# 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 і структури даних

»

Варіант 23

Виконав студент Панченко Сергій Віталійович

(шифр, прізвище, ім'я, по батькові)

Перевірив

( прізвище, ім'я, по батькові)

Київ 202

## Лабораторна робота 7

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

**Постановка:**

C:\Users\User\AppData\Local\Microsoft\Windows\INetCache\Content.Word\goal.png

**Математична модель**:

**Складемо таблицю імен змінних**

|  |  |  |  |
| --- | --- | --- | --- |
| Зміна | Тип | Ім’я | Призначення |
| rows | Const int | Рядки | Проміжне значення |
| cols | Const int | Стовпчики | Проміжне значення |
| arr | Double[][] | Двовимірний масив | Проміжне значення |
| temp | Double | Тимчасова змінна | Проміжне значення |
| Flag | Bool | Флаг | Проміжне значення |
| P | Double \* | Динамічний масив добутків | Результат |
| Size | Int | Розмір динамічного масиву добутків | Проміжне значення |
| New\_p | Double \* | Новий динамічний масив добутків | Проміжне значення |
| i | Int | Розмір динамічного масиву добутків | Проміжне значення |
| j | int | Змінна циклу | Проміжне значення |

Заповнивши масив arr, заповнимо масив arr добутками від’ємних значень, пересортуємо його вставкою у порядку зростання.

**Псевдокод:**

Крок 1: Визначимо основні дії

Крок 2: деталізуємо заповнення arr

Крок 3: деталізуємо заповнення p

Крок 4: деталізуємо сортування p

Крок 1:

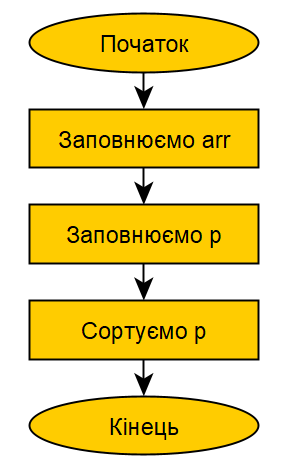
Початок

Заповнюємо arr

Заповнюємо p

Сортуємо p

Кінець



Крок 2:

Початок

rows = 8;

cols = 4;

arr[rows][cols];

Поки i=0,rows:

Поки j=0, cols

arr[i][j] = (double)(-10000 + rand() %20000)/1000;

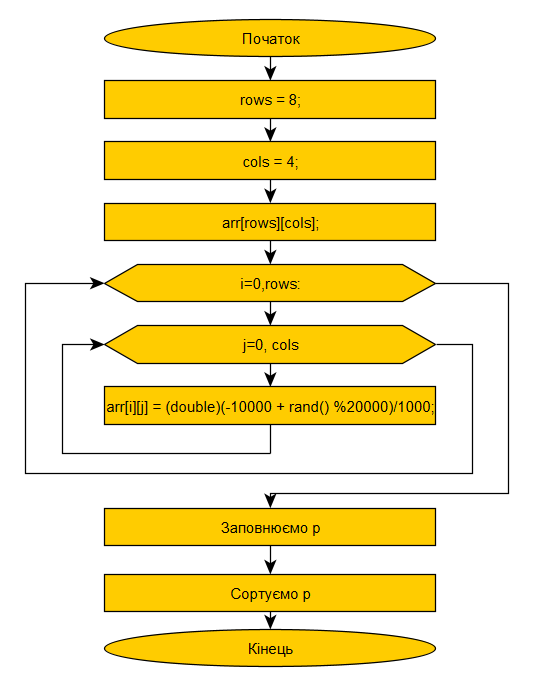
Поки все

Поки все

Заповнюємо p

Сортуємо p

Кінець



Крок 3:

Початок

rows = 8;

cols = 4;

arr[rows][cols];

Поки i=0,rows:

Поки j=0, cols

arr[i][j] = (double)(-10000 + rand() %20000)/1000;

Поки все

size = 0;

Поки i=0,rows:

Поки j=0, cols

Якщо arr[i][j]<0:

flag = true;

temp \*= arr[i][j];

Все якщо

Якщо flag == true:

Якщо size == 0:

Size++

p = new double[size];

\*p = temp;

Якщо все

Інакше

size++;

double\* new\_p = new double[size];

Поки k=0,size-1:

new\_p[k] = p[k];

Поки все

new\_p[size - 1] = temp;

delete[] p;

p = new\_p;

Все інакше

Поки все

flag = false;

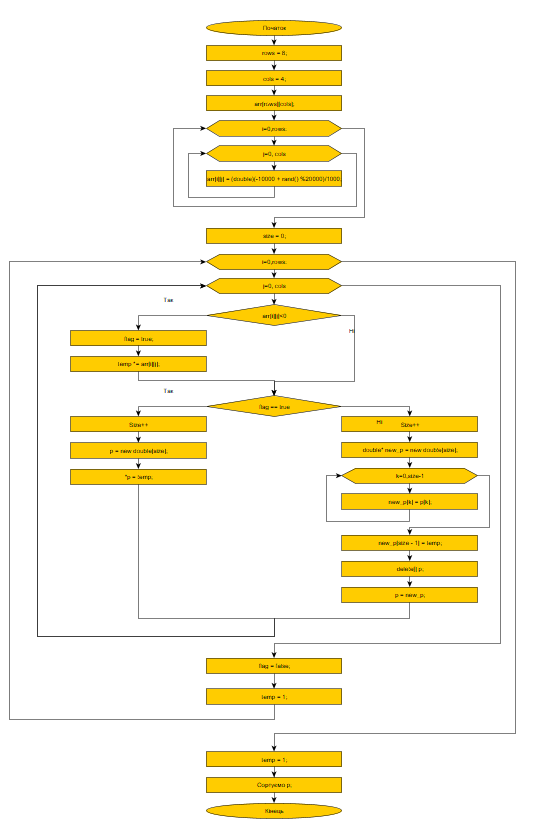
temp = 1;

Поки все

temp = 1;

Сортуємо p;

Кінець



Крок 4:

Початок

rows = 8;

cols = 4;

arr[rows][cols];

Поки i=0,rows:

Поки j=0, cols

arr[i][j] = (double)(-10000 + rand() %20000)/1000;

Поки все

size = 0;

Поки i=0,rows:

Поки j=0, cols

Якщо arr[i][j]<0:

flag = true;

temp \*= arr[i][j];

Все якщо

Якщо flag:

Якщо size == 0:

Size++

p = new double[size];

\*p = temp;

Якщо все

Інакше

size++;

double\* new\_p = new double[size];

Поки k=0,size-1:

new\_p[k] = p[k];

Поки все

new\_p[size - 1] = temp;

delete[] p;

p = new\_p;

Все інакше

Поки все

flag = false;

temp = 1;

Поки все

temp = 1;

Поки i=0,rows:

Поки j=0,cols;

Виводимо arr[i][j];

Поки все

Поки все

Якщо size >0:

Поки i=0,size:

Виводимо p[i];

Поки все

Поки i=0, size:

temp = p[i];

j=i-1

Поки j >= 0 && p[j] > temp:

p[j + 1] = p[j];

j--;

Поки все

p[j + 1] = temp;

Поки все

Поки i=0,size:

Виводимо p[i];

Поки все

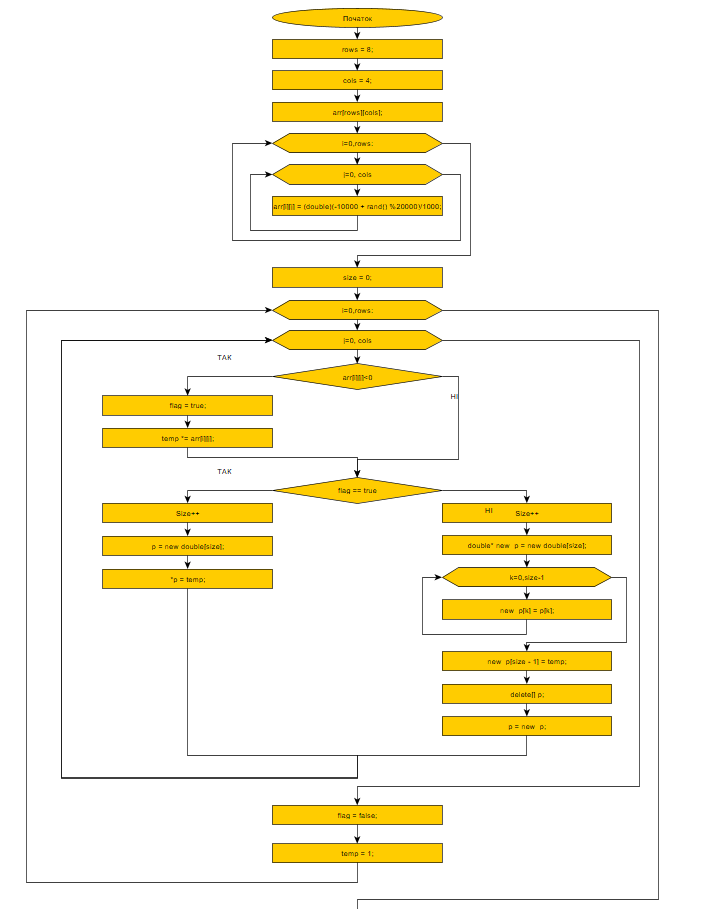
Все якщо

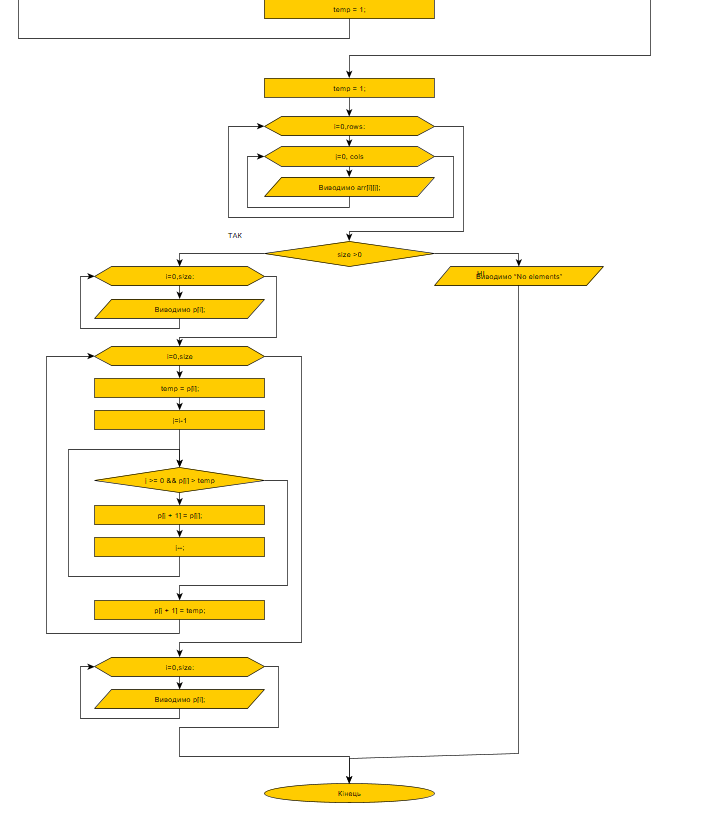
Інакше

Виводимо “No elements”

Все інакше

Кінець





Код:

#include <iostream>

using namespace std;

int main()

{

srand(time(NULL));

const int rows = 8;

const int cols = 4;

double arr[rows][cols];

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

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

arr[i][j] = (double)(-10000 + rand() %20000)/1000;

}

}

double temp = 1;

bool flag = false;

double \*p = nullptr;

int size = 0;

int i;

int j;

for (i = 0;i < rows;i++) {

for (j = 0;j < cols;j++) {

if (arr[i][j] < 0) {

flag = true;

temp \*= arr[i][j];

}

}

if (flag) {

if (size == 0) {

size++;

p = new double[size];

\*p = temp;

}

else {

size++;

double\* new\_p = new double[size];

for (int k = 0;k < size - 1;k++) {

new\_p[k] = p[k];

}

new\_p[size - 1] = temp;

delete[] p;

p = new\_p;

}

}

flag = false;

temp = 1;

}

temp = 1;

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

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

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

}

cout << endl;

}

if (size > 0) {

cout << endl;

for (i = 0;i < size;i++) {

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

}

cout << endl;

for (i = 0;i < size;i++) {

temp = p[i];

for (j = i - 1;j >= 0 && p[j] > temp;j--) {

p[j + 1] = p[j];

}

p[j + 1] = temp;

}

cout << endl;

for (i = 0;i < size;i++) {

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

}

cout << endl;

}

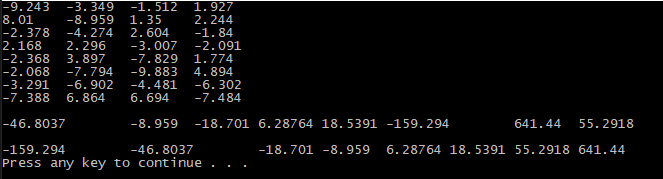
else {

cout << "No elements <0" << endl;

}

system("pause");

}



Перевірка:

Створимо таблицю для перевірки:

|  |  |  |
| --- | --- | --- |
| p |  | Відсортований p |
| 0 | -46.8037 | -159.294 |
| 1 | -8.959 | -46.8037 |
| 2 | -18.701 | -18.701 |
| 3 | 6.28764 | -8.959 |
| 4 | 18.5391 | 6.28764 |
| 5 | -159.294 | 18.5391 |
| 6 | 641.44 | 55.2918 |
| 7 | 55.2918 | 641.44 |

Отже, бачимо, що алгоритм працює. Алгоритм сортує масив.

**Висновок:**

Під час лабораторної роботи дослідили алгоритми пошуку та сортування, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Математична модель, блок-схеми, псевдокод, код випробовування наведені. Оскільки масив правильно відсортований, то алгоритм правильно подає результат.