

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Факультет прикладної математики

Кафедра системного програмування і спеціальних комп’ютерних систем

**Лабораторна робота №2**

з дисципліни “Паралельні та розподілені обчислення”

Тема: «Робота з компіляторами мов С та Java в режимі командного рядка»

Виконав:

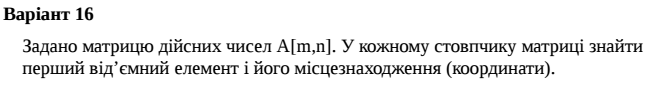
студент 2 курсу групи КВ-41:

Мелащенко Олександр

**Перевірено:**

Київ 2016

**Завдання**



**Реалізація на мові Java**

**<Main.java>**

public class Main

{

private static final int m = 3;

private static final int n = 5;

private static Random rand = new Random();

public static void main(String[] args)

{

int[][] arr = new int[m][n];

for(int i = 0; i < m; i++)

for(int j = 0; j < n; j++)

arr[i][j] = rand.nextInt(201) - 100;

MatrixOutput.show(arr, n, m);

LinkedList<Coordinates> coordinates = FindAFirstNegativeElementInTheColumns.Find(arr, n, m);

System.out.println();

for(Coordinates coord : coordinates)

System.out.println(coord);

}

}

**<** **Coordinates.java>**

public class Coordinates{

private int x, y;

public Coordinates(int \_x, int \_y) {

x = \_x;

y = \_y;

}

@Override

public String toString() {

return "x=" + x + " y=" + y + ";";

}

}

**<** **MatrixOutput.java>**

public class MatrixOutput{

public static void show (int[][] arr, int sizeX, int sizeY) {

for(int i = 0; i < sizeY; i++) {

for (int j = 0; j < sizeX; j++)

System.out.printf("%4d", arr[i][j]);

System.out.println();

}

}

}

**<Makefile>**

JFLAGS = -g

JC = javac

JVM= java

JAR= jar

MANI = main.manifest

FILE = Test.jar

.SUFFIXES: .java .class .jar

.java.class:

$(JC) $(JFLAGS) $\*.java

CLASSES = \

Main.java \

Coordinates.java \

FindAFirstNegativeElementInTheColumns.java \

MatrixOutput.java

MAIN = Main

default: clean classes run manifest jar jarRun

classes: $(CLASSES:.java=.class)

run: $(MAIN).class

$(JVM) $(MAIN)

jarRun:

$(JVM) -$(JAR) $(FILE)

manifest:

touch $(MANI)

echo "Main-Class: Main" > $(MANI)

jar:

$(JAR) cfm $(FILE) $(MANI) \*.class

clean:

$(RM) \*.class

$(RM) \*.jar

**Реалізація на мові С**

**<main.c>**

#include <stdio.h>

#include "TypeDefs.h"

#include "InitArray.h"

#include "PrintArray.h"

#include "Coordinates.h"

#include "HandleArray.h"

#define DEBUG

int main(){

int decision;

printf("Chose type of initialization:\n");

printf("1. By Random\n");

printf("2. By user\n");

printf("3. Ordered\n\n");

scanf("%d", decision);

void(\*init)(MyArray arr, int sizeX, int sizeY);

switch (decision) {

case 1:

init = &InitByRandom;

break;

case 2:

init = &InitByUser;

break;

default:

init = &InitOrdered;

break;

}

MyArray arr;

init(arr, n, m);

println();

pretty\_print(arr, n, m);

println();

#ifdef DEBUG

Find\_debug(arr, n, m);

#else

for (Coordinates \*coord : Find(arr, n, m))

coord->print();

#endif

}

**<** **Coordinates.c>**

#include "Coordinates.h"

Coordinates::Coordinates(int \_x, int \_y){

x = \_x;

y = \_y;

}

void print(){

cout << "x="<< x << " y=" << y << ";";

}

**<** **HandleArray.c>**

#include "HandleArray.h"

Coordinates\* FindNegative(MyArray arr, int coordX, int sizeY)

{

for (int i = 0; i < sizeY; i++)

if (arr[i][coordX] < 0) return new Coordinates(coordX, i);

return nullptr;}

MyVector Find(MyArray arr, int sizeX, int sizeY){

MyVector vec;

Coordinates \*coord;

for (int i = 0; i < sizeX; i++) {

coord = FindNegative(arr, i, sizeY);

if (coord)

vec.push\_back(coord);

}

return vec;

}

MyVector Find\_debug(MyArray arr, int sizeX, int sizeY){

MyVector vec;

Coordinates \*coord;

for (int i = 0; i < sizeX; i++) {

printf("Check %d column\n", i);

coord = FindNegative(arr, i, sizeY);

if (coord) {

printf("Find in %d row\n", coord->y);

vec.push\_back(coord);

} }

return vec;}

**<** **InitArray.h>**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "InitArray.h"

void InitByRandom(MyArray arr, int sizeX, int sizeY){

for (int i = 0; i < m; i++)

for (int j = 0; j < n; j++)

arr[i][j] = firs\_value + rand() % last\_value;

}

void InitByUser(MyArray arr, int sizeX, int sizeY){

for (int i = 0; i < m; i++)

for (int j = 0; j < n; j++) {

printf("Array[%d][%d]=", i, j);

scanf("%d", &arr[i][j]);

}

}

void InitOrdered(MyArray arr, int sizeX, int sizeY){

int first\_ordered\_value = -n\*m / 2;

for (int i = 0; i < m; i++)

for (int j = 0; j < n; j++)

arr[i][j] = first\_ordered\_value++;

}

**<Makefile>**

CC=g++

CFLAGS=-c -Wall

LDFLAGS=

SOURCES=main.cpp Coordinates.cpp HandleArray.cpp InitArray.cpp PrintArray.cpp

OBJECTS=$(SOURCES:.cpp=.o)

EXECUTABLE=Array

all: $(SOURCES) $(EXECUTABLE)

$(EXECUTABLE): $(OBJECTS)

$(CC) $(LDFLAGS) $(OBJECTS) -o $@

.cpp.o:

$(CC) $(CFLAGS) $< -o $@

clean:

rm -rf \*.o