Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 8 з дисципліни

«Алгоритми та структури даних-1.

Основи алгоритмізації»

«Організація циклічних процесів. Ітераційні цикли»

Варіант 32

Виконав студент ІП-11 Фукс Вікторія Ігорівна

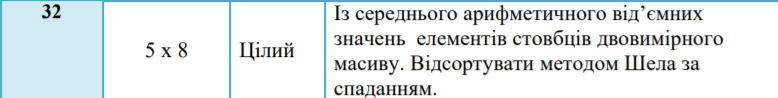
Мартинова Оксана Петрівна

Перевірив \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ 2021**Лабораторна робота 8**

**Дослідження алгоритмів пошуку та сортування**

**Мета** – дослідити алгоритми пошуку та сортування, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

**Варіант 32:** **Постановка задачі**: Описуємо двовимірний масив 5х8 випадковими цілими числами. Створення одновимірний масив та ініціюємо середнім арифм. від’ємних значень двовимірного масиву. Сортуємо методом Шела за спаданням.

**Математична модель**:

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| рядки | цілочисельний | ROW | проміж.результ |
| стовбці | цілочисельний | COLS | проміж.результ |
| матриця | цілочисельний | matrix[ROW][COLS] | проміж.результ |
| масив | дійсний | arr[COLS] | результат |
| сумма від’ємних | цілочисельний | calc | проміж.результ |
| лічильник | цілочисельний | k | проміж.результ |
| лічильник | дійсний | colvo | проміж.результ |
| масив | цілочисельний | p\_mtx[ROW] | результат |
| лічильник | цілочисельний | i | проміж.результ |

**Псевдокод:**

**Крок 1:** Визначимо основні дії

**Крок 2:** Деталізуємо умову з використанням підпрограми та умовних операторів.

**Крок 1:**

**Початок**

row = 5

col = 8

Заповнення матриці

Заповнення масиву

Сортування масиву

**Кінець**

**Крок 2:**

**Підпрограма** input\_matrix(const int ROW, const int COLS, int\*\* p\_mtx)**:**

**виконати**(int i = 0; i < ROW; i++)

**виконати**(int j = 0; j < COLS; j++)

p\_mtx[i][j] = rand() % 101 – 50

**все виконати**

**все виконати**

**Кінець підпрограми**

**Підпрограма** print\_arr(const int COLS, float arr[])**:**

**виконати**(int j = 0; j < COLS; j++)

**вивести** arr[j]

**все виконати**

**Кінець підпрограми**

**Підпрограма** print\_matrix(const int ROW, const int COLS, int\*\* p\_mtx)**:**

**виконати**(int i = 0; i < ROW; i++)

**виконати**(int j = 0; j < COLS; j++)

**вивести** p\_mtx[i][j]

**все виконати**

**все виконати**

**Кінець підпрограми**

**Підпрограма** proccess\_matrix(const int ROW, const int COLS, int calc, int colvo, int\*\* p\_mtx, float arr[]):

**виконати**(int j = 0; j < COLS; j++)

**виконати**(int k = 0; k < ROW; k++)

**якщо** (p\_mtx[0 + k][j] < 0)

calc += p\_mtx[0 + k][j]

colvo++

**все якщо**

**все виконати**

arr[j] = calc / colvo

calc = 0

colvo = 0

**все виконати**

**Кінець підпрограми**

**Підпрограма** Shell\_Sort(int n, float arr2[]):

**виконати**(step = n / 2; step > 0; step /= 2)

**виконати**(i = step; i < n; i++)

tmp = arr2[i]

**виконати** (j = i; j >= step; j -= step)

**якщо** (tmp > arr2[j - step])

arr2[j] = arr2[j - step]

**інакше**

break

**все якщо**

**все виконати**

arr2[j] = tmp

**все виконати**

**все виконати**

**Кінець підпрограми**

**Початок**

row = 5

col = 8

input\_matrix(ROW, COLS, p\_mtx)

proccess\_matrix(ROW, COLS, calc, colvo, p\_mtx, arr)

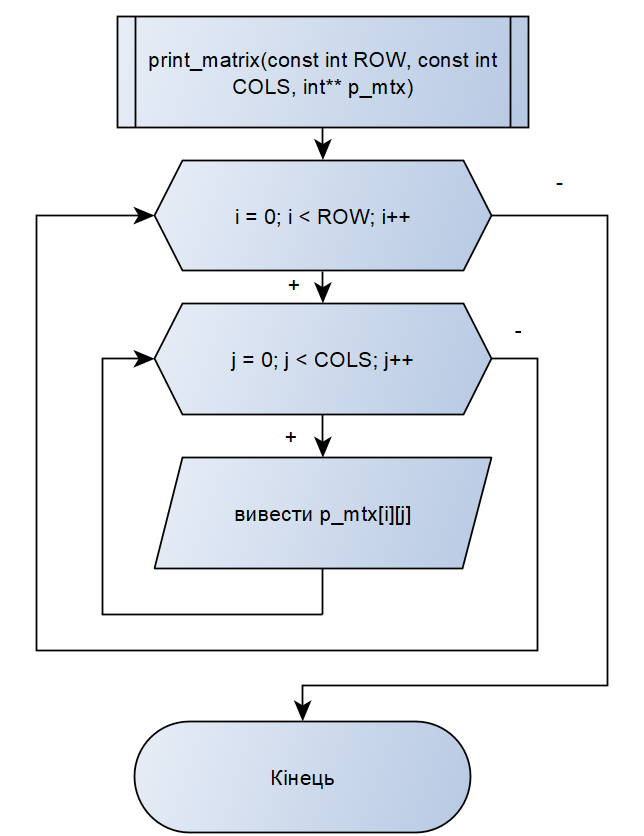
print\_matrix(COLS, ROW, p\_mtx);

print\_arr(COLS, arr);

Shell\_Sort(COLS, arr);

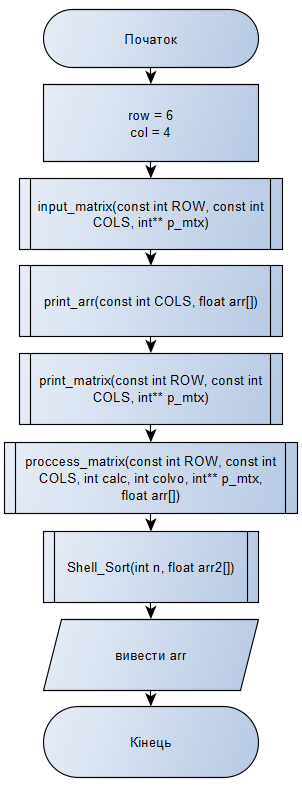
print\_arr(COLS, arr);

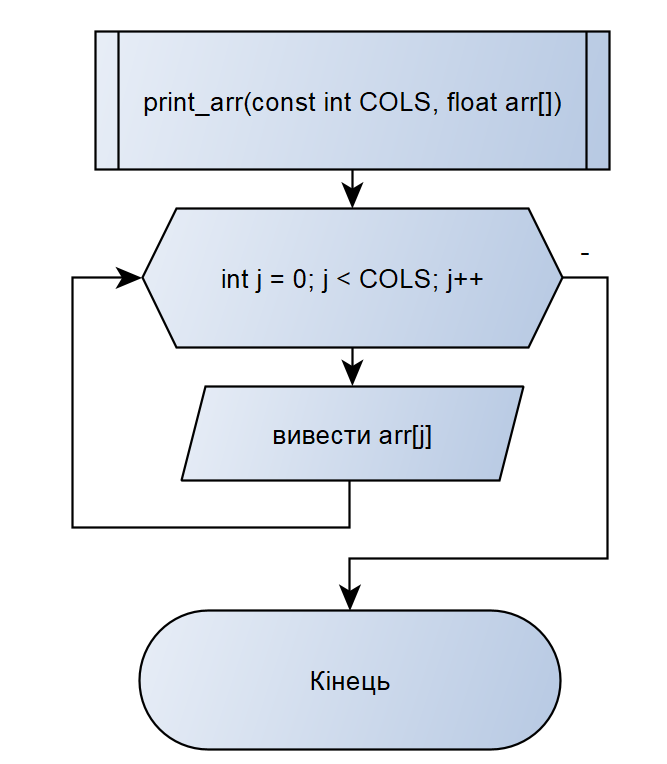
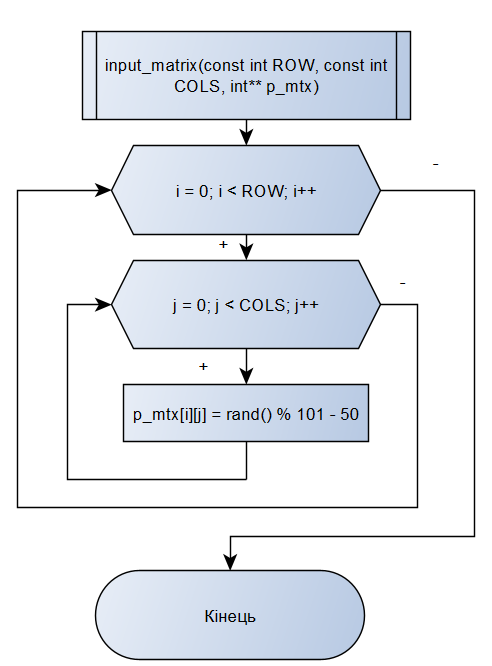
**Кінець**

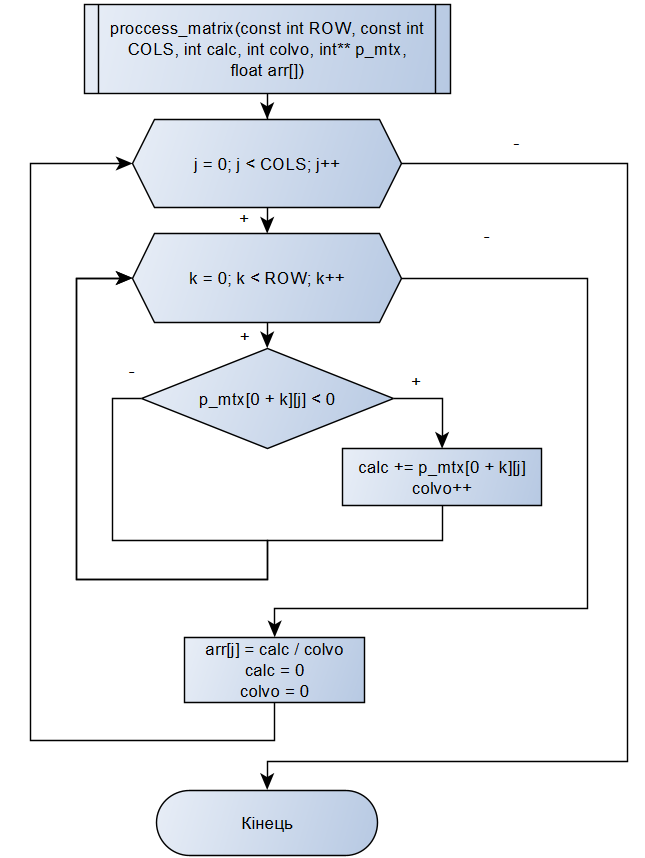
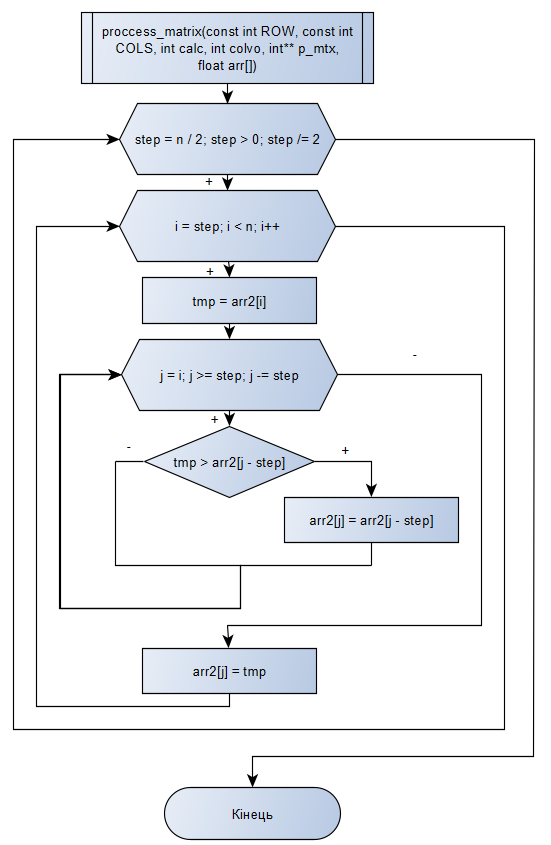
**Блок-схеми:**

**Крок 2:**

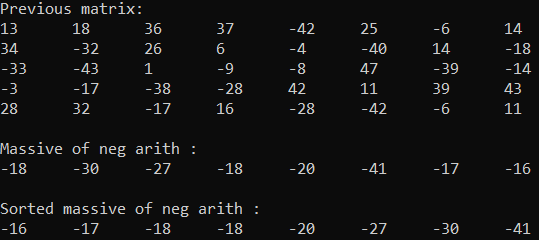
**Крок 1:**



****

****

**Випробування алгоритму:**

****

**Код:**

**С++**

#include <iostream>

#include <ctime>

using namespace std;

void input\_matrix(const int ROW, const int COLS, int\*\* p\_mtx);

void Shell\_Sort(int n, float arr2[]);

void print\_arr(const int COLS, float arr2[]);

void print\_matrix(const int COLS, const int ROW, int\*\* p\_mtx);

void proccess\_matrix(const int ROW, const int COLS, int calc, int colvo, int\*\* p\_mtx, float arr2[]);

int main()

{

const int ROW = 5;

const int COLS = 8;

int matrix[ROW][COLS];

float arr[COLS];

int calc = 0, k = 0;

float colvo = 0;

int\* p\_mtx[ROW];

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

p\_mtx[i] = matrix[i];

}

srand(time(NULL));

input\_matrix(ROW, COLS, p\_mtx);

proccess\_matrix(ROW, COLS, calc, colvo, p\_mtx, arr);

cout << "Previous matrix: " << endl;

print\_matrix(COLS, ROW, p\_mtx);

cout << endl;

cout << "Massive of neg arith : " << endl;

print\_arr(COLS, arr);

cout << endl << endl;

cout << "Sorted massive of neg arith : " << endl;

Shell\_Sort(COLS, arr);

print\_arr(COLS, arr);

return 0;

}

void Shell\_Sort(int n, float arr2[])

{

int i, j, step;

float tmp;

for (step = n / 2; step > 0; step /= 2)

{

for (i = step; i < n; i++)

{

tmp = arr2[i];

for (j = i; j >= step; j -= step)

{

if (tmp > arr2[j - step])

arr2[j] = arr2[j - step];

else

break;

}

arr2[j] = tmp;

}

}

}

void proccess\_matrix(const int ROW, const int COLS, int calc, int colvo, int\*\* p\_mtx, float arr[])

{

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

{

for (int k = 0; k < ROW; k++)

{

if (p\_mtx[0 + k][j] < 0)

{

calc += p\_mtx[0 + k][j];

colvo++;

}

}

arr[j] = calc / colvo;

calc = 0;

colvo = 0;

}

}

void input\_matrix(const int ROW, const int COLS, int\*\* p\_mtx)

{

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

{

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

{

p\_mtx[i][j] = rand() % 101 - 50;

}

}

}

void print\_arr(const int COLS, float arr[])

{

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

{

cout << arr[j] << "\t";

}

}

void print\_matrix(const int COLS, const int ROW, int\*\* p\_mtx)

{

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

{

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

{

cout << p\_mtx[i][j] << "\t";

}

cout << endl;

}

}

**Висновок:**

Ми опанували та дослідили алгоритми пошуку та сортування, набули практичних навичок використання цих алгоритмів під час складання програмних специфікацій. У результаті лабораторної роботи розробили код, математичну модель, що відповідає постановці задачі, псевдокод, розробили покрокові блок-схеми, які пояснюють логіку алгоритму. Написали код на С++. Зробили випробування алгоритму.