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 алгоритмів пошуку та сортування»

Варіант 30

Виконав студент ІП-13 Симотюк Денис Андрійович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота 8

Дослідження алгоритмів пошуку і сортування

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

Варіант 30

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

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

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.

30	7 x 6	Дійсний	Із добутку від'ємних значень елементів рядків двовимірного масиву. Відсортувати методом Шела за спаданням.

Постановка задачі

Для вирішення задачі необхідно створити 2 масиви. Перший – даний двовимірний, а другий – шуканий одновимірний. Після введення елементів першого масиву обчислюється кількість рядків, де є хоча б одне від'ємне число. Результат буде кількістю елементів другого масиву. Після визначення елементів другого масиву він сортується за алгоритмом Шела.

Побудова математичної моделі

Сортування методом Шела - це алгоритм сортування, що є узагальненням сортування включенням. Але коли сортування включення працює з цілим масивом, алгоритм Шела обробляє його по частинам. Наприклад: спочатку поділяє масив на n/2 груп по 2 елементи, після чого кожна група обробляється сортуванням включення. Після цього створює n/4 груп по 4 елементи і так далі, поки масив не буде відсортовано.

Побудуємо таблицю змінних:

Змінна	Тип	Призначення
Масив **arr1(7, 6)**	Дійсний	Вхідні дані
Лічильник циклу **i**	Цілий	Проміжні дані, лічильник
Лічильник циклу **j**	Цілий	Проміжні дані, лічильник
Лічильник циклу **d**	Цілий	Проміжні дані, лічильник
Кількість рядків з від’ємними числами **n**	Цілий	Проміжні дані
Змінна для обчислення елементів масиву arr2 **res**	Дійсний	Проміжні дані
Змінна для сортування масиву arr2 **temp**	Дійсний	Проміжні дані
Змінна для заповнення масиву arr2 **k**	Цілий	Проміжні дані
Обчислення рядків з від’ємними елементами **Count_Elements**	Функція	Проміжні дані
Обчислення елементів arr2 **Innit_Arr2**	Процедура	Проміжні дані
Сортування arr2 **Sort_Res**	Процедура	Проміжні дані
Масив **arr2**	Дійсний	Вихідні дані

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо знаходження n.

Крок 3. Деталізуємо обчислення елементів масиву arr2.

Крок 4. Деталізуємо сортування масиву arr2.

Крок 5. Деталізуємо функцію Count_Elements.

Крок 6. Деталізуємо процедуру Innit_Arr2.

Крок 7. Деталізуємо процедуру Sort_Res.

Псевдокод алгоритму

Початок

Введення arr1

n = Count_Elements(arr1)

Innit_Arr2(arr1, arr2)

Output_Res(arr2, n)

Виведення arr2

Кінець

Функція Count_Elements(arr1)

n = 0;

повторити для i **від** 0 **до** 7 **з кроком 1**

повторити для j від 0 до 6 з кроком 1

якщо arr1[i, j] < 0

то

n = n + 1

break

все якщо

все повторити

все повторити

виведення n

все функція

Процедура Innit_Arr2(arr1, arr2)

k = 0

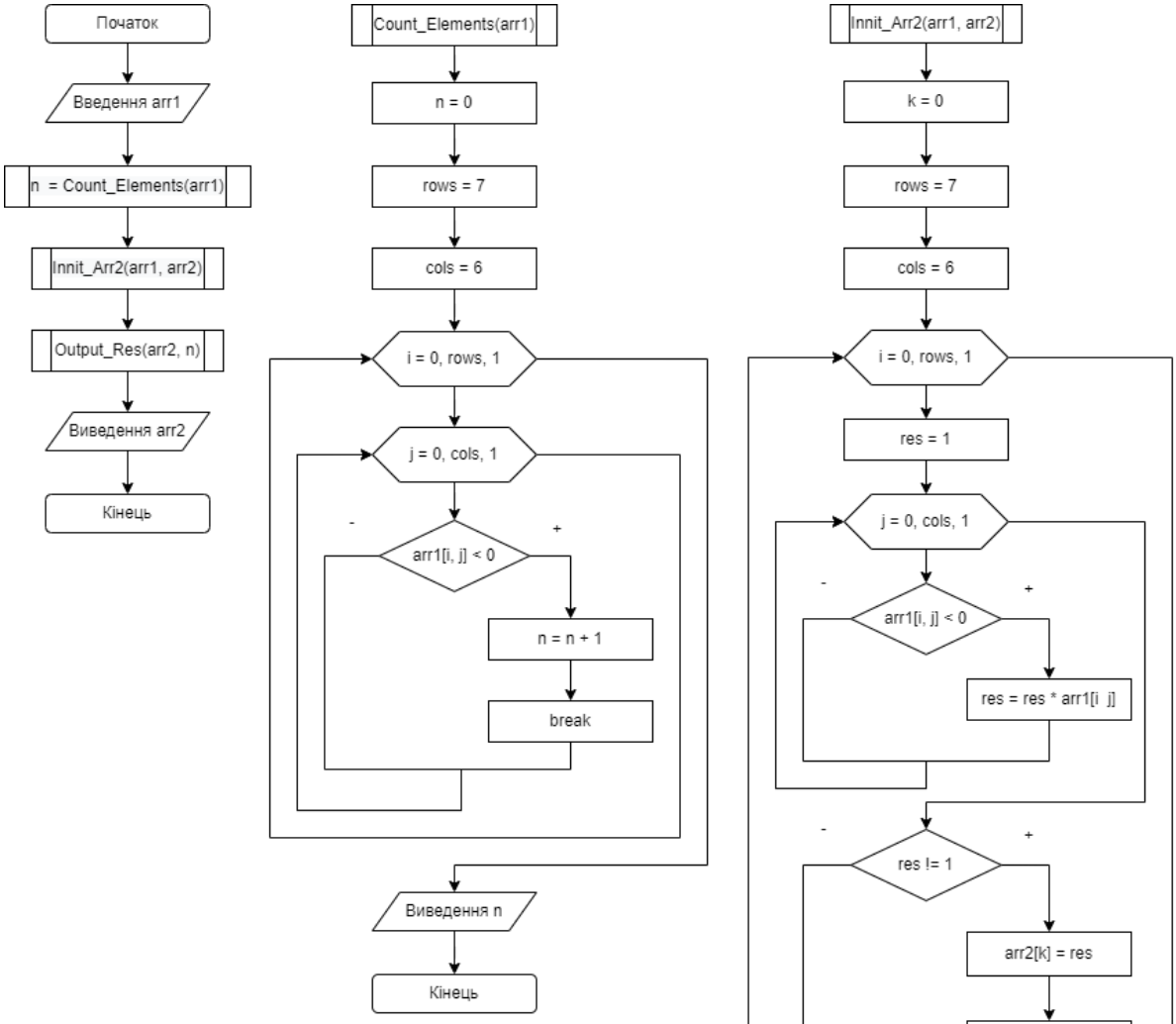
Повторити для і від 0 до 7 з кроком 1

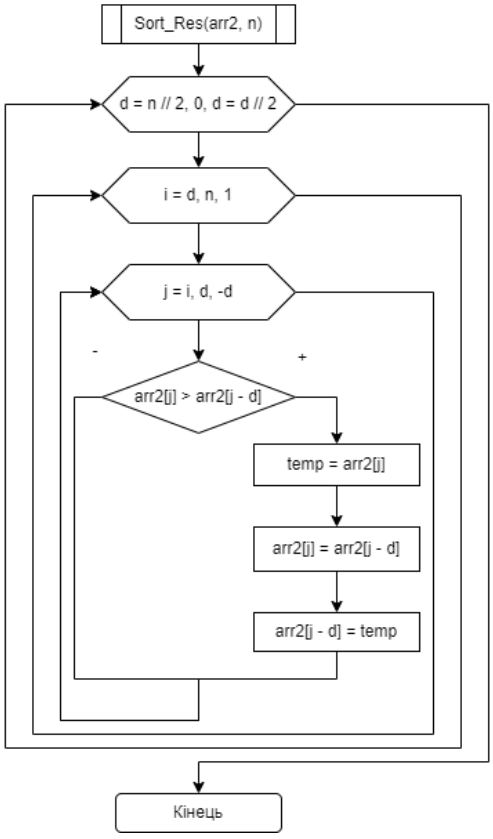
res = 1

```
            повторити для j від 0 до 6 з кроком 1
                якщо arr1 [i, j] < 0
                    то
                        res = res * arr1[i, j]
                все якщо
            все повторити
            якщо res != 1
                то
                    arr2[k] = res
                    k++
            все якщо
        все повторити
все процедура

процедура Sort_Res(arr2, n)
        повторити для d = n // 2 до 0 з кроком d = d // 2
            повторити для i від d до n з кроком 1
                повторити для j від i до d включно з кроком –d
                    якщо arr2[j] > arr2[j - d]
                        то
                            temp = arr2[j]
                            arr2[j] = arr2[j - d]
                            arr2[j - d] = temp
                    все якщо
                все повторити
            все повторити
        все повторити
все процедура
```

Блок-схема алгоритму





Код програми (C++)

```cpp

#include <iostream>
#include<iomanip>
using namespace std;

void Output_Res(double arr2[], int n);
void Sort_Res(double arr2[], int n);
void Innit_Matrix(double arr1[7][6]);
void Innit_Arr2(double arr1[7][6], double* arr2);
int Count_Elements(double arr1[7][6]);

int main()
{
    double arr1[7][6];
    double* arr2;
    int n;
    double temp;
    srand(time(NULL));
    cout.precision(4);

    Innit_Matrix(arr1);

    n = Count_Elements(arr1);
    arr2 = new double[n];

    Innit_Arr2(arr1, arr2);

    cout.precision(6);
    cout << "Unsorted result : ";
    Output_Res(arr2, n);
    cout << endl;

    Sort_Res(arr2, n);

    cout << "Sorted result : ";
    Output_Res(arr2, n);
    cout << endl;

    delete[] arr2;
}

```

```
40      }
41
42      void Output_Res(double arr2[], int n)
43      {
44          for (int i = 0; i < n; i++)
45              cout << arr2[i] << " ";
46          cout << endl;
47      }
48
49      void Sort_Res(double arr2[], int n)
50      {
51          double temp;
52          for (int d = n / 2; d > 0; d = d / 2)
53              for (int i = d; i < n; i++)
54                  for (int j = i; j >= d; j = j - d)
55                      if (arr2[j] > arr2[j - d])
56                      {
57                          temp = arr2[j];
58                          arr2[j] = arr2[j - d];
59                          arr2[j - d] = temp;
60                      }
61      }
62
63      void Innit_Matrix(double arr1[7][6])
64      {
65          for (int i = 0; i < 7; i++)
66          {
67              for (int j = 0; j < 6; j++)
68              {
69                  arr1[i][j] = (double)(rand()) / RAND_MAX * 40 - 20;
70
71                  cout << right << setw(7) << arr1[i][j] << " ";
72              }
73              cout << endl;
74          }
75      }
76
77      void Innit_Arr2(double arr1[7][6], double* arr2)
78      {
79          double  res;
```

```
77  void Innit_Arr2(double arr1[7][6], double* arr2)
78  {
79      double  res;
80      int k = 0;
81      for (int i = 0; i < 7; i++)
82      {
83          res = 1;
84          for (int j = 0; j < 6; j++)
85          {
86              if (arr1[i][j] < 0)
87              {
88                  res *= arr1[i][j];
89              }
90          }
91          if (res != 1)
92          {
93              arr2[k] = res;
94              k++;
95          }
96      }
97  }
98
99  int Count_Elements(double arr1[7][6])
100 {
101     int n = 0;
102     for (int i = 0; i < 7; i++)
103     {
104         for (int j = 0; j < 6; j++)
105         {
106             if (arr1[i][j] < 0)
107             {
108                 n += 1;
109                 break;
110             }
111
112         }
113     }
114     return n;
115 }
116
```

Тестування програми

```
 2.076   17.84  -9.166   16.84  -13.71  -11.92
-5.368   12.22   8.673   8.897   7.763   7.769
 12.29  -17.96  -7.944   16.61   1.914  -19.29
 5.358   12.89  -7.879   10.27  -15.64  -11.72
-17.36   7.158   12.78  -7.611  -17.76    12.3
-13.75  -16.78   2.353  -4.838    5.02   13.93
 1.722  -15.17   5.866   -17.8  -16.66  -15.77
Unsorted result : -1497.54 -5.36821 -2752.32 -1445.25 -2345.76 -1116.52 70913.6

Sorted result : 70913.6 -5.36821 -1116.52 -1445.25 -1497.54 -2345.76 -2752.32
```

Висновки

В даній роботі я дослідив алгоритми пошуку та сортування, набув практичних навичок їх використання під час складання алгоритмів та написання програм на мові програмування C++.