# 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»

«Дослідження лінійних алгоритмів»

Варіант 25

Виконав студент ІП-11 Прищепа Владислав Станіславович

(шифр, прізвище, ім'я, по батькові)

Перевірив Мартинова Оксана Петрівна

( прізвище, ім'я, по батькові)

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 25

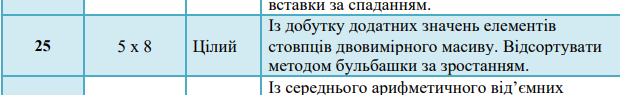
**Індивідуальне завдання:**

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (табл. 1).

2. Ініціювання змінної, що описана в п.1 даного завдання.

3. Створення нової змінної індексованого типу (одновимірний масив) та її ініціювання значеннями, що обчислюються згідно з варіантом (табл. 1).



**Постановка задачі:**

Вивести матрицю 5х8, яка заповнена випадковими цілими числами. Вивести масив добутків додатних елементів матриці. Впорядкувати масив методом бульбашки за зростанням і вивести ще раз цей масив.

**Математична модель:**

|  |  |  |  |
| --- | --- | --- | --- |
| Зміна | Тип | Ім’я | Призначення |
| К-сть рядків | int | n | Вхідне значення |
| К-сть стовпців | int | k | Вхідне значення |
| Матриця | int\*\* | A | Проміжне значення |
| Масив добутків | int\* | B | Проміжне значення, результат |
| Значення к-сті рядків | int | q | Змінна підпрограми |
| Значення к-сті стовпців | int | z | Змінна підпрограми |
| Покажчик на матрицю | int\*\* | M | Змінна підпрограми |
| Покажчик на масив | int\* | Q | Змінна підпрограми |
| Ітерація 1 | int | i | Змінна підпрограми |
| Ітерація 2 | int | j | Змінна підпрограми |

Використаймо у основній програмі підпрограми для генерування динамічної матриці 5\*8, її заповнення цілими числами, виведення матриці, порозрядного пошуку додатних елементів стовпців матриці із подальшим їх перемноженням і заповненням динамічного масиву, сортування елементів масиву методом бульбашки, виведення масиву та видалення динамічної матриці. Всі підпрограми будуть використовувати арифметичні цикли або складні, що являють собою арифметичні цикли із вкладеними арифметичними циклами.

**Розв’язання**

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

Крок 1: Визначимо основні дії

Крок 2: деталізуємо ініціалізацію динамічної матриці А

Крок 3: деталізуємо заповнення матриці А

Крок 4: деталізуємо виведення матриці А

Крок 5: деталізуємо заповнення динамічного масиву В

Крок 6: деталізуємо сортування масиву В

Крок 7: деталізуємо виведення масиву В

Крок 8: деталізуємо видалення динамічної матриці А

**Псевдокод**

**Основна програма**

**Початок**

n:=5

k:=8

A := array\_2d(n, k)

generate\_2d\_array(A, n, k)

**Виведення** "matrix:"

output\_2d\_array(n, k, A)

int\* B := new int[k]

fill\_arr(A, B, n, k)

**Виведення** "array before arregement:"

output\_array(B, k)

arragement(B, k)

**Виведення** "array after arregement:"

output\_array(B, k)

delition\_of\_2d\_array(A, n)

delete[] B

**Кінець**

**Підпрограми**

**array\_2d(q, z)**

int\*\* M := new int\* [q]

**для** і **від** 0 **до** q, **з кроком** 1 **повторити**

M[i] := new int[z]

**все повторити**

return M

**Кінець**

**generate\_2d\_array(M, q, z)**

srand(time(NULL))

**для** і **від** 0 **до** q, **з кроком** 1 **повторити**

**для** j **від** 0 **до** z, **з кроком** 1 **повторити**

M[i][j] := rand() % 101 – 50

**все повторити**

**все повторити**

**Кінець**

**output\_2d\_array(q, z, M)**

**для** і **від** 0 **до** q, **з кроком** 1 **повторити**

**для** j **від** 0 **до** z, **з кроком** 1 **повторити**

**Виведення** M[i][j], “\t”

**все повторити**

**все повторити**

**Виведення** endl

**Кінець**

**fill\_arr(M, Q, q, z)**

**для** j **від** 0 **до** z, **з кроком** 1 **повторити**

Q[j] := 1

**для** i **від** 0 **до** q, **з кроком** 1 **повторити**

**Якщо** M[i][j] > 0

Q[j] := M[i][j] \* Q[j]

**все якщо**

**все повторити**

**все повторити**

**Кінець**

**arragement(Q, z)**

**для** j **від** 1 **до** z, **з кроком** 1 **повторити**

**для** i **від** 0 **до** z - j, **з кроком** 1 **повторити**

**Якщо** Q[i] > Q[i + 1]

s := Q[i]

Q[i] := Q[i + 1]

Q[i + 1] := s

**все якщо**

**все повторити**

**все повторити**

**Кінець**

**output\_array(Q, z)**

**для** j **від** 0 **до** z, **з кроком** 1 **повторити**

**Виведення** Q[j], “\t”

**все повторити**

**Виведення** endl

**Кінець**

**delition\_of\_2d\_array(M, q)**

**для** і **від** 0 **до** q, **з кроком** 1 **повторити**

delete[] M[i]

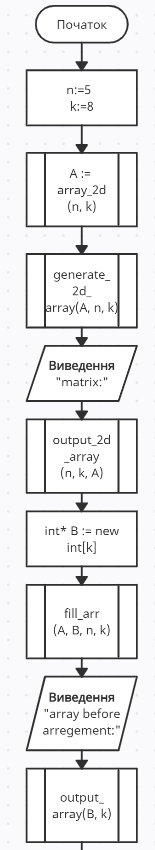
**все повторити**

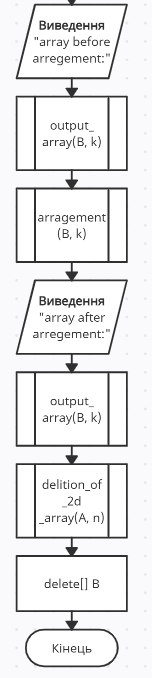
delete[] M

**Кінець**

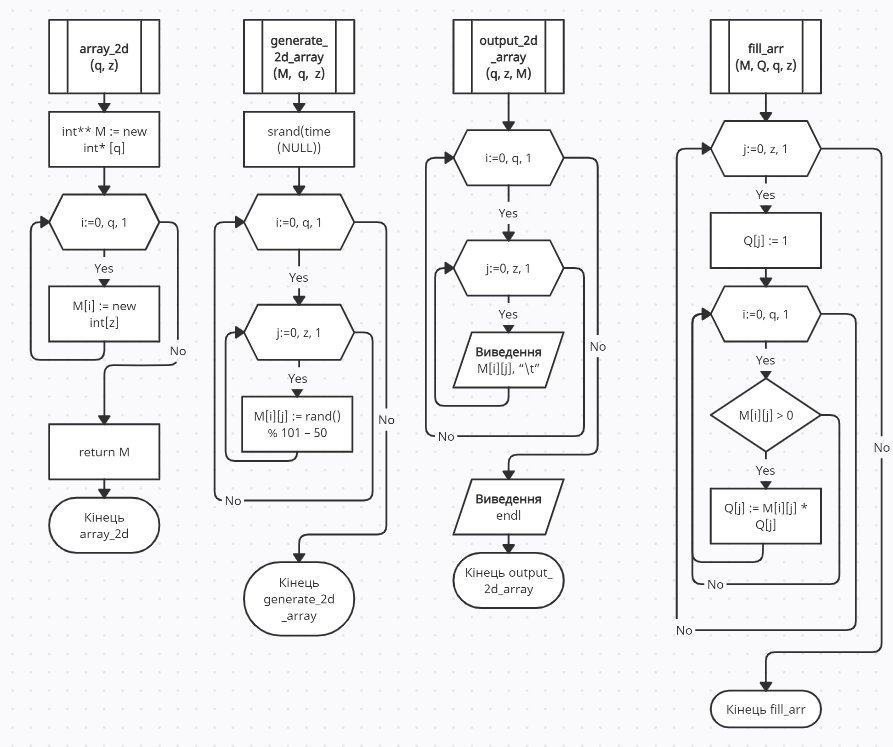
**Блок-схема алгоритму**

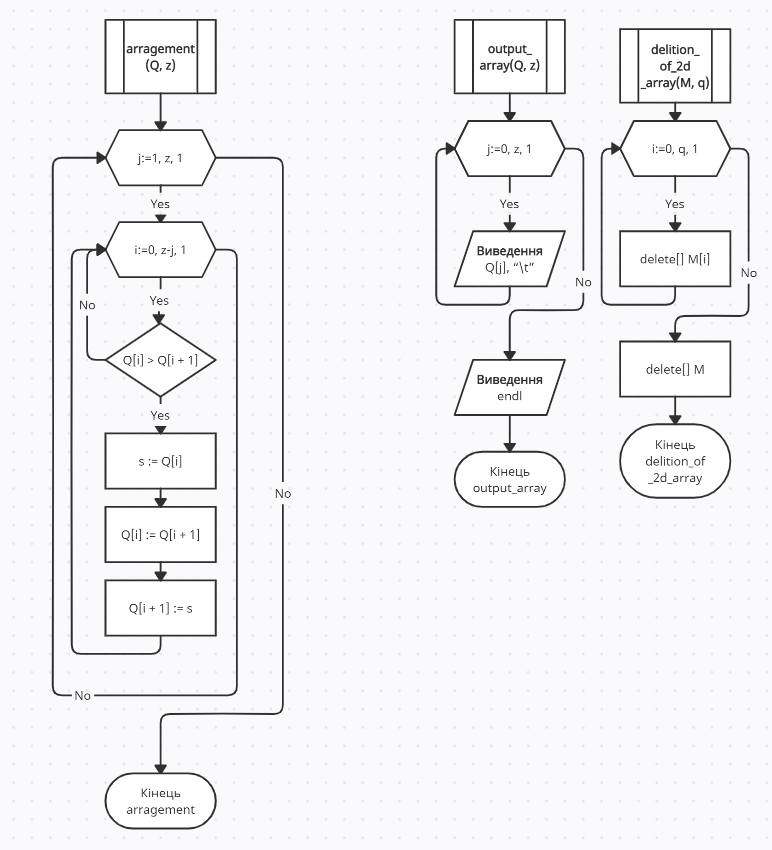
**Основна програма**

****

****

**Підпрограми**

****

****

**Код та виконання коду на С++:**

#include <iostream>

#include<cmath>

#include <time.h>

using namespace std;

int\*\* array\_2d(int, int);

void generate\_2d\_array(int\*\*, int, int);

void output\_2d\_array(int, int, int\*\*);

void fill\_arr(int\*\*, int\*, int, int);

void arragement(int\*, int);

void output\_array(int\*, int);

void delition\_of\_2d\_array(int\*\*, int);

void main() {

int n = 5;

int k = 8;

int\*\* A;

A = array\_2d(n, k);

generate\_2d\_array(A, n, k);

cout << "matrix:" << endl;

output\_2d\_array(n, k, A);

int\* B = new int[k];

fill\_arr(A, B, n, k);

cout << "array before arregement:" << endl;

output\_array(B, k);

arragement(B, k);

cout << "array after arregement:" << endl;

output\_array(B, k);

delition\_of\_2d\_array(A, n);

delete[] B;

system("pause");

}

int\*\* array\_2d(int q, int z) {

int\*\* M = new int\* [q];

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

M[i] = new int[z];

}

return M;

}

void generate\_2d\_array(int\*\* M, int q, int z) {

srand(time(NULL));

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

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

M[i][j] = rand() % 101 - 50;

}

}

}

void output\_2d\_array(int q, int z, int\*\* M) {

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

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

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

}

cout << endl;

}

}

void fill\_arr(int\*\* M, int\* Q, int q, int z) {

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

Q[j] = 1;

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

if (M[i][j] > 0) {

Q[j] \*= M[i][j];

}

}

}

}

void arragement(int\* Q, int z) {

for (int j = 1; j < z; j++) {

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

if (Q[i] > Q[i + 1]) {

int s = Q[i];

Q[i] = Q[i + 1];

Q[i + 1] = s;

}

}

}

}

void output\_array(int\* Q, int z) {

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

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

}

cout << endl;

}

void delition\_of\_2d\_array(int\*\* M, int q) {

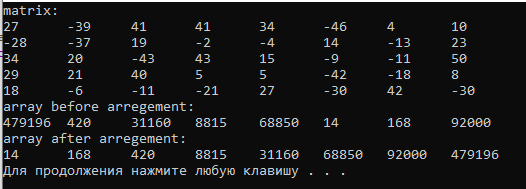
for (int i = 0; i < q; i++) {

delete[] M[i];

}

delete[] M;

}

****

**Випробування алгоритму**

|  |  |
| --- | --- |
| **Крок** | **Дія** |
|  | **Початок** |
| **1** | Згенерована матриця:  27 -39 41 41 34 -46 4 10  -28 -37 19 -2 -4 14 -13 23  34 20 -43 43 15 -9 -11 50  29 21 40 5 5 -42 -18 8  18 -6 -11 -21 27 -30 42 -30 |
| **2** | Виведення матриці |
| **3** | Добуток додатних елементів стовпців:  1 2 3 4 5 6 7 8  479196 420 31160 8815 68850 14 168 92000 |
| **4** | Виведення послідовності |
| **5** | Відсортована послідовність:  14  168  420 8815 31160 68850 92000 479196 |
| **6** | Виведення відсортованої послідовності |
|  | **Кінець** |

**Висновок**

Отже, у результаті лабораторної роботи ми дослідили алгоритми пошуку та сортування та набути практичних навичок їх використання під час складання програмних специфікацій підпрограм; навчилися будувати математичну модель, що відповідає постановці задачі, псевдокод та блок-схеми, які пояснюють логіку алгоритму. В результаті виконання лабораторної роботи ми отримали алгоритм для знаходження впорядкованої послідовності добутків додатних елементів стовпців матриці, декомпозували задачу на 8 кроків: визначили основні дії, деталізували ініціалізацію динамічної матриці А, її заповнення та виведення, заповнення динамічного масиву В, його сортування та виведення, видалення динамічної матриці А. У алгоритмі використовується підпрограма порозрядного пошуку додатних елементів кожного стовпця матриці. Оскільки формула перевірені вручну результати розрахунку членів послідовності відповідають дійсності, то алгоритм правильно подає результат.