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 №7 з дисципліни «Алгоритми та структури даних-1. Основи алгоритмізації»

«Дослідження лінійного пошуку в послідовностях»

Варіант 29

Виконав студент ІП-11 Тарасьонок Дмитро Євгенович

(шифр, прізвище, ім'я, по батькові)

Перевірив Мартинова Оксана Петрівна

( прізвище, ім'я, по батькові)

Київ 2021

**Лабораторна робота 7**

**Дослідження рекурсивних алгоритмів**

**Мета**: дослідити методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набути практичних навичок їх використання під час складання програмних специфікацій.

**Варіант 29**

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

# 1. Опису трьох змінних індексованого типу з 10 символьних значень.

# 2. Ініціювання двох змінних виразами i + 58 та 63 - i

# 3. Ініціювання третьої змінної рівними значеннями двох попередніх змінних.

# 4. Виведення елементів, менших за середньоарифметичне

1. **Постановка задачі:**

Постановка задачі зводиться до того, що нам необхідно знайти спільні елементи двох масивів та вивести ті, коди яких менші за середнє арифметичне всіх цих кодів

# Побудова математичної моделі. Складемо таблицю імен змінних:

|  |  |  |  |
| --- | --- | --- | --- |
| Змінна | Тип | Ім’я | Призначення |
| Перший масив | Масив символів | arr1 | Початкові дані |
| Другий масив | Масив символів | arr2 | Початкові дані |
| Третій масив | Масив символів | res\_arr | Проміжні дані |
| Спільні елементи | Цілий | common\_elements | Проміжні дані |
| Сума елементів | Цілий | sum\_of\_elements | Проміжні дані |
| Середнє арифметичне | Дійсний | average | Проміжні дані |
| Чи належить елемент масиву | Булевий | is\_in\_arr | Результат функції |

Для вирішення даної задачі необхідно ініціалізувати два масиви початковими даними, після чого пройтись по всіх елементах першого масиву і перевірити наявність такого елемента в другому масиві за допомогою підпрограми лінійного пошуку. Далі необхідно знайти суму всіх елементів третього масиву, після чого пройтися по всіх елементах цього масиву й вивести ті, що менші за середнє арифметичне.

Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію ініціювання перших двох змінних

Крок 3. Деталізуємо дію ініціювання третьої змінної

Крок 4. Деталізуємо дію знаходження середнього арифметичного

Крок 5. Деталізуємо дію виведення всіх елементів, менших за середнє арифметичне

1. **Псевдокод:**

*Крок 1*

# Початок

Опис змінних

Ініціювання перших двох змінних

Ініціювання третьої змінної

Знаходження середнього арифметичного

Виведення всіх елементів, менших за середнє арифметичне

**Кінець**

*Крок 2*

# Початок

arr1[10]

arr2[10]

arr3[10]

Ініціювання перших двох змінних

Ініціювання третьої змінної

Знаходження середнього арифметичного

Виведення всіх елементів, менших за середнє арифметичне

**Кінець**

*Крок 3*

# Початок

arr1[10]

arr2[10]

arr3[10]

**для i від 1 до 10**

**повторити**

arr1[i] = char(i+58)

arr2[i] = char(63-i)

**все повторити**

Ініціювання третьої змінної

Знаходження середнього арифметичного

Виведення всіх елементів, менших за середнє арифметичне

**Кінець**

*Крок 4*

# Початок

arr1[10]

arr2[10]

arr3[10]

**для i від 1 до 10**

**повторити**

arr1[i] = char(i+58)

arr2[i] = char(63-i)

**все повторити**

common\_elements := 0

**для i від 1 до 10**

**повторити**

is\_in\_arr := search(arr2, arr1[i])

**якщо is\_in\_arr == True**

**то**

res\_arr[common\_elements + 1] := arr[i]

common\_elements := common\_elements + 1

**все якщо**

**все повторити**

Знаходження середнього арифметичного

Виведення всіх елементів, менших за середнє арифметичне

**Кінець**

*Крок 5*

# Початок

arr1[10]

arr2[10]

arr3[10]

**для i від 1 до 10**

**повторити**

arr1[i] = char(i+58)

arr2[i] = char(63-i)

**все повторити**

common\_elements := 0

**для i від 1 до 10**

**повторити**

is\_in\_arr := search(arr2, arr1[i])

**якщо is\_in\_arr == True**

**то**

res\_arr[common\_elements + 1] := arr[i]

common\_elements := common\_elements + 1

**все якщо**

**все повторити**

sum\_of\_elements := 0

**для i від 1 до common\_elements**

**повторити**

sum\_of\_elements := sum\_of\_elements + res\_arr[i]

**все повторити**

average := sum\_of\_elements div common\_elements

Виведення всіх елементів, менших за середнє арифметичне

**Кінець**

*Крок 6*

# Початок

arr1[10]

arr2[10]

arr3[10]

**для i від 1 до 10**

**повторити**

arr1[i] = char(i+58)

arr2[i] = char(63-i)

**все повторити**

common\_elements := 0

**для i від 1 до 10**

**повторити**

is\_in\_arr := search(arr2, arr1[i])

**якщо is\_in\_arr == True**

**то**

res\_arr[common\_elements + 1] := arr[i]

common\_elements := common\_elements + 1

**все якщо**

**все повторити**

sum\_of\_elements := 0

**для i від 1 до common\_elements**

**повторити**

sum\_of\_elements := sum\_of\_elements + res\_arr[i]

**все повторити**

average := sum\_of\_elements div common\_elements

**для i від 1 до common\_elements**

**повторити**

**якщо res\_arr[i] < average**

**то**

Виведення res\_arr[i]

**все якщо**

**все повторити**

**Кінець**

**Початок search(arr, element)**

is\_in\_arr := False

**для i від 1 до 10**

**повторити**

**якщо arr[i] == element**

**то**

is\_in\_arr := True

**все якщо**

**все повторити**

**повернути is\_in\_arr**

**Кінець search(arr, element)**

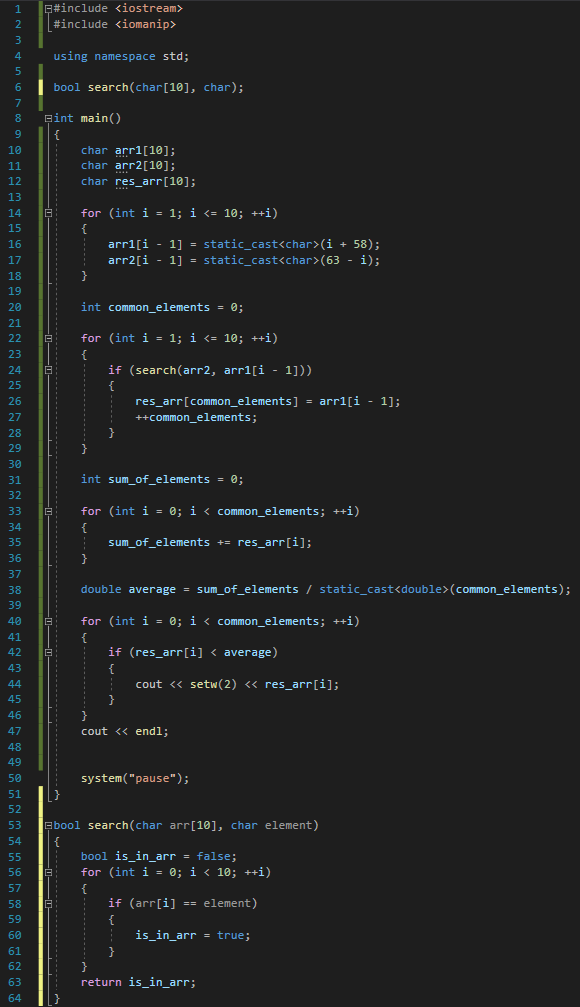
**4. Блок-схема:**



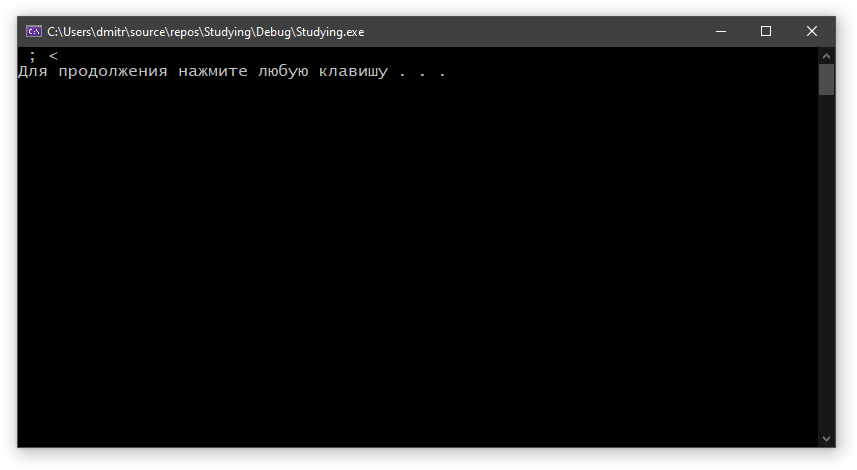
# 5. Випробування алгоритму:

|  |  |
| --- | --- |
| **Блок** | **Дія** |
|  | **Початок** |
| **1** | arr1 = [;, <, =, >, ?, @, A, B, C, D]  arr2 = [>, =, <, ;, :, 9, 8, 7, 6, 5] |
| **2** | res\_arr = [;, <, =, >] |
| **3** | sum\_of\_elements = 242 |
| **4** | average = 60.5 |
| **5** | Виведення ; < |
|  | **Кінець** |

1. **Код програми мовою C++:**



1. **Виконання програми мовою C++:**



# 8. Висновки:

За допомогою математичної моделі та алгоритму лінійного пошуку можна запрограмувати знаходження спільних елементів двох масивів. Для виконання своєї задачі я спроектував арифметичний цикл для ініціювання двох масивів, арифметичний цикл для ініціювання третього, розробив підпрограму, за допомогою якої й здійснював пошук, після чого за допомогою арифметичного циклу обрахував середнє арифметичне, а потім за допомогою арифметичного циклу та оператора умови вивів усі елементи масиву, що мають код менший за середнє арифметичне. Особливістю написання коду стало те, що в С++ нумерація масивів починається з нуля, а не одиниці. В результаті виконання роботи я набув необхідних знань для використання алгоритму лінійного пошуку.