {  
 const *matrix* = new *Mtrx*([[2, 4], [6, 8]]);  
 *expect*(*matrix*).to.deep.equal(new *Mtrx*([[2, 4], [6, 8]]));  
 });  
  
 *it*('return the correct dimensions of the matrix', () => {  
 const *matrix* = new *Mtrx*([[1, 2, 3], [4, 5, 6]]);  
 *expect*(*matrix*.rows).to.equal(2);  
 *expect*(*matrix*.cols).to.equal(3);  
 });  
  
 *it*('correctly add two matrices', () => {  
 const *matrix1* = new *Mtrx*([[1, 2], [3, 4]]);  
 const *matrix2* = new *Mtrx*([[5, 6], [7, 8]]);  
 const *result* = *matrix1*.add(*matrix2*);  
 *expect*(*result*).to.deep.equal(new *Mtrx*([[6, 8], [10, 12]]));  
 });  
  
 *it*('correctly multiply two matrices', () => {  
 const *matrix1* = new *Mtrx*([[1, 2], [3, 4]]);  
 const *matrix2* = new *Mtrx*([[2, 0], [1, 2]]);  
 const *result* = *matrix1*.mul(*matrix2*);  
 *expect*(*result*).to.deep.equal(new *Mtrx*([[4, 4], [10, 8]]));  
 });  
  
 *it*('transpose the matrix', () => {  
 const *matrix* = new *Mtrx*([[1, 2], [3, 4]]);  
 const *result* = *matrix*.T();  
 *expect*(*result*).to.deep.equal(new *Mtrx*([[1, 3], [2, 4]]));  
 });  
  
 *it*('create a zero matrix', () => {  
 const *zeroMatrix* = *Mtrx*.*zeros*(2, 3);  
 *expect*(*zeroMatrix*).to.deep.equal(new *Mtrx*([[0, 0, 0], [0, 0, 0]]));  
 });  
  
 *it*('create an identity matrix', () => {  
 const *identityMatrix* = *Mtrx*.*eye*(3);  
 *expect*(*identityMatrix*).to.deep.equal(new *Mtrx*([[1, 0, 0], [0, 1, 0], [0, 0, 1]]));  
 });  
  
 *it*('create a diagonal matrix', () => {  
 const *diagMatrix* = *Mtrx*.*diag*([1, 2, 3]);  
 *expect*(*diagMatrix*).to.deep.equal(new *Mtrx*([[1, 0, 0], [0, 2, 0], [0, 0, 3]]));  
 });  
  
 *it*('check if two matrices have the same shape', () => {  
 const *matrix1* = new *Mtrx*([[1, 2], [3, 3]]);  
 const *matrix2* = new *Mtrx*([[5, 6], [7, 7]]);  
 const *matrix3* = new *Mtrx*([[1, 2, 3], [4, 5, 6]]);  
  
 *expect*(*Mtrx*.*isSameShape*(*matrix1*, *matrix2*)).to.be.false;  
 *expect*(*Mtrx*.*isSameShape*(*matrix1*, *matrix3*)).to.be.false;  
 });  
  
  
});

Результати



Контрольні запитання

Наведіть види тестування

* Модульне тестування
* Інтеграційне тестування
* Системне тестування
* Приймальне тестування
* Функціональне тестування
* Регресійне тестування
* Димове тестування
* Альфа і бета тестування

Особливості модульного та приймального тестування

Модульне тестування:

* Перевіряє окремі компоненти або модулі коду
* Виконується розробниками
* Фокусується на технічних деталях реалізації
* Допомагає виявити помилки на ранніх етапах розробки
* Зазвичай автоматизоване

Приймальне тестування:

* Перевіряє відповідність системи вимогам замовника
* Виконується замовником або кінцевими користувачами
* Фокусується на функціональності з точки зору користувача
* Проводиться на завершальному етапі розробки
* Може бути як ручним, так і автоматизованим

Призначення бібліотек Mocha та Chai

Mocha:

* Фреймворк для тестування JavaScript
* Дозволяє організовувати та запускати тести
* Підтримує асинхронне тестування
* Надає функції для опису тестових сценаріїв

Chai:

* Бібліотека тверджень для Node.js і браузера
* Використовується разом з фреймворками тестування
* Надає різні стилі тверджень (expect, should, assert)
* Дозволяє писати зрозумілі та виразні тести