NATIONAL TECHNICAL UNIVERSITY OF UKRAINE

“Igor Sikorsky KYIV POLYTECHNIC INSTITUTE”

Faculty of Applied Mathematics

Department of Computer Systems Software

Course project

of the discipline "Technologies of Software System Design"

titled

““An Application For Array Sorting and Matrix Operations””

Done by Guo Wanyu

student of study group КП-92M

Project curator

senior teacher Ruslan Hadyniak

Points

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(signature)

Kyiv 2019

1. **The task variant 3**
2. **The requirements analysis 3**
3. **The code listing 4**
   1. **array.json 4**
   2. **matrixA.json & matrix.json 4**
   3. **calculation.js 4**
   4. **sort.js 5**
   5. **Main.js 6**
   6. **matrixCal.js 7**
   7. **matrixMain.js 10**
4. **The results 11**
5. **The summary 14**

**The task**

1. **Course Project Task Variant**

* User interaction type: **command line arguments**
* Matrix operations: **subtracting, multiplication, transpose**
* Sorting algorithm: **bubble**

**Requirements and Analysis**

In this section, project’s requirements will be determined and related analysis will be considered. The purpose of this project is to perform predetermined mathematical operations and sort using the user's matrices and array.  
  
 To be easy to understand, sort function, matrices, array and the main program will be in separate titles.

The whole project is have eight different files.

Main files: “matrixMain.js” “main.js”

Function files: “sort.js” “matrixCal.js” “calculation.js”

Data files: “array.json” “matrixA.json” “matrix.json”

**The code listing**

**1.array.json**

Array data

[1,3,2,8,9,1,5]

**2.matrixA.json & matrix.json**

Matrix data

MatrixA

[

  [14,21,3],

  [65,40,6],

  [1,79,8]

]

MatrixB

[

  [-10,12,5],

  [32,-7,0],

  [2,-78,13]

]

**3.calculation.js**

Array data calculation module

module.exports = class Calculation {

    constructor(array) {

        this.array = array;

    }

Array data sum module

    sum() {

        console.log("===[" + this.array + "]sum start===")

        let sum = 0;

        for (let i = 0; i < this.array.length; i++) {

            sum += this.array[i];

        }

        console.log("===[" + this.array + "]sum end, sum : " + sum + "===")

        return sum

    }

}

**4.sort.js**

Sorting Algorithm Module

Bubble sort

bubble() {

        console.log("===[" + this.array + "]bubble sort start===")

        let i = this.array.length, j

        let cnt = 0

        while (i > 0) {

            for (j = 0; j < i - 1; j++) {

                if (this.array[j] > this.array[j + 1]) {

                    cnt++

                    this.print("==bubble sort process " + cnt + "==", j, j + 1)

                    this.exchange(j, j + 1)

                }

            }

            i--

        }

        console.log("===[" + this.array + "]bubble sort finish, exchange " + cnt + " times===")

        return this.array;

    }

Insert sort

    insert() {

        console.log("===[" + this.array + "]insert sort start===")

        if (this.array == null | this.array.length < 2) {

            return this.array

        }

        let cnt = 0

        for (let i = 0; i < this.array.length; i++) {

            for (let j = i - 1; j >= 0 && this.array[j] > this.array[j + 1]; j--) {

                cnt++

                this.print("==insert sort process " + cnt + "==", j, j + 1)

                this.exchange(j, j + 1)

            }

        }

        console.log("===[" + this.array + "]insert sort finish, exchange " + cnt + " times===")

        return this.array;

    }

Select sort

    select() {

        console.log("===[" + this.array + "]select sort start===")

        if (this.array == null | this.array.length < 2) {

            return this.array

        }

        let cnt = 0

        for (let i = 0; i < (this.array.length - 1); i++) {

            let minIndex = i;

            for (let j = i + 1; j < this.array.length; j++) {

                minIndex = this.array[j] < this.array[minIndex] ? j : minIndex

            }

            cnt++

            this.print("==select sort process " + cnt + "==", i, minIndex)

            this.exchange(i, minIndex)

        }

        console.log("===[" + this.array + "]select sort finish, exchange " + cnt + " times===")

        return this.array;

    }

Internal data exchange value

    exchange(i, j) {

        let a = this.array[i]

        this.array[i] = this.array[j]

        this.array[j] = a

    }

Sort process print

    print(str, i, j) {

        console.log(str, 'array array[' + i + ']:' + this.array[i] + ' and array[' + j + ']:' + this.array[j] + ' exchange')

    }

}

**5.Main.js**

Array main program entry

Print results are raw data - data summation - bubble sort - insert sort - select sort

const sourceArray = require('./array.json')

console.log("====source array:" + sourceArray)

// calculation

const sumArray = JSON.parse(JSON.stringify(sourceArray))

const Calculation = require('./calculation.js')

const myCalculation = new Calculation(sumArray)

myCalculation.sum()

// bubble sort

const sortArray = JSON.parse(JSON.stringify(sourceArray))

const Sort = require('./sort.js')

const mySort = new Sort(sortArray)

mySort.bubble()

// insert sort

const insertArray = JSON.parse(JSON.stringify(sourceArray))

const insertSort = new Sort(insertArray)

insertSort.insert()

// select sort

const selectArray = JSON.parse(JSON.stringify(sourceArray))

const selectSort = new Sort(selectArray)

selectSort.select()

**6.matrixCal.js**

Matrix Algorithm Module

module.exports = class matrixCal {

    constructor(matrixA, matrixB) {

        this.matrixA = matrixA;

        this.matrixB = matrixB;

    }

Matrix add

    add() {

        console.log("matrix add start===")

        let matrix = [];

        if (!this.matrixAddOrSubtractionCheck()) {

            console.log("current matrixs can't add ===")

            return matrix;

        }

        let row = this.matrixA.length;

        let col = this.matrixA[0].length;

        for (let i = 0; i < row; i++) {

            let array = [];

            for (let j = 0; j < col; j++) {

                array.push(this.matrixA[i][j] + this.matrixB[i][j])

            }

            matrix.push(array);

        }

        console.log("===matrix add end, result :");

        this.print(matrix);

        return matrix;

    }

Matrix subtraction

    subtraction() {

        console.log("matrix subtraction start===")

        let matrix = [];

        if (!this.matrixAddOrSubtractionCheck()) {

            console.log("current matrixs can't subtraction ===")

            return matrix;

        }

        let row = this.matrixA.length;

        let col = this.matrixA[0].length;

        for (let i = 0; i < row; i++) {

            let array = [];

            for (let j = 0; j < col; j++) {

                array.push(this.matrixA[i][j] - this.matrixB[i][j])

            }

            matrix.push(array);

        }

        console.log("===matrix subtraction end, result :");

        this.print(matrix);

        return matrix;

    }

Matrix multiplication

    multiplication() {

        console.log("matrix multiplication start===")

        let matrix = [];

        if (!this.matrixMultiplicationCheck()) {

            console.log("current matrixs can't multiplication ===")

            return matrix;

        }

        let row = this.matrixA.length;

        let col = this.matrixA[0].length;

        for (let i = 0; i < row; i++) {

            let array = [];

            for (let j = 0; j < col; j++) {

                let value = 0;

                for (let k = 0; k < col; k++) {

                    value += this.matrixA[i][k] \* this.matrixB[k][j];

                }

                array.push(value)

            }

            matrix.push(array);

        }

        console.log("===matrix multiplication end, result :");

        this.print(matrix);

        return matrix;

    }

Matrix transposition

    transposition(sourceMatrix){

        console.log("matrix transposition start===source:")

        this.print(sourceMatrix);

        let matrix = [];

        let row = sourceMatrix.length;

        let col = sourceMatrix[0].length;

        for (let i = 0; i < col; i++) {

            let array = [];

            for (let j = 0; j < row; j++) {

                array.push(sourceMatrix[j][i])

            }

            matrix.push(array);

        }

        console.log("===matrix transposition end, result :");

        this.print(matrix);

        return matrix;

    }

Matrix print

    print(matrix) {

        let h;

        for (let i = 0; i < matrix.length; i++) {

            let array = matrix[i];

            h = '';

            for (let j = 0; j < array.length; j++) {

                h += array[j] + '\t';

            }

            console.log(h)

        }

    }

Matrix add or subtraction check

    matrixAddOrSubtractionCheck(){

        if(this.matrixA.length==0||this.matrixB.length==0||this.matrixA[0].length==0 || this.matrixB[0].length==0){

            return 0;

        }

        let row = this.matrixA.length;

        let col = this.matrixA[0].length;

        if(this.matrixB.length!=row){

            return 0;

        }

        for(let i=0;i<row;i++){

            if(this.matrixA[i]==undefined||this.matrixB[i]==undefined||this.matrixA[i].length!=col||this.matrixB[i].length!=col){

                return 0;

            }

        }

        return 1;

    }

Matrix multiplication check

    matrixMultiplicationCheck(){

        if(this.matrixA.length==0||this.matrixB.length==0||this.matrixA[0].length==0 || this.matrixB[0].length==0){

            return 0;

        }

        let rowA = this.matrixA.length;

        let colA = this.matrixA[0].length;

        for(let i=0;i<rowA;i++){

            if(this.matrixA[i]==undefined||this.matrixA[i].length!=colA){

                return 0;

            }

        }

        let rowB = this.matrixB.length;

        let colB = this.matrixB[0].length;

        for(let i=0;i<rowB;i++){

            if(this.matrixB[i]==undefined||this.matrixB[i].length!=colB){

                return 0;

            }

        }

        if(rowA!=colB||rowB!=colA){

            return 0;

        }

        return 1;

    }

}

**7.matrixMain.js**

Matrix main program entry

Print raw data - matrix addition - matrix subtraction - matrix multiplication - matrixA transposition

const matrixA = require('./matrixA.json')

const matrixB = require('./matrixB.json')

const MatrixCal = require('./matrixCal')

const myMatrixCal = new MatrixCal(matrixA,matrixB)

console.log('===print first matrix===')

myMatrixCal.print(matrixA)

console.log('===print first matrix end===')

console.log('===print second matrix===')

myMatrixCal.print(matrixB)

console.log('===print second matrix end===')

myMatrixCal.add()

myMatrixCal.subtraction()

myMatrixCal.multiplication()

myMatrixCal.transposition(matrixA)

**The results**

**Run main.js to get array results**

Print raw array

====source array:1,3,2,8,9,1,5

Array sum

===[1,3,2,8,9,1,5]sum start===

===[1,3,2,8,9,1,5]sum end, sum : 29===

Bubble sort

===[1,3,2,8,9,1,5]bubble sort start===

==bubble sort process 1== array array[1]:3 and array[2]:2 exchange

==bubble sort process 2== array array[4]:9 and array[5]:1 exchange

==bubble sort process 3== array array[5]:9 and array[6]:5 exchange

==bubble sort process 4== array array[3]:8 and array[4]:1 exchange

==bubble sort process 5== array array[4]:8 and array[5]:5 exchange

==bubble sort process 6== array array[2]:3 and array[3]:1 exchange

==bubble sort process 7== array array[1]:2 and array[2]:1 exchange

===[1,1,2,3,5,8,9]bubble sort finish, exchange 7 times===

Insert sort

===[1,3,2,8,9,1,5]insert sort start===

==insert sort process 1== array array[1]:3 and array[2]:2 exchange

==insert sort process 2== array array[4]:9 and array[5]:1 exchange

==insert sort process 3== array array[3]:8 and array[4]:1 exchange

==insert sort process 4== array array[2]:3 and array[3]:1 exchange

==insert sort process 5== array array[1]:2 and array[2]:1 exchange

==insert sort process 7== array array[4]:8 and array[5]:5 exchange

===[1,1,2,3,5,8,9]insert sort finish, exchange 7 times===

Select sort

===[1,3,2,8,9,1,5]select sort start===

==select sort process 1== array array[0]:1 and array[0]:1 exchange

==select sort process 2== array array[1]:3 and array[5]:1 exchange

==select sort process 3== array array[2]:2 and array[2]:2 exchange

==select sort process 4== array array[3]:8 and array[5]:3 exchange

==select sort process 5== array array[4]:9 and array[6]:5 exchange

==select sort process 6== array array[5]:8 and array[5]:8 exchange

===[1,1,2,3,5,8,9]select sort finish, exchange 6 times===

**Run matrixMain.js to get matrix calculate results**

Print matrixA & matrixB.

===print first matrix===

14 21 3

65 40 6

1 79 8

===print first matrix end===

===print second matrix===

-10 12 5

32 -7 0

2 -78 13

===print second matrix end===

Matrix addition

matrix add start===

===matrix add end, result :

4 33 8

97 33 6

3 1 21

Matrix subtraction

matrix subtraction start===

===matrix subtraction end, result :

24 9 -2

33 47 6

-1 157 -5

Matrix multiplication

matrix multiplication start===

===matrix multiplication end, result :

538 -213 109

642 32 403

2534 -1165 109

Matrix transposition

matrix transposition start===source:

14 21 3

65 40 6

1 79 8

===matrix transposition end, result :

14 65 1

21 40 79

3 6 8

**Summery**

The purpose of the project is to create an write code using JavaScript to implement matrix operations (subtraction, multiplication, transposition ) and bubble sort algorithm.