

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 __16__

Виконав студент __ІП-15, Куманецька І. В.____
Перевірив _Вечерковська А. С._____

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.

Індивідуальне завдання

Варіант 16

Завдання

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.

16	$115 + i$	$125 - 2 * i$	Суму двох максимальних елементів
----	-----------	---------------	----------------------------------

Постановка задачі

Створити два масиви за формулами $115+i$ та $125-2*i$. Створити третій масив з однакових значень перших двох. Знайти в ньому два максимальні елементи та вирахувати їх суму.

Побудова математичної моделі

Змінна	Тип	Ім'я	Призначення
Перший масив	Символьний	first	Проміжні дані
Другий масив	Символьний	second	Проміжні дані
Третій масив	Символьний	equals	Проміжні дані
Кількість елементів у масивах	Ціле число	length	Вхідні дані

Кількість елементів у третьому масиві	Ціле	quantity	Проміжні дані
Перший масив, що розглядається у функції	Символьний	list_1	Проміжні дані
Другий масив, що розглядається у функції	Символьний	list_2	Проміжні дані
Лічильник, можливий індекс	Ціле	i	Проміжні дані
Лічильник, можливий індекс	Ціле	j	Проміжні дані
Максимальний елемент третього масиву	Символьний	max_el	Проміжні дані
Другий максимальний елемент третього масиву	Символьний	sec_max_el	Проміжні дані
Сума двох максимальних елементів третього масиву	Ціле	res	Проміжні дані

first та second генеруються за допомогою функції get_lists(first, second) за відповідними формулами. Третій масив генерується за допомогою функції get_equal(first, second, equals) пошуком однакових значень перших двох масивів. Функція sum_max(equals) перебирає усі елементи третього масиву, знаходить два найбільших та повертає їх суму.

Через $a += b$ позначається операція $a = a + b$.

Розв'язання

Програмні специфікації запишемо у псевдокодi, графічній формi у вигляді блок-схеми та у вигляді коду.

Крок 1. Визначення основних дій.

Крок 2. Деталізація генерації перших двох масивів за допомогою підпрограми.

Крок 3. Деталізація генерації третього масиву за допомогою підпрограми.

Крок 4. Деталізація знаходження суми двох максимальних елементів третього масиву.

Псевдокод

початок

length := 10

get_lists(first, second)

get_equal(first, second, equals)

res := sum_max(equals)

виведення res

кінець

Підпрограми

get_lists(list_1, list_2)

повторити для i від 1 до length

list_1[i] := 115 + i

list_2[i] := 125 - 2*i

все повторити

кінець

get_equal(list_1, list_2, list_3)

quantity := 0

повторити для i від 1 до length

повторити для j від 1 до length

якщо list_1[i]=list_2[j]

то

quantity += 1

list_3[quantity] := list_1[i]

все якщо

все повторити

все повторити

кінець

sum_max(list_1)

max_el := 0

sec_max_el := 0

повторити для i від 1 до length

якщо list_1[i] > max_el

то

sec_max_el := max_el

max_el := list_1[i]

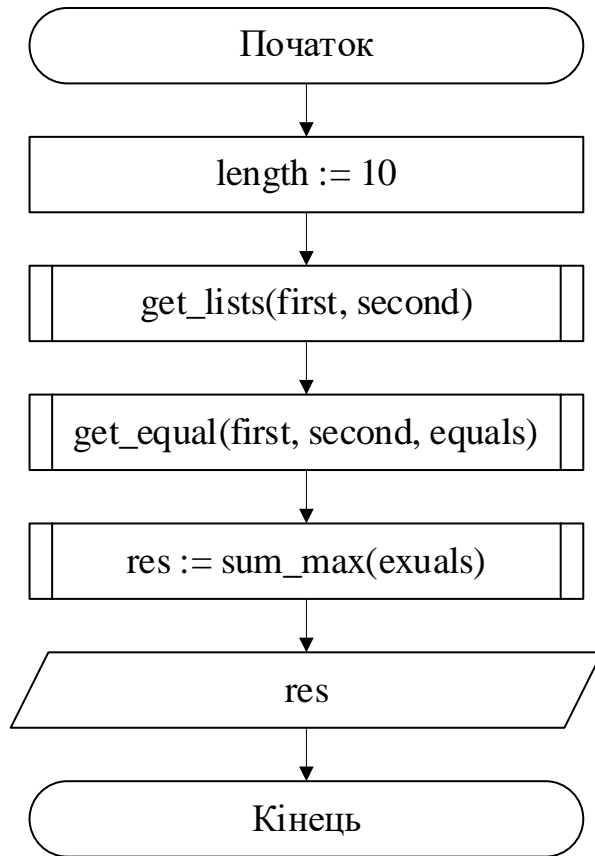
все якщо

все повторити

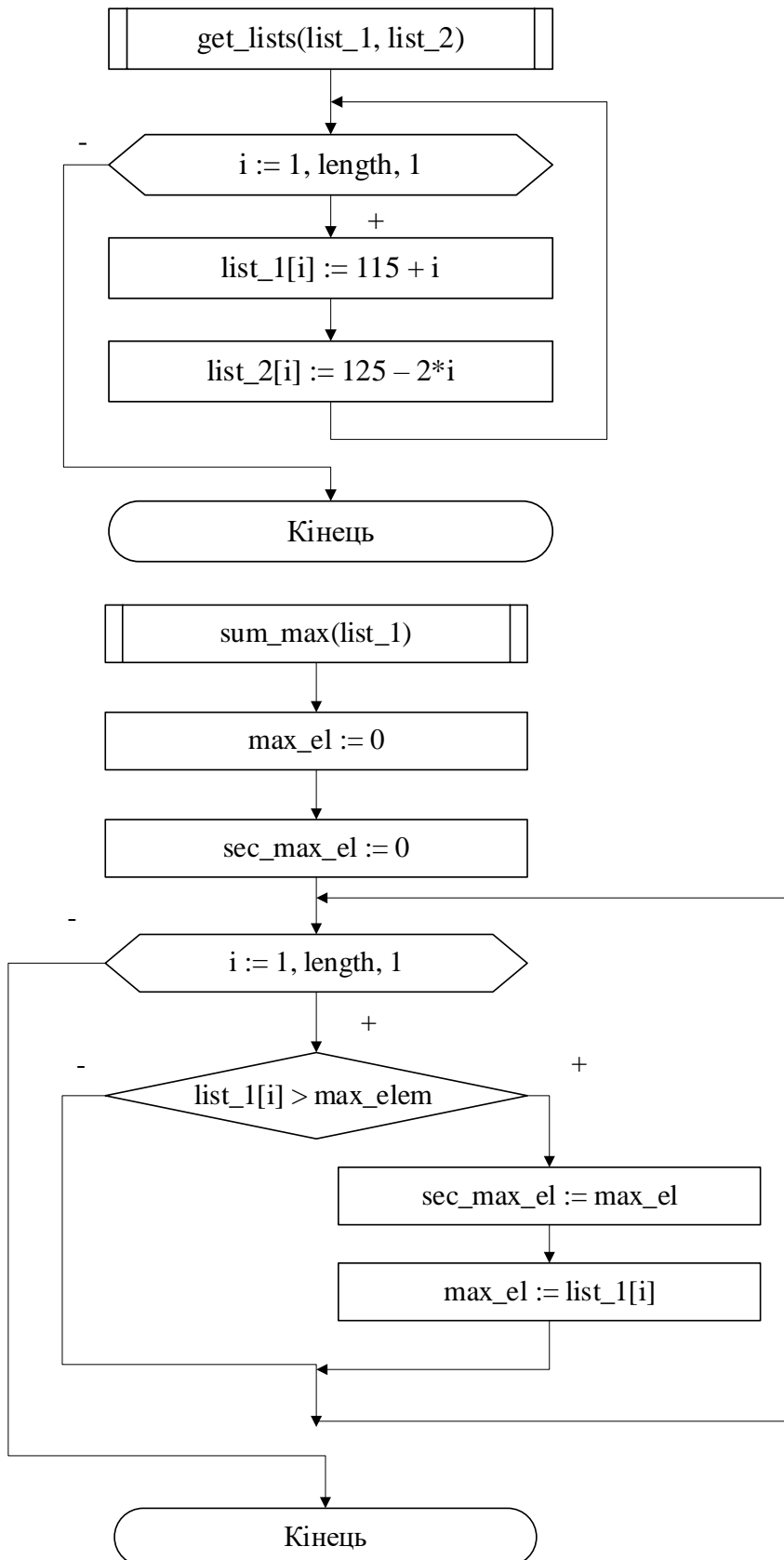
повернути (max_el + sec_max_el)

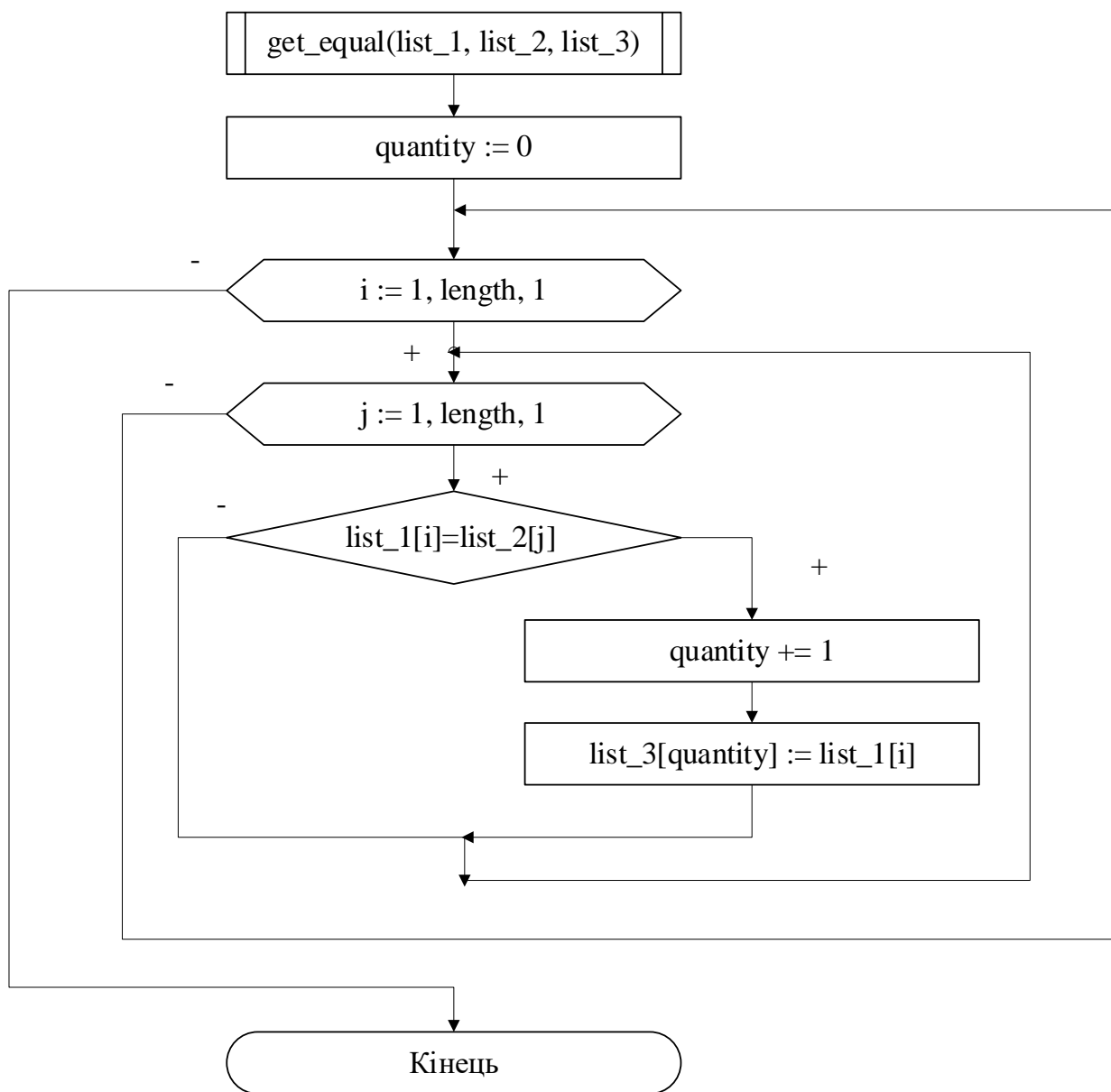
кінець

Блок-схема



Підпрограми





Код

```
#include <stdio.h>
#include <math.h>
#include <iostream>

using namespace std;

const int length = 10;

int main()
{
    char first[10], second[10], equals[10]{};
    void get_lists(char[], char[]), get_equal(char[], char[], char[]);
    get_lists(first, second);
    get_equal(first, second, equals);

    int sum_max(char[]), res;
    res = sum_max(equals);

    cout << "The result: " << res;
}

//генеруємо перші два масиви
void get_lists(char list_1[], char list_2[]) {
    for (int i = 0; i < length; i++)
    {
        list_1[i] = 115 + i;
        list_2[i] = 125 - 2 * i;
    }
    //виводимо масиви
    for (int i = 0; i < length; i++)
    {
        cout << list_1[i] << " ";
    }
    cout << endl;
    for (int i = 0; i < length; i++)
    {
        cout << list_2[i] << " ";
    }
    cout << endl;
}
```

```

//генеруємо третій масив з однакових елементів
void get_equal(char list_1[], char list_2[], char list_3[]) {
    int quantity = 0;
    for (int i = 0; i < length; i++)
    {
        for (int j = 0; j < length; j++)
        {
            if (list_1[i] == list_2[j])
            {
                list_3[quantity] = list_1[i];
                quantity += 1;
            }
        }
    }
    //виводимо третій масив
    for (int i = 0; i < length; i++)
    {
        cout << list_3[i] << " ";
    }
    cout << endl;
}

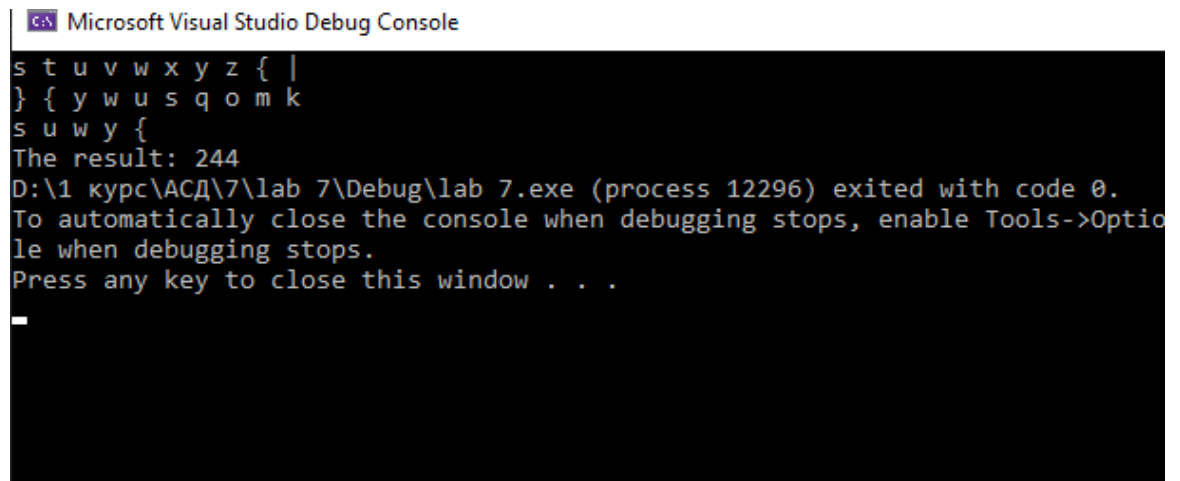
//сума двох макс значень 3-го масиву
int sum_max(char list_1[]) {
    int max_el = 0, sec_max_el = 0;

    for (int i = 0; i < length; i++)
    {
        if (list_1[i] > max_el)
        {
            sec_max_el = max_el;
            max_el = list_1[i];
        }
    }

    return max_el + sec_max_el;
}

```

Тестування коду



```
Microsoft Visual Studio Debug Console
s t u v w x y z { |
} { y w u s q o m k
s u w y {
The result: 244
D:\1 курс\АСД\7\lab 7\Debug\lab 7.exe (process 12296) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->
  Automatically close console when debugging stops.
Press any key to close this window . . .
```

Висновок

Було досліджено методи послідовного пошуку у впорядкованих і неупорядкованих послідовностях та набуто практичних навичок їх використання під час складання програмних специфікацій. В результаті виконання лабораторної роботи було знайдено суму двох максимальних елементів масиву, утвореного однаковими елементами інших двох, розділивши задачу на 4 кроки: визначення основних дій, деталізація генерації перших двох масивів за допомогою підпрограми, деталізація генерації третього масиву за допомогою підпрограми, деталізація знаходження суми двох максимальних елементів третього масиву. В процесі випробування було розраховано значення 244.