

Міністерство освіти і науки України

НТУУ «Київський політехнічний інститут»

Факультет прикладної математики

**Лабораторна робота №2**

**з дисципліни: «Паралельні та розподілені обчислення»**

**Варіант №5**

**Виконав:**

Студент групи КВ-73

Гриневицький Іван

**Перевірив:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Київ 2019

**Постановка завдання для програми мовою C**

1. Написати програму розв’язання задачі пошуку (за варіантом) у двовимірному масиві (матриці) одним з алгоритмів методу лінійного пошуку.

2. Розміри матриці m та n взяти самостійно у межах від 7 до 10.

3. Програма обов'язково повинна бути написана і структурована наступним чином:

a) оголошення структур даних (typedef) повинно бути зроблено у окремому заголовочному файлі;

b) повинно бути щонайменше три файли із вихідним кодом (не враховуючи необхідні заголовочні файли), що міститимуть реалізації функцій введення (випадкові значення наперед сортовані значення, з клавіатури), обробки, та виведення на друк (pretty\_print) елементів матриці;

c) для виконання завдання обробки елементів матриці повинно бути написано дві різні функції:

1) з додатковими операторами виведення налагоджувальної інформації на друк (debug-версія);

2) з виконанням заданих дій без додаткового виведення налагоджувальної інформації (release-версія).

4. Для компіляції написаної багатофайлової програми написати окремий make-файл, причому:

a) при зміні одного із вихідних файлів повинен перекомпільовуватися лише цей файл (а також відбуватися дії, необхідні для генерації бінарного файлу);

b) при видаленні бінарного файлу та незмінних вихідних файлах повинна відбуватися лише лінковка;

c) забезпечити окрему ціль для очистки згенерованих файлів;

5. Вміти компілювати написану багатофайлову програму двома способами:

a) за допомогою однієї команди gcc;

b) за допомогою make-файлу.

6. Виконати тестування та налагодження програми на комп’ютері. При тестуванні програми необхідно підбирати такі вхідні набори початкових значень матриці, щоб можна було легко відстежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

**Варіант 21**

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти

в ній останній максимальний елемент і його місцезнаходження (координати).

**Текст програми мовою C**

**//typedef.h**

#ifndef TYPEDEF\_H

#define TYPEDEF\_H

#include <stdio.h>

#include <stdlib.h>

#define n 8

#define m 8

int \*\*array;

void randomArray(int \*\*array);

void sortedArray(int \*\*array);

void usersArray(int \*\*array);

void output(int \*\*array);

void changeMinMax(int \*\*array);

#endif

**//input.c**

#include "typedef.h"

void randomArray(int \*\*array)

{

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

{

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

{

array[i][j] = rand() % 50;

}

}

}

void sortedArray(int \*\*array)

{

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

{

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

{

array[i][j] = i \* 10 + j;

}

}

}

void usersArray(int \*\*array)

{

printf("Enter %d", m \* n);

printf(" elements in array\n");

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

{

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

{

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

}

}

}

**//output.c**

#include "typedef.h"

void output(int \*\*array)

{

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

{

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

{

printf("%3d", array[i][j]);

}

printf("\n");

}

}

**//program.c**

#include "typedef.h"

void changeMinMax(int \*\*array)

{

int tempMax = 0;

int tempMin = array[0][n - 1];

int imax = 0;

int jmax = 0;

int imin = 0;

int jmin = 0;

int j = 0;

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

{

if (tempMax <= array[i][n - 1 - j])

{

tempMax = array[i][n - 1 - j];

imax = i;

jmax = n - 1 - j;

}

if (tempMin >= array[i][n - 1 - j])

{

tempMin = array[i][n - 1 - j];

imin = i;

jmin = n - 1 - j;

}

j++;

}

tempMax = array[imin][jmin];

array[imin][jmin] = array[imax][jmax];

array[imax][jmax] = tempMax;

}

**//main.c**

#include "typedef.h"

int main()

{

array = (int \*\*)malloc(sizeof(int \*) \* m);

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

{

array[i] = (int \*)malloc(sizeof(int) \* n);

}

printf("1. Random Array\n2. Sorted Array\n3. Introduced Array\n0. Exit\nChoose a mode: ");

int temp;

while (1)

{

scanf("%d", &temp);

switch (temp)

{

case 1:

printf("Test: random array\n");

randomArray(array);

output(array);

changeMinMax(array);

output(array);

printf("1. Random Array\n2. Sorted Array\n3. Introduced Array\n0.Exit \nChoose a mode: ");

break;

case 2:

printf("Test: sorted array\n");

sortedArray(array);

output(array);

changeMinMax(array);

output(array);

printf("1. Random Array\n2. Sorted Array\n3. Introduced Array\n0. Exit \nChoose a mode: ");

break;

case 3:

printf("Test: introduced array\n");

usersArray(array);

output(array);

changeMinMax(array);

output(array);

printf("1. Random Array\n2. Sorted Array\n3. Introduced Array\n0. Exit \nChoose a mode: ");

break;

case 0:

return 0;

default:

printf("Input correct number\n");

printf("1. Random Array\n2. Sorted Array\n3. Introduced Array\n0. Exit \nChoose a mode: ");

}

}

}

**Командні рядки для компілювання та запуску програми мовою C**

gcc -o Lab2 output.c input.c program.c main.c

./Lab2

Для make-файлу:

make

./Lab2

make clean

**Make-файл для компілювання та запуску програми мовою C**

.PHONY: clean

Lab2: input.o output.o program.o main.o

gcc input.o output.o program.o main.o -o Lab2

input.o: input.c

gcc -c input.c -o input.o

output.o: output.c

gcc -c output.c -o output.o

program.o: program.c

gcc -c program.c -o program.o

main.o: main.c

gcc -c main.c -o main.o

clean:

rm -rf Lab2 \*.o

**Постановка завдання для програми мовою Java**

1. Написати консольну програму розв’язання задачі пошуку (за варіантом) у двовимірному масиві (матриці) одним з алгоритмів методу лінійного пошуку.

2. Розміри матриці m та n взяти самостійно у межах від 7 до 10.

3. При написанні програми повинно бути щонайменше три класи, один із

яких буде відповідати за пошук елементу в матриці, другий відповідати за

ввід-вивід матриці, а третій — головний клас, що міститиме метод main.

4. Для компіляції та запуску написаної програми написати окремий

make-файл, причому забезпечити окремі цілі для очистки згенерованих

файлів, а також генерації JAR-архіву.

5. Вміти компілювати написану програму двома способами:

a) за допомогою однієї команди javac;

b) за допомогою make-файлу.

6. Виконати тестування та налагодження програми на комп’ютері. При тестуванні програми необхідно підбирати такі вхідні набори початкових значень матриці, щоб можна було легко відстежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

**Варіант 6**

Задано матрицю дійсних чисел A[m,n]. При обході матриці по рядках знайти

в ній останній максимальний елемент і його місцезнаходження (координати).

**Текст програми мовою Java, вхідні дані**

**//Output.java**

package Out;

public class Output {

public static void OutputArray(int array[][], int m, int n) {

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

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

System.out.format("%3d", array[i][j]);

}

System.out.println();

}

System.out.println();

}

}

**//Input.java**

package Inp;

import static java.lang.Math.random;

import java.util.Scanner;

public class Input {

public static void RandomArr(int array[][], int m, int n) {

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

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

array[i][j] = (int) ((random() \* 1000) % 50);

}

}

}

public static void SortedArr(int array[][], int m, int n) {

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

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

array[i][j] = 10 \* i + j;

}

}

}

public static void UsersArr(int array[][], int m, int n) {

System.out.println("Enter" + m \* n + "elements in array\n");

Scanner scan = new Scanner(System.in);

int temp;

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

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

String nextIntString = scan.nextLine();

boolean numeric = true;

try {

Integer num = Integer.parseInt(nextIntString);

} catch (NumberFormatException e) {

numeric = false;

}

if (numeric) {

temp = Integer.parseInt(nextIntString);

array[i][j] = temp;

} else {

System.out.println("You entered wrong number\n");

if (j != 0) {

j--;

} else if (i != 0 && j == 0)

i--;

j = n - 2;

}

}

}

scan.close();

}

}

**//Program.java**

package Prog;

public class Program {

public static void ChangeMinMax(int array[][], int m, int n) {

int tempMax = 0;

int tempMin = array[0][n - 1];

int imax = 0;

int jmax = 0;

int imin = 0;

int jmin = 0;

int j = 0;

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

if (tempMax <= array[i][n - 1 - j]) {

tempMax = array[i][n - 1 - j];

imax = i;

jmax = n - 1 - j;

}

if (tempMin >= array[i][n - 1 - j]) {

tempMin = array[i][n - 1 - j];

imin = i;

jmin = n - 1 - j;

}

j++;

}

tempMax = array[imin][jmin];

array[imin][jmin] = array[imax][jmax];

array[imax][jmax] = tempMax;

}

}

**//Lab2.java**

import Out.Output;

import Inp.Input;

import Prog.Program;

import java.util.Scanner;

public class Lab2 {

public static void main(String[] args) {

final int n = 5;

final int m = 5;

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

int temp;

System.out.printf("1. Random Array\n2. Sorted Array\n3. Users Array\nChoose a mode: ");

Scanner scan = new Scanner(System.in);

while (true) {

String nextIntString = scan.nextLine();

boolean numeric = true;

try {

Integer num = Integer.parseInt(nextIntString);

} catch (NumberFormatException e) {

numeric = false;

}

if (numeric) {

temp = Integer.parseInt(nextIntString);

} else {

temp = 5;

}

switch (temp) {

case 1:

System.out.println("Test: random array");

Input.RandomArr(array, m, n);

Output.OutputArray(array, m, n);

Program.ChangeMinMax(array, m, n);

Output.OutputArray(array, m, n);

System.out.printf("1. Random Array\n2. Sorted Array\n3. Users Array\nChoose a mode: ");

break;

case 2:

System.out.println("Test: sorted array");

Input.SortedArr(array, m, n);

Output.OutputArray(array, m, n);

Program.ChangeMinMax(array, m, n);

Output.OutputArray(array, m, n);

System.out.printf("1. Random Array\n2. Sorted Array\n3. Users Array\nChoose a mode: ");

break;

case 3:

System.out.println("Test: users array");

Input.UsersArr(array, m, n);

Output.OutputArray(array, m, n);

Program.ChangeMinMax(array, m, n);

Output.OutputArray(array, m, n);

System.out.printf("1. Random Array\n2. Sorted Array\n3. Users Array\nChoose a mode: ");

break;

case 0:

return;

default:

System.out.printf("Input correct number! \n1. Random Array\n2. Sorted Array\n3. Users Array\nChoose a mode: ");

break;

}

}

}

}

**Командні рядки для компілювання та запуску програми мовою Java**

javac -d . Input.java

javac -d . Output.java

javac -d . Program.java

javac Lab2.java

java Lab2

jar -cfm Lab2.jar manifest.txt Lab2.class Prog/Program.class Out/Output.class Inp/Input.class

java -jar Lab2.jar

Для make-файлу:

make

make Jar

java Lab2

make clean

**Make-файл для компілювання та запуску програми мовою Java**

.PHONY: clean, mkjar

Lab2.class: Lab2.java Inp/Input.class Out/Output.class Prog/Program.class

javac Lab2.java

Inp/Input.class: Input.java

javac -d . Input.java

Out/Output.class: Output.java

javac -d . Output.java

Prog/Program.class: Program.java

javac -d . Program.java

mkjar:

jar -cfm Lab2.jar manifest.txt Lab2.class Prog/Program.class Out/Output.class Inp/Input.class

clean:

rm -rf \*.class

rm -r Inp Out Prog Lab2.jar