

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 23

Виконав студент    ІП-15, Мочалов Дмитро Юрійович

Перевірив        Вечєрковська Анастасія Сергіївна

Київ 2021

## Лабораторна робота 7

### Дослідження лінійного пошуку в послідовностях

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

#### Варіант 23

Задача.

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 згідно з варіантом (табл. 1).
3. Ініціювання третьої змінної рівними значеннями двох попередніх змінних.
4. Обробки третьої змінної згідно з варіантом.

23	$35 + 3 * i$	$56 - 2 * i$	Середнє арифметичне елементів, коди яких більше 38
----	--------------	--------------	--

#### Мат. модель

Змінна	Тип	Ім'я	Призначення
перший масив	Символ	a	результат
другий масив	Символ	b	результат
третій масив	Символ	c	результат
Середнє арифметичне	Дійсний	avg	результат
Задання значень першого та другого масиву	процедура	create_a_b	проміжний
Задання значень третього масиву	процедура	create_c	проміжний
ітератор	цілий	i	проміжний
ітератор	цілий	j	проміжний
Індекс масиву c	цілий	ind	проміжний
Кількість елементів c	цілий	count	проміжний
Сума елементів c	цілий	sum	проміжний
результат	дійсний	result	результат

Таким чином, математичне модулювання зводиться до генерування значень масивів a та b за допомогою циклу у підпрограмі create\_a\_b(), масив c за допомогою циклу з перевіркою еквівалентності значень a та b, якщо так то записати значення або елементу a або елементу b до масиву c, робимо це у підпрограмі create\_c. За допомогою підпрограми avg знаходимо середнє арифметичне елементів масиву c, у яких код більше 38. Для збільшення значення змінної на 1 використовуємо ++,

Крок1: визначитись з алгоритмом

Крок2: деталізуємо алгоритм задання значень масивам за допомогою підпрограм

Крок2: деталізуємо алгоритм знаходження середнього арифметичного за допомогою підпрограми.

### **Псевдокод**

Крок1

#### **Початок**

деталізуємо алгоритм задання значень масивам

деталізуємо алгоритм знаходження середнього арифметичного

#### **Кінець**

Крок2

#### **Початок**

create\_a\_b(a,b)

create\_c(a,b,c)

деталізуємо алгоритм знаходження середнього арифметичного

#### **Кінець**

### **Підпрограми:**

create\_a\_b(a,b)

#### **початок**

**повторити для i від 0 до 9**

$a[i] = 35 + 3*i$

$b[i] = 56 - 2*i$

**все повторити**

#### **кінець**

create\_c(a,b,c)

**початок**

ind = 0

**повторити для і від 0 до 9**

**повторити для j від 0 до 9**

**якщо a[i] == b[j] то**

c[ind] = a[i]

ind++

**все якщо**

**все повторити**

**все повторити**

**кінець**

Крок3

**Початок**

create\_a\_b(a,b)

create\_c(a,b,c)

result = avg(c)

**Кінець**

**Підпрограми:**

create\_a\_b(a,b)

**початок**

**повторити для і від 0 до 9**

a[i] = 35 + 3\*i

b[i] = 56 - 2\*i

**все повторити**

**кінець**

create\_c(a,b,c)

**початок**

ind = 0

**повторити для і від 0 до 9**

**повторити для j від 0 до 9**

**якщо a[i] == b[j] то**

c[ind] = a[i]

ind++

**все якщо**

**все повторити**

**все повторити**

**кінець**

avg(c)

**початок**

count = 0

sum = 0

**повторити для і від 0 до 9**

**якщо c[i] > 38 то**

count++

suma = suma + c[i]

**все якщо**

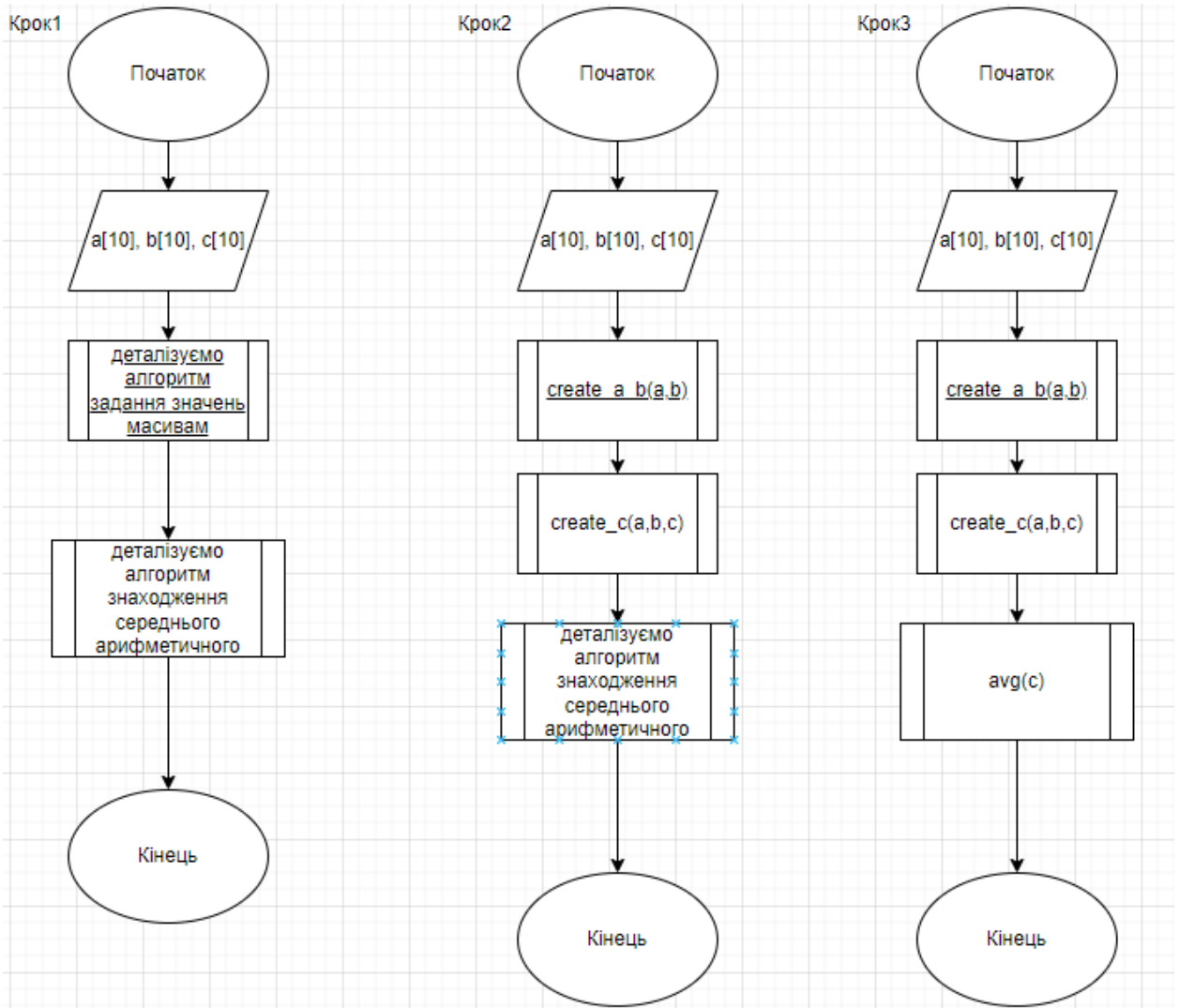
**все повторити**

avg = suma / count

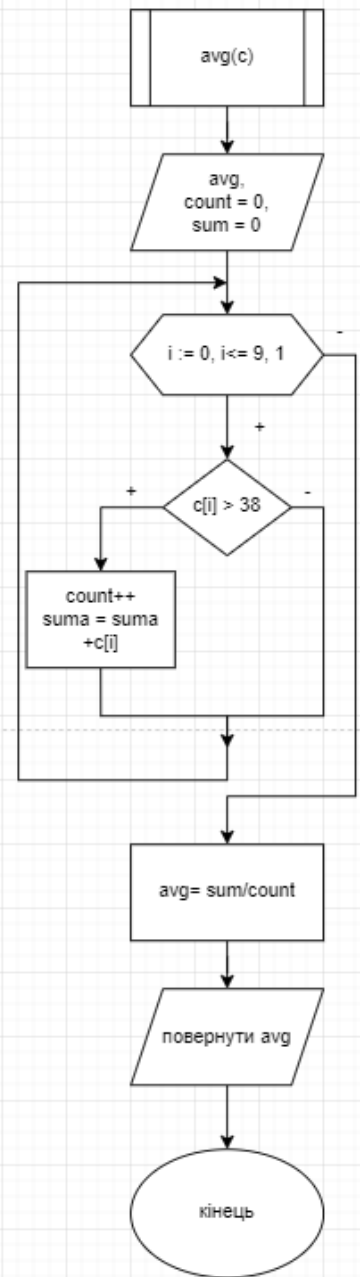
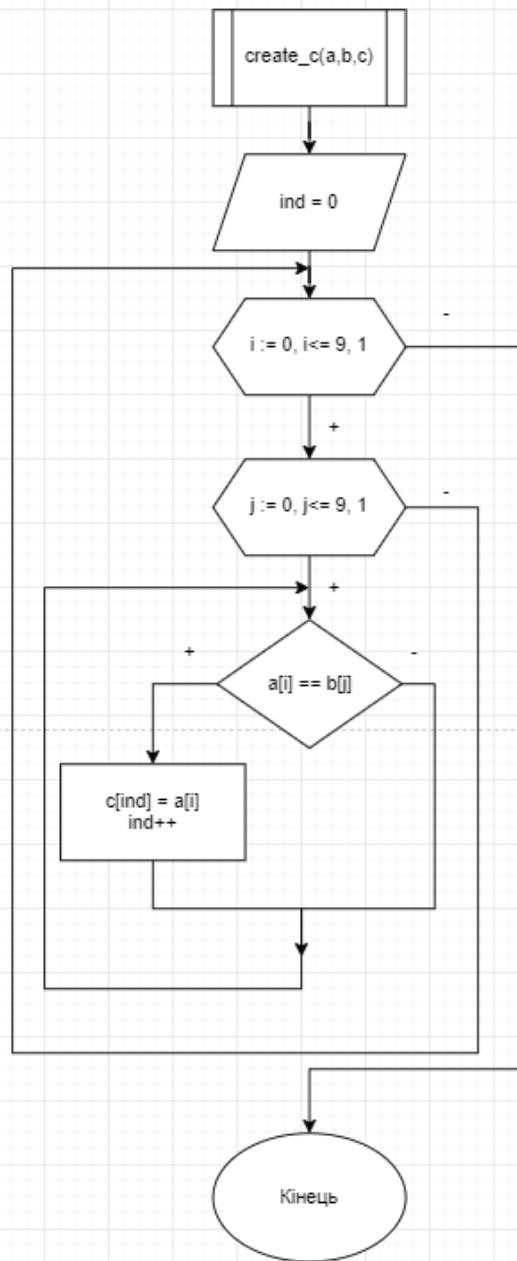
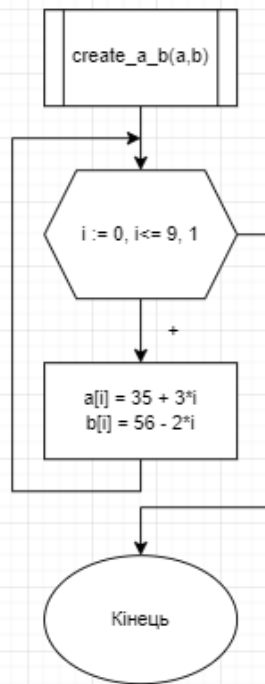
**повернути avg**

**кінець**

## Блок-схема



# Підпрограми



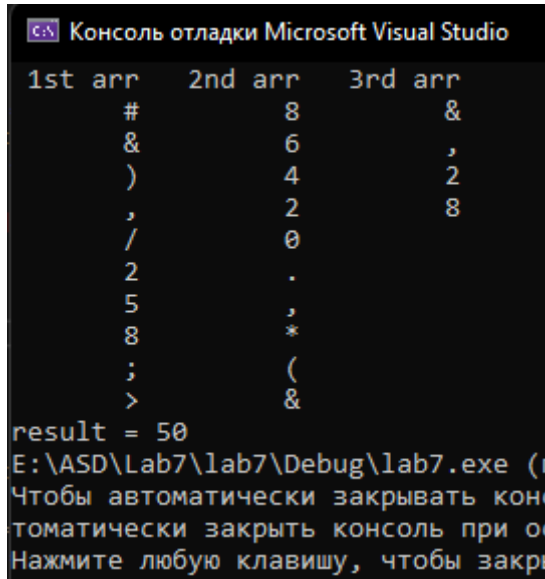
## Код

```
lab7.cpp  X
lab7 (Глобальная обла

1  #include <iostream>
2  #include <stdio.h>
3  #include <iomanip>
4
5  using namespace std;
6
7  void get_arr(char a[], char b[]) {
8      for (int i = 0; i < 10; i++) {
9          a[i] = 35 + 3 * i;
10         b[i] = 56 - 2 * i;
11     }
12 }
13
14 float avg(char c[]) {
15     float avg;
16     int count = 0, sum = 0;
17     for (int i = 0; i < 10; i++) {
18         if (c[i] > 38) {
19             count++;
20             sum += c[i];
21         }
22     }
23     avg = sum / count;
24     return avg;
25 }
26
27 void create_c(char a[], char b[], char c[]) {
28     int ind = 0;
29     for (int i = 0; i < 10; i++) {
30         for (int j = 0; j < 10; j++) {
31             if (a[i] == b[j]) {
32                 c[ind] = int(a[i]);
33                 ind++;
34             }
35         }
36     }
37 }
38
39 int main()
40 {
41     char a[10], b[10], c[10];
42
43     for (int i = 0; i < 10; i++) {
44         c[i] = 0;
45     }
46
47     get_arr(a, b);
48     create_c(a, b, c);
49
50     cout << setw(8) << "1st arr" << setw(10) << "2nd arr" << setw(10) << "3rd arr" << endl;
51     for (int i = 0; i < 10; i++) {
52         cout << setw(8) << a[i] << setw(10) << b[i] << setw(10) << c[i] << endl;
53     }
54     cout << "result = " << avg(c);
55 }
```



## Тестування



```
Консоль отладки Microsoft Visual Studio
1st arr    2nd arr    3rd arr
#          8          &
&          6          ,
)          4          2
,          2          8
/          0
2          .
5          ,
8          *
;          (
>          &

result = 50
E:\ASD\Lab7\lab7\Debug\lab7.exe (1)
Чтобы автоматически закрывать консоль при отладке, нажмите Ctrl+F5.
Чтобы автоматически закрыть консоль при отладке, нажмите Ctrl+F5.
Нажмите любую клавишу, чтобы закрыть консоль.
```

**Висновок:** Ми дослідили методи послідовного пошуку у впорядкованих і невлпорядкованих послідовностях та набули практичних навичок їх використання під час складання програмних специфікацій.