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 25

Виконав ІП-15, Плугатирьов Дмитро Валерійович

студент (шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська Анастасія Сергіївна

(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

Завдання

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

№ варіанта	Розмірність	Тип даних	Обчислення значень елементів одновимірного масиву
25	5 x 8	Цілий	Із добутку додатних значень елементів стовпців двовимірного масиву. Відсортувати методом бульбашки за зростанням.

1. Постановка задачі

Описати змінну індексованого типу, котра вміщує 5 елементів: послідовності змінних такого ж типу по вісім елементів кожна. Заповнити утворені послідовності випадковими цілими значеннями. Виконати добуток додатних елементів кожного стовпця утвореного двовимірного масиву, розміщуючи результати в новостворений послідовності. Відсортувати цю послідовність методом бульбашки за зростанням.

2. Побудова математичної моделі

Змінна	*Тип*	*Ім'я*	*Призначення*
Кількість рядків матриці	Цілочисельний	ROW	Початкові дані
Кількість колонок матриці	Цілочисельний	COL	Початкові дані
Матриця	Цілочисельний	arr[ROW][COL]	Початкові дані
Новоутворена послідовність	Цілочисельний	newArr[COL]	Початкові дані
Добуток додатних елементів стовпців	Підпрограма	mult_cols	Проміжні дані
Заповнення матриці	Підпрограма	fill_arr	Початкові дані
Сортування новоутвореної послідовності	Підпрограма	sort	Результат
Поточний індекс у новій послідовності	Цілочисельний	pos	Проміжні дані
Значення елемента послідовності	Цілочисельний	temp	Проміжні дані
Лічильник циклу	Цілочисельний	u	Проміжні дані
Лічильник циклу	Цілочисельний	i	Проміжні дані

Робота з матрицею відбувається переважно з використанням арифметичних вкладених циклів. В тому числі, й обрахунок значень елементів нової послідовності з участю стовпців матриці шляхом перемножування їх додатних елементів. Сортування відбувається з ліва на право методом бульбашки за зростанням: створюється тимчасова змінна для забезпечення обміном значень між сусідніми елементами за умови більшості лівішого.

Дія x *= i означає x := x * i.

Дія x++ означає x := x + 1.

Дія x := rand() % num1 – num2 означає присвоєння згенерованого псевдовипадковим чином цілого числа до змінної x в межах від num1 – num2 - 1 до –num2.

3. Р о з в ' я з а н н я

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

Крок 1. Визначити основні дії.

Крок 2. Ініціалізація новоствореної послідовності добутками додатних елементів.

Крок 3. Сортування послідовності за зростанням.

4. П с е в д о к о д

О с н о в н а п р о г р а м а:

Крок 1

початок

ініціалізація новоствореної послідовності добутками додатних елементів

сортування послідовності за зростанням

кінець

Крок 2

початок

ROW := 5

COL := 8

fill_arr(arr, ROW, COL)

mult_cols(arr, newArr, ROW, COL)

сортування послідовності за зростанням

кінець

Крок 3

початок

ROW := 5

COL := 8

fill_arr(arr, ROW, COL)

mult_cols(arr, newArr, ROW, COL)

sort(newArr, COL)

кінець

П і д п р о г р а м и:

fill_arr(arr, ROW, COL)

повторити для u **від** 0 **до** ROW

повторити для i **від** 0 **до** COL

arr[u][i] := rand() % 500 - 250

все повторити

все повторити

кінець

mult_cols(dArr, newArr, ROW, COL)

pos := 0

повторити для i **від** 0 **до** COL

повторити для u **від** 0 **до** ROW

якщо dArr[u][i] > 0

то

newArr[pos] *= dArr[u][i]

все якщо

все повторити

якщо newArr[pos] == 1

то

newArr[pos] := 0

все якщо

pos++

все повторити

кінець

sort(arr, SIZE)

повторити для u **від** 0 **до** SIZE

повторити для i **від** 0 **до** SIZE - 1

якщо arr[i] > arr[i + 1]

то

temp := arr[i]

arr[i] := arr[i + 1]

arr[i + 1] := temp

все якщо

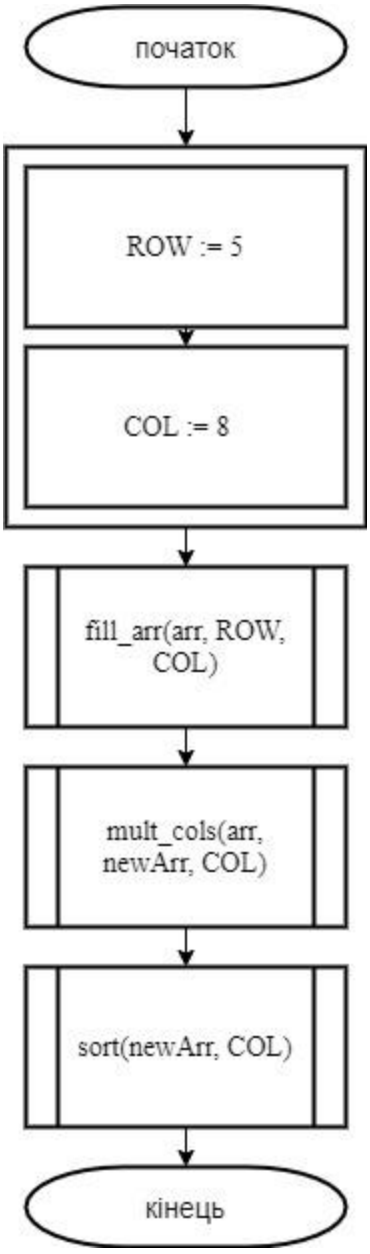
все повторити

все повторити

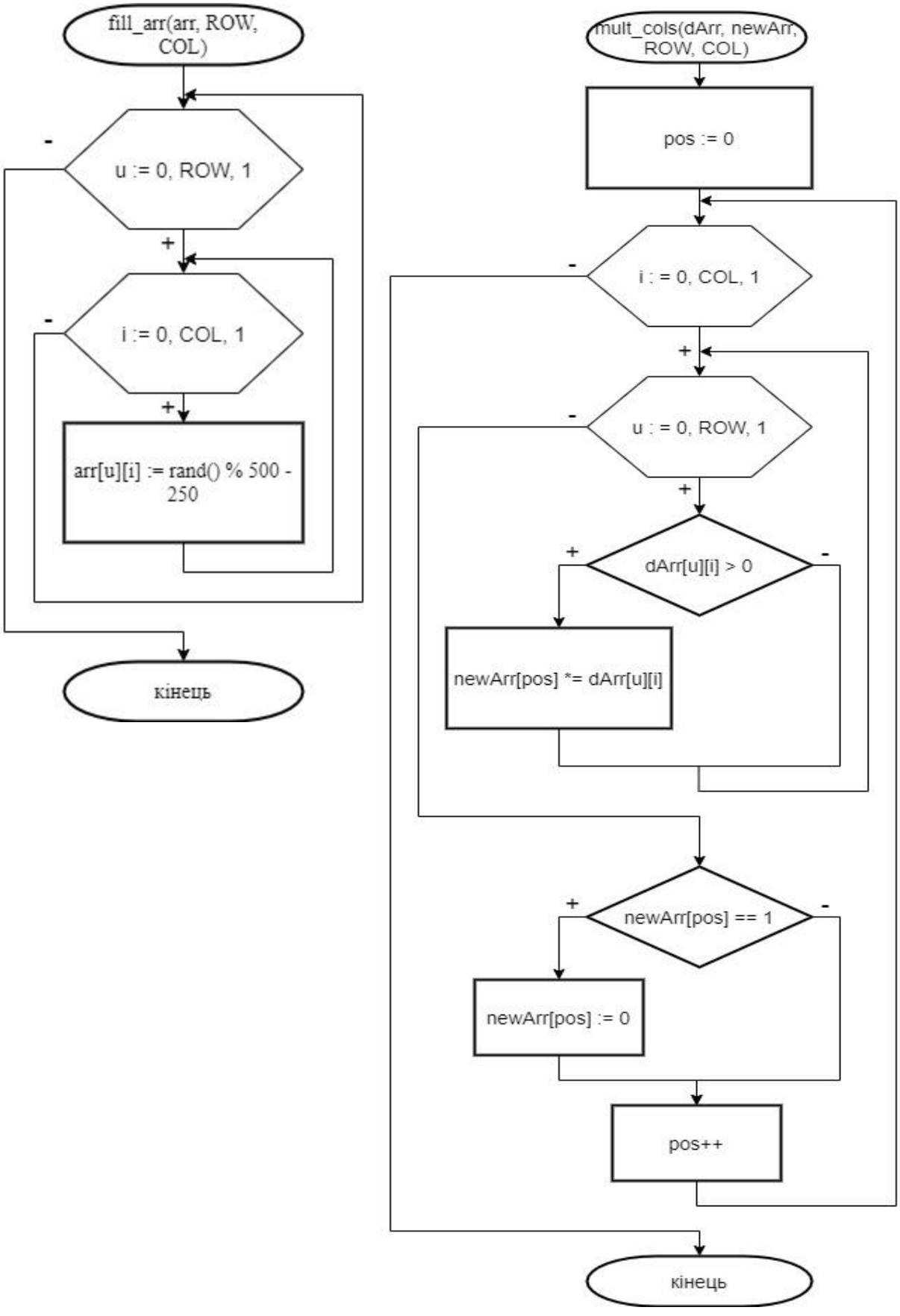
кінець

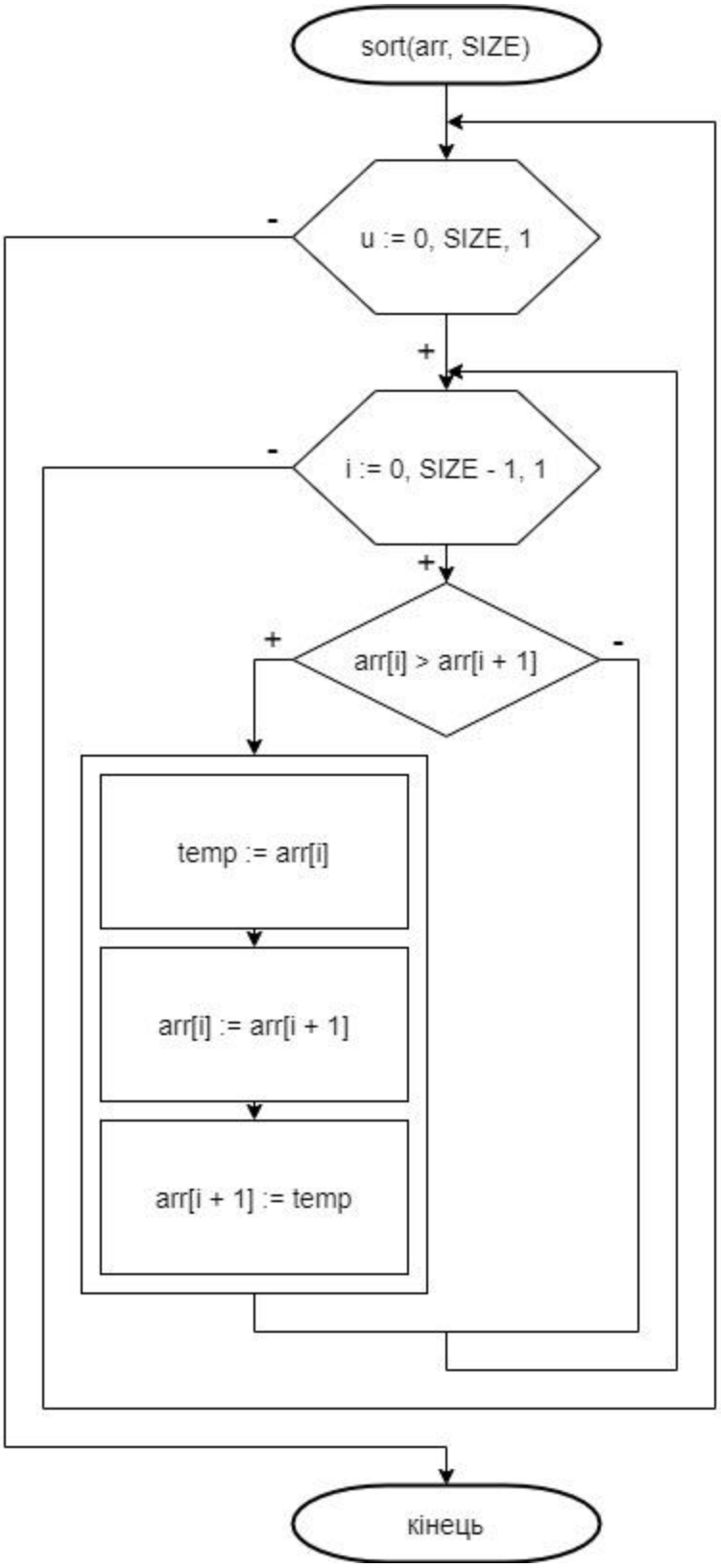
Блок-схема

О с н о в н а п р о г р а м а:


початок
ROW := 5
COL := 8
fill_arr(arr, ROW, COL)
mult_cols(arr, newArr, COL)
sort(newArr, COL)
кінець

П і д п р о г р а м и:




sort(arr, SIZE)
u := 0, SIZE, 1
-
+
i := 0, SIZE - 1, 1
-
+
arr[i] > arr[i + 1]
+
-
temp := arr[i]
arr[i] := arr[i + 1]
arr[i + 1] := temp
кінець

5. Код програми

Prototypes.h | Lab. work 8.cpp

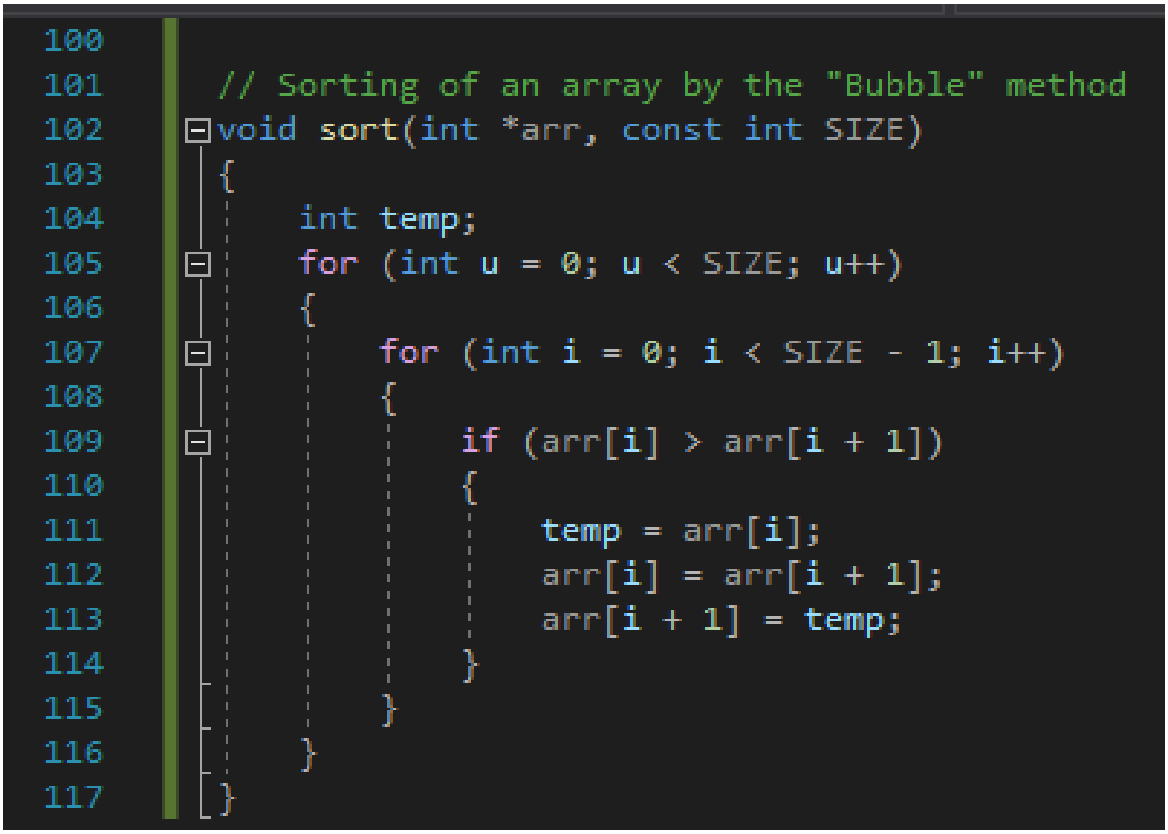
Lab. work 8 | (Глобальная область)

```
1   #include <iostream>
2   #include <iomanip>
3   #include <cstdlib>
4   #include <ctime>
5   #include "Prototypes.h"
6
7   /* ... */
11
12  int main()
13  {
14      // Basic input
15      srand(time(NULL));
16      const int ROW = 5, COL = 8;
17      int** arr = new int* [ROW], * newArr = new int[COL] {1, 1, 1, 1, 1, 1, 1, 1};
18      create_d_arr(arr, ROW, COL);
19      fill_arr(arr, ROW, COL);
20      print_arr(arr, ROW, COL);
21
22      // Calculations
23      mult_cols(arr, newArr, ROW, COL);
24      std::cout << "The created sequnce: " << std::endl;
25      print_arr(newArr, COL);
26      sort(newArr, COL);
27      std::cout << "\nThe sorted sequence: " << std::endl;
28      print_arr(newArr, COL);
29  }
30
31  // Creation of a double array
32  template<typename T>
33  void create_d_arr(T** arr, const int ROW, const int COL)
34  {
35      for (int i = 0; i < ROW; i++)
36      {
```

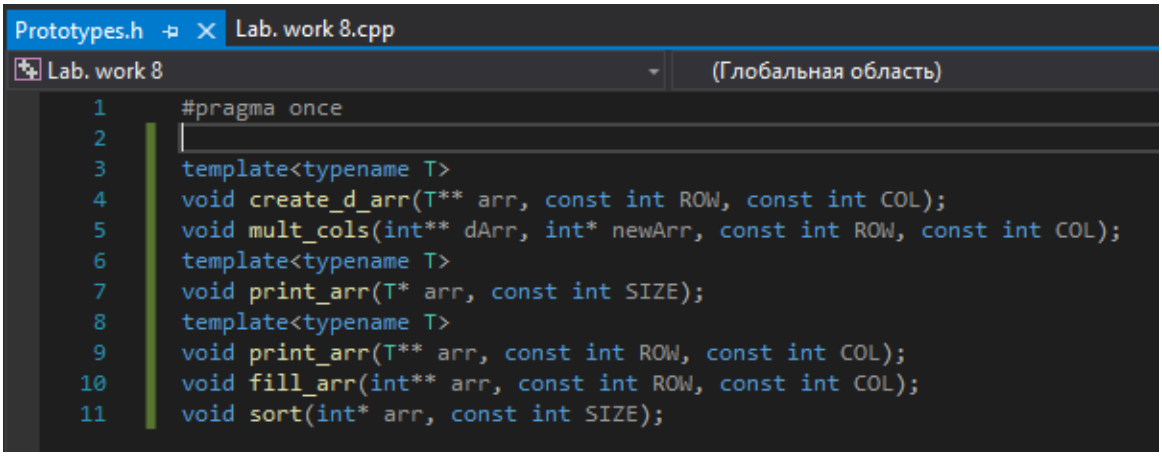
100 % | Проблемы не найдены. | 0 изменений | 0 авторов, 0 изменений

```
37          arr[i] = new int[COL];
38      }
39  }
40
41  // Multiplication of positive numbers in cols
42  void mult_cols(int **dArr, int *newArr, const int ROW, const int COL)
43  {
44      int pos = 0;
45      for (int i = 0; i < COL; i++)
46      {
47          for (int u = 0; u < ROW; u++)
48          {
49              if (dArr[u][i] > 0)
50              {
51                  newArr[pos] *= dArr[u][i];
52              }
53          }
54          if (newArr[pos] == 1)
55          {
56              newArr[pos] = 0;
57          }
58          pos++;
59      }
60  }
61
62  // Print of an array
63  template<typename T>
64  void print_arr(T* arr, const int SIZE)
65  {
66      std::cout << "Here is a sequence:\n";
67      for (int i = 0; i < SIZE; i++)
68      {
69          std::cout << arr[i] << " ";
```

```
70      }
71      std::cout << std::endl;
72  }
73
74  template<typename T>
75  void print_arr(T** arr, const int ROW, const int COL)
76  {
77      std::cout << "Here is a scope of sequences:\n";
78      for (int u = 0; u < ROW; u++)
79      {
80          for (int i = 0; i < COL; i++)
81          {
82              std::cout << std::setw(7) << arr[u][i];
83          }
84          std::cout << std::endl;
85      }
86      std::cout << std::endl;
87  }
88
89  // Initialization of an array by random numbers
90  void fill_arr(int** arr, const int ROW, const int COL)
91  {
92      for (int u = 0; u < ROW; u++)
93      {
94          for (int i = 0; i < COL; i++)
95          {
96              arr[u][i] = rand() % 500 - 250;
97          }
98      }
99  }
100
101 // Sorting of an array by the "Bubble" method
102 void sort(int *arr, const int SIZE)
```



```
100
101     // Sorting of an array by the "Bubble" method
102     void sort(int *arr, const int SIZE)
103     {
104         int temp;
105         for (int u = 0; u < SIZE; u++)
106         {
107             for (int i = 0; i < SIZE - 1; i++)
108             {
109                 if (arr[i] > arr[i + 1])
110                 {
111                     temp = arr[i];
112                     arr[i] = arr[i + 1];
113                     arr[i + 1] = temp;
114                 }
115             }
116         }
117     }
```

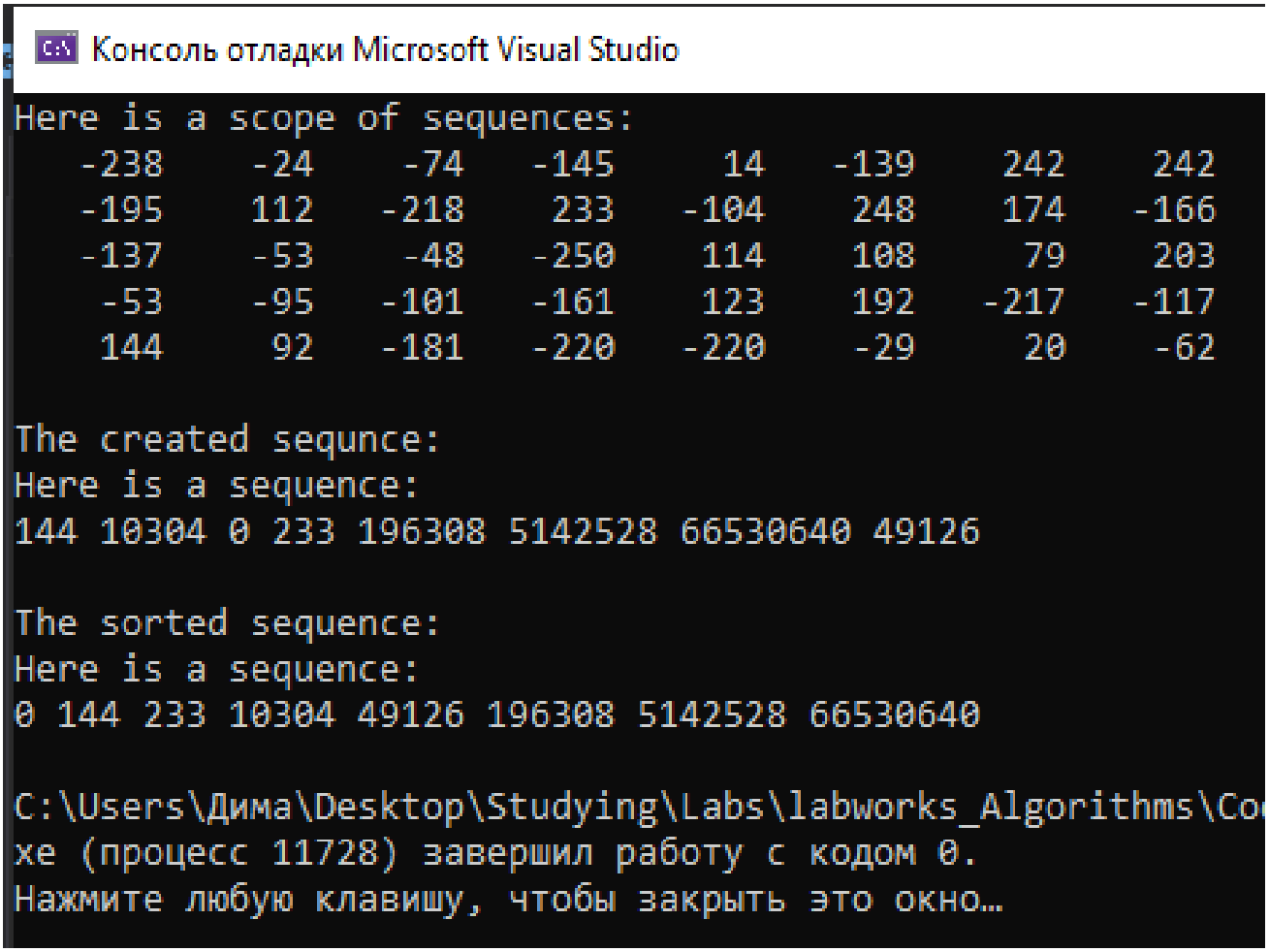


Prototypes.h | Lab. work 8.cpp

Lab. work 8 | (Глобальная область)

```
1   #pragma once
2
3   template<typename T>
4   void create_d_arr(T** arr, const int ROW, const int COL);
5   void mult_cols(int** dArr, int* newArr, const int ROW, const int COL);
6   template<typename T>
7   void print_arr(T* arr, const int SIZE);
8   template<typename T>
9   void print_arr(T** arr, const int ROW, const int COL);
10  void fill_arr(int** arr, const int ROW, const int COL);
11  void sort(int* arr, const int SIZE);
```

5. Т е с т у в а н н я



6. В и с н о в о к

На цій лабораторній роботі я дослідив алгоритми пошуку та сортування та набув практичних навичок використання цих алгоритмів під час складання програмних специфікацій. Мною був використаний метод «бульбашки» для сортування послідовності за зростанням.