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 28

Виконав студент ІП-11 Сідак Кирил Ігорович
(шифр, прізвище, ім'я, по батькові)

Перевірив Мартинова Оксана Петрівна
(прізвище, ім'я, по батькові)

Київ 2021

Лабараторна робота№7

Дослідження лінійного пошуку в послідовностях

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

Індивідуальне завдання:

Варіант 28

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.

28	66 + 3 * i	78 - i	Суму кодів мінімального та максимального елементів

Постановка задачі

Потрібно описати 3 змінні індексованого типу з 10 символьних значень, тобто три масиви символьного, які містять 10 елементів. Потім, використовуючи арифметичні цикли, заповнити 2 з них відповідно до умови. Третій масив треба заповнити спільними елементами для 1 та 2 масивів, тобто ті, що мають однаковий код. У третьому масиві треба знайти елементи з максимальним та мінімальним кодом та знайти їх суму.

Побудова математичної моделі

Складемо таблицю змінних

Змінна	Тип	Ім’я	Призначення
Перший масив	символьний	arr1	Вхідне дане
Другий масив	символьний	arr2	Вхідне дане
Третій масив	символьний	arr3	Проміжне дане
Мінімальний код символу в	цілий	min_code	Проміжне дане

третьому масиві			
Максимальний код символу в третьому масиві	цілий	max_code	Проміжне дане
Сума мінімального та максимального кодів третьому масиві	цілий	sum	Результат
Зміна для пошуку мінімального(максимального) коду в третьому масиві	цілий	temp_code	Проміжне дане, змінна підпрограми
Індекс в третьому масиві, в який треба помістити спільний елемент для першого та другого масивів	цілий	k	Проміжне дане, змінна підпрограми

Таким чином, формування задачі зводиться до заповнення першого та другого масиву символів за допомогою підпрограми fill_arrays, яка використовує арифметичний цикл для заповнення масивів. Використовуючи підпрограму display_array, виведемо перший та другий масиви. За допомогою підпрограми fill_third_array заповнимо третій масив спільними елементами з першого та другого масивів, тобто такими, які мають однаковий код. Потім, використовуючи підпрограми find_max_code та find_min_code, знайдемо максимальний та мінімальний коди третього масиву. Обчислимо шукане значення sum, тобто суму цих кодів.

Розв’язання

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

Крок 1. Визначимо основні дії

Крок 2. Деталізуємо заповнення перших двох масивів

Крок 3. Деталізуємо виведення масиву

Крок 4. Деталізуємо заповнення третього масиву

Крок 5. Деталізуємо знаходження мінімального коду третього масиву

Крок 6. Деталізуємо знаходження максимального коду третього масиву

Псевдокод

Основна програма

Початок

n := 10

arr1[n], arr2[n], arr3[n]

fill_arrays(arr1, arr2, n)

виведення “Array 1:\n”

display_array(arr1, n)

виведення “Array 2:\n”

display_array(arr2, n)

fill_third_array(arr1, arr2, arr3, n)

виведення “Array 3:\n”

display_array(arr3, n)

min_code := find_min_code(arr3, n)

виведення “Min code is “, min_code, “\n”

max_code := find_max_code(arr3, n)

виведення “Max code is “, max_code, “\n”

sum := min_code + max_code

виведення “Sum of code of minimal and maximum elements is ”, sum

Кінець

Підпрограми

fill_arrays(arr1, arr2, length)

повторити для i **від** 0 **до** length

arr1[i] := 66 + 3 * i

arr2[i] := 78 – i

все повторити

Кінець fill_arrays

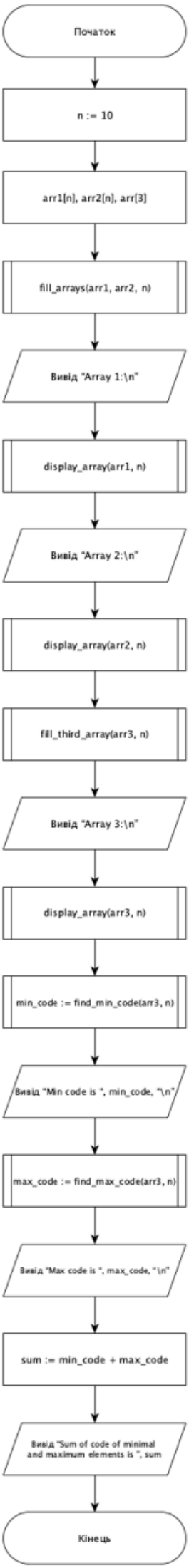
display_array(arr, length)

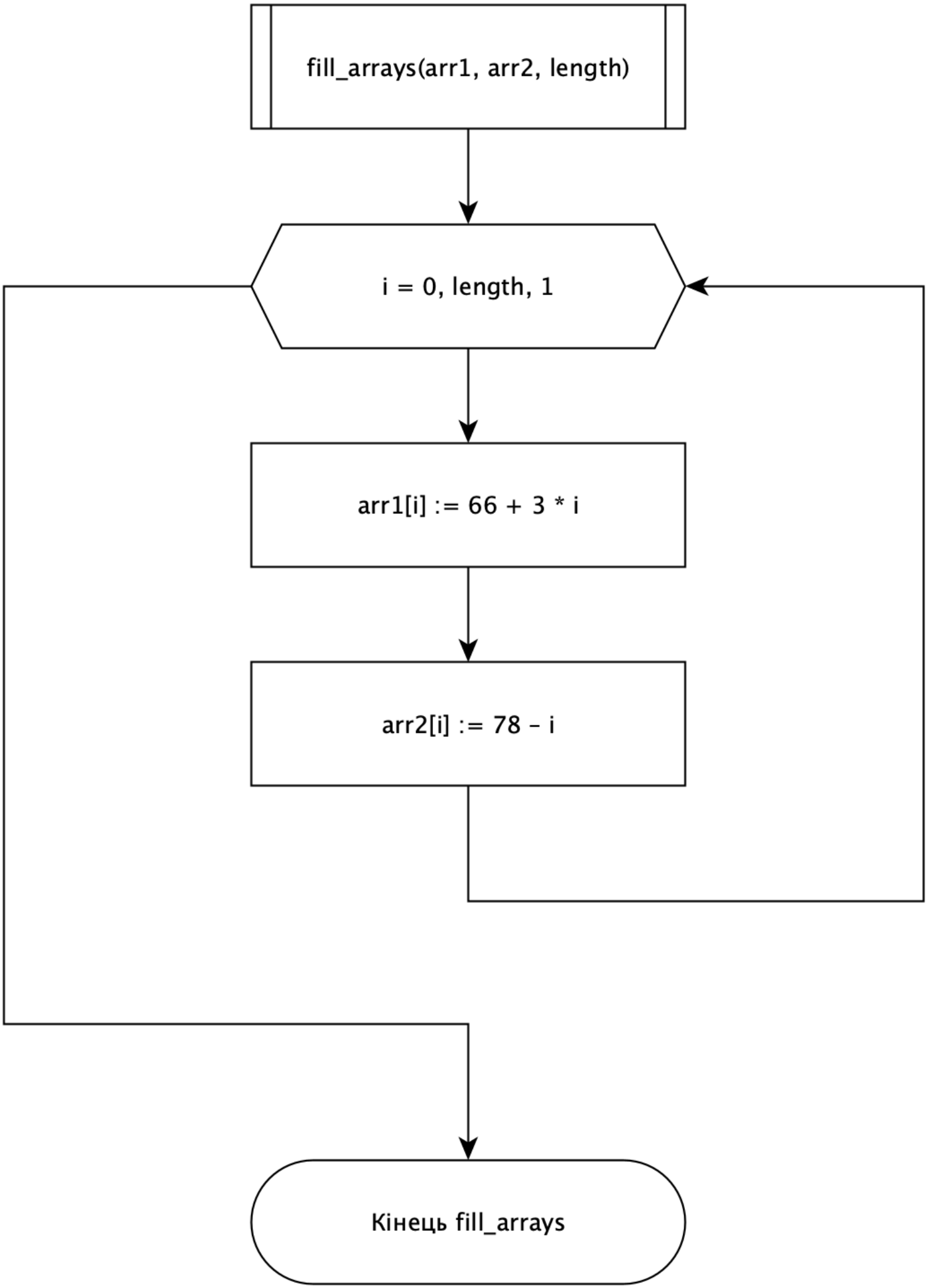
повторити для i **від** 0 **до** length

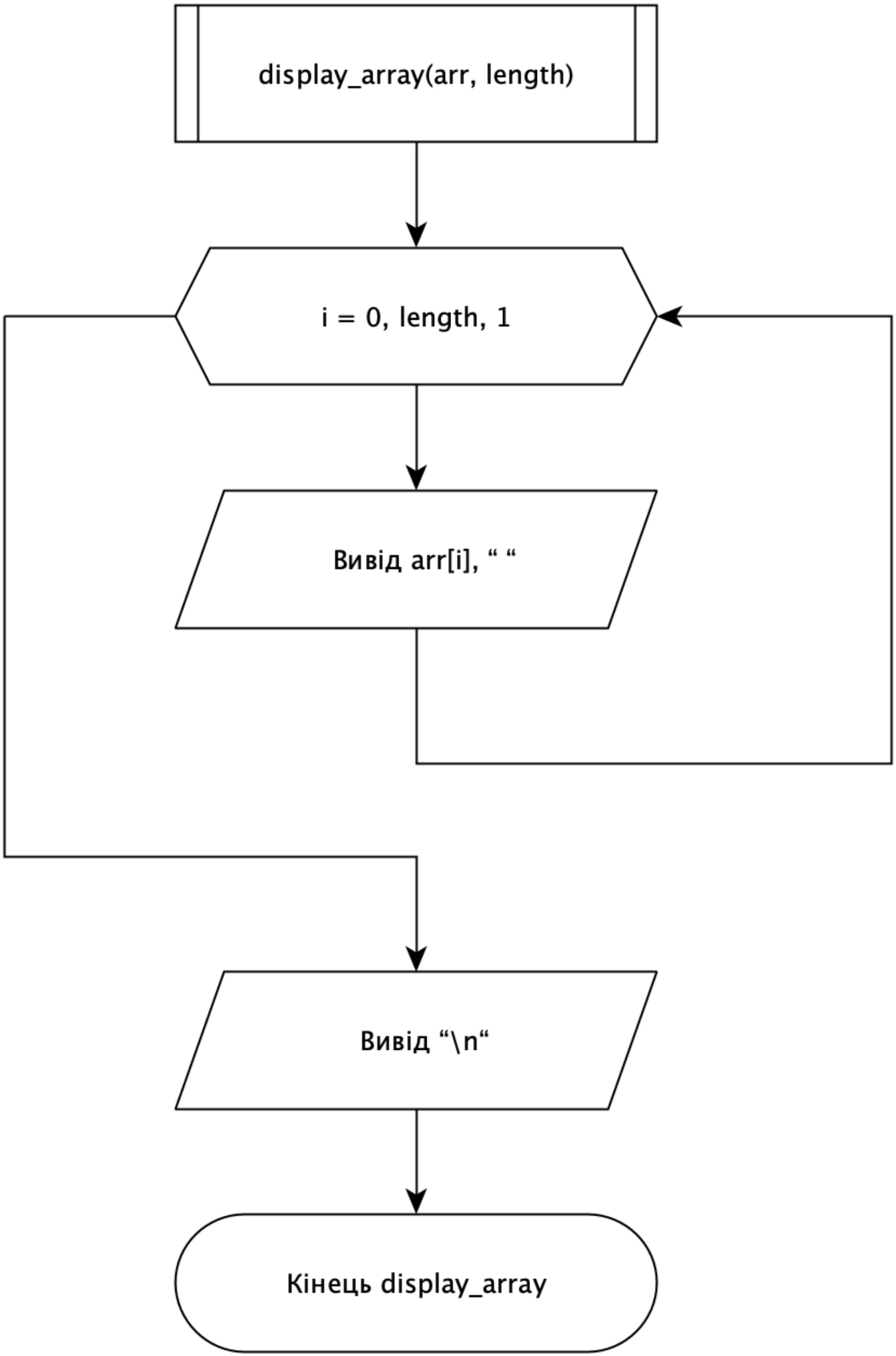
```
        виведення arr[i], “ “
    все повторити
    виведення “\n”
Кінець display_array

fill_third_array(arr1, arr2, arr3, length)
    k := 0
    повторити для i від 0 до length
        повторити для j від 0 до length
            якщо arr1[i] == arr2[j]
                то
                    arr3[k] := arr1[i]
                    k += 1
            все якщо
        все повторити
    все повторити
Кінець fill_third_array

min_code(arr, length)
    i := 1
    temp_code := arr[0]
    поки arr[i] != 0 і i < length повторити
        якщо arr[i] < temp_code
            то
                temp_code := arr[i]
        все якщо
        i += 1
    все повторити
    повернути temp_code
Кінець min_code
```

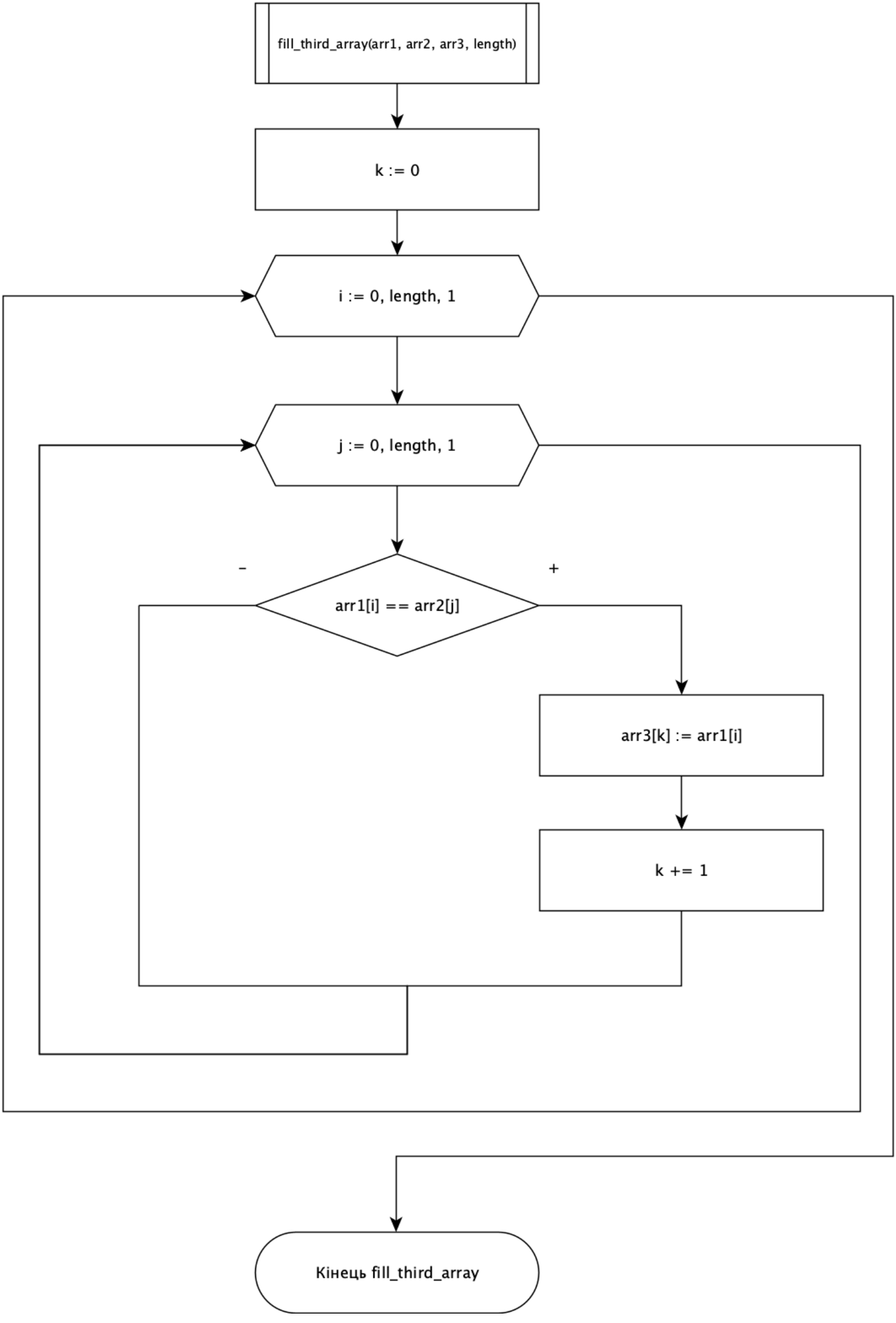
```
max_code(arr, length)
        i := 1
        temp_code := arr[0]
        поки arr[i] != 0 і i < length повторити
            якщо arr[i] > temp_code
               то
                  temp_code := arr[i]
            все якщо
            i += 1
        все повторити
        повернути temp_code
Кінець max_code
```

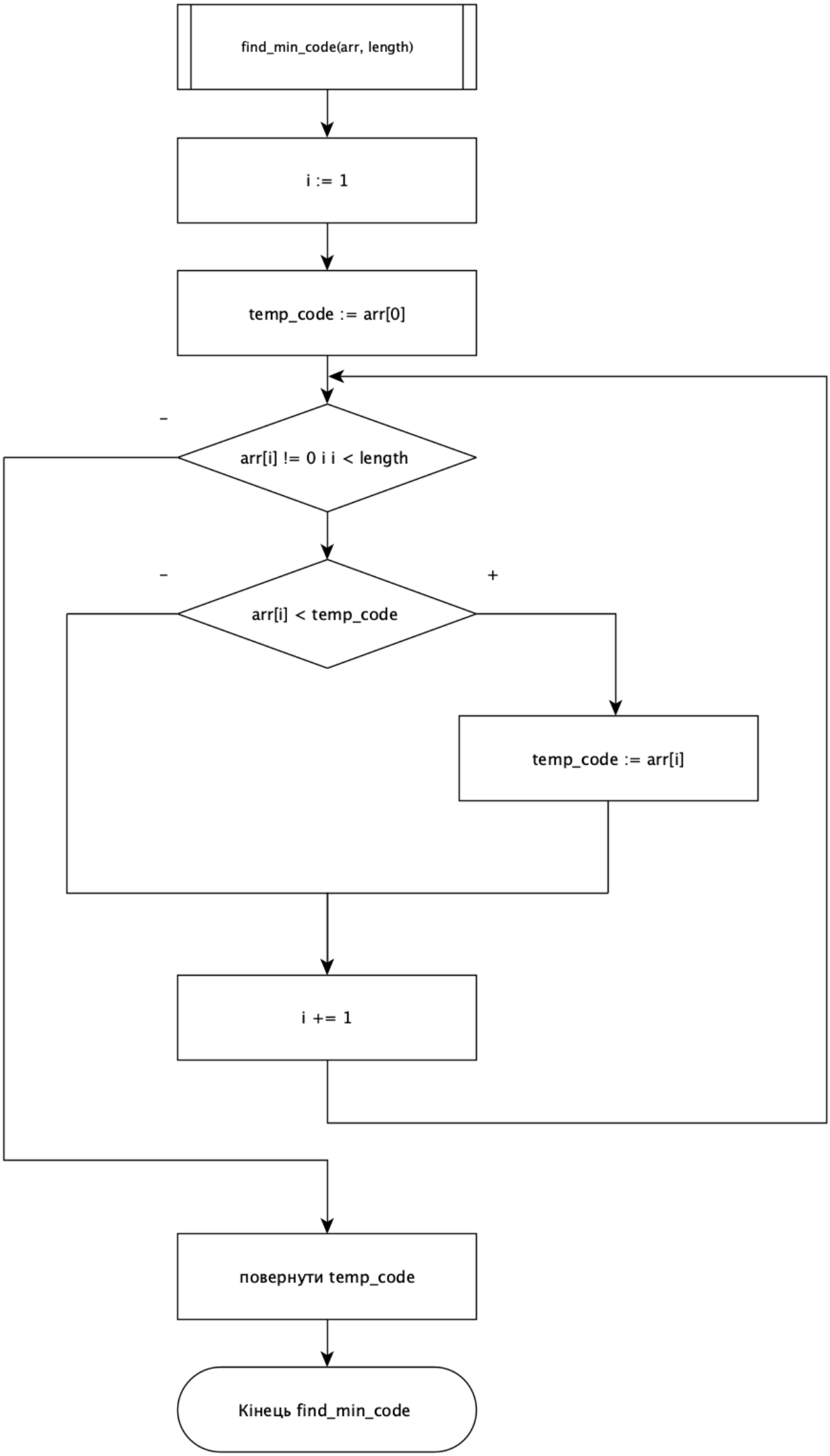
Блок-схема

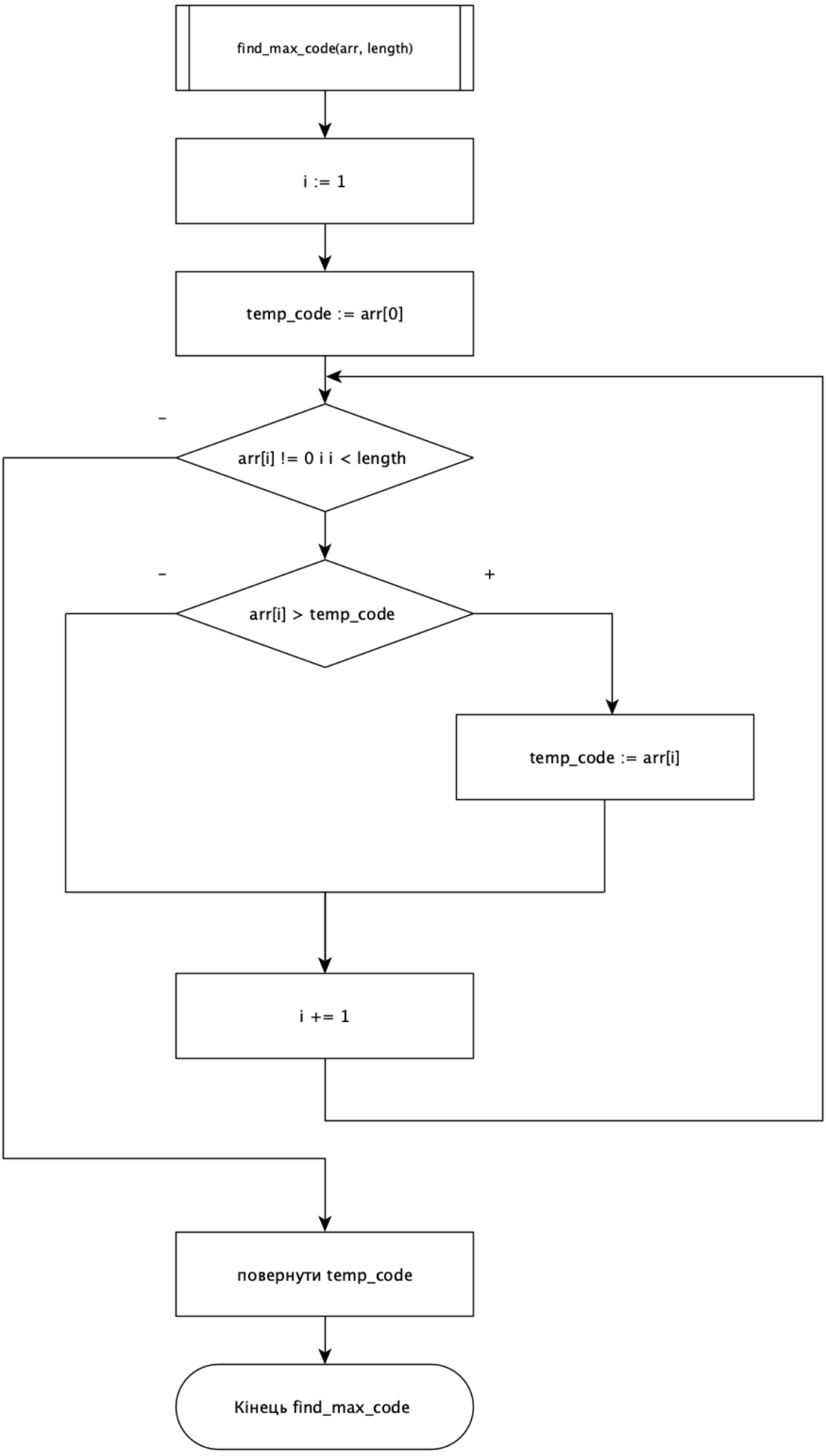

Початок
n := 10
arr1[n], arr2[n], arr[3]
fill_arrays(arr1, arr2, n)
Вивід "Array 1:\n"
display_array(arr1, n)
Вивід "Array 2:\n"
display_array(arr2, n)
fill_third_array(arr3, n)
Вивід "Array 3:\n"
display_array(arr3, n)
min_code := find_min_code(arr3, n)
Вивід "Min code is ", min_code, "\n"
max_code := find_max_code(arr3, n)
Вивід "Max code is ", max_code, "\n"
sum := min_code + max_code
Вивід "Sum of code of minimal and maximum elements is ", sum
Кінець


fill_arrays(arr1, arr2, length)
i = 0, length, 1
arr1[i] := 66 + 3 * i
arr2[i] := 78 - i
Кінець fill_arrays


display_array(arr, length)
i = 0, length, 1
Вивід arr[i], “ “
Вивід “\n“
Кінець display_array


fill_third_array(arr1, arr2, arr3, length)
k := 0
i := 0, length, 1
j := 0, length, 1
arr1[i] == arr2[j]
-
+
arr3[k] := arr1[i]
k += 1
Кінець fill_third_array

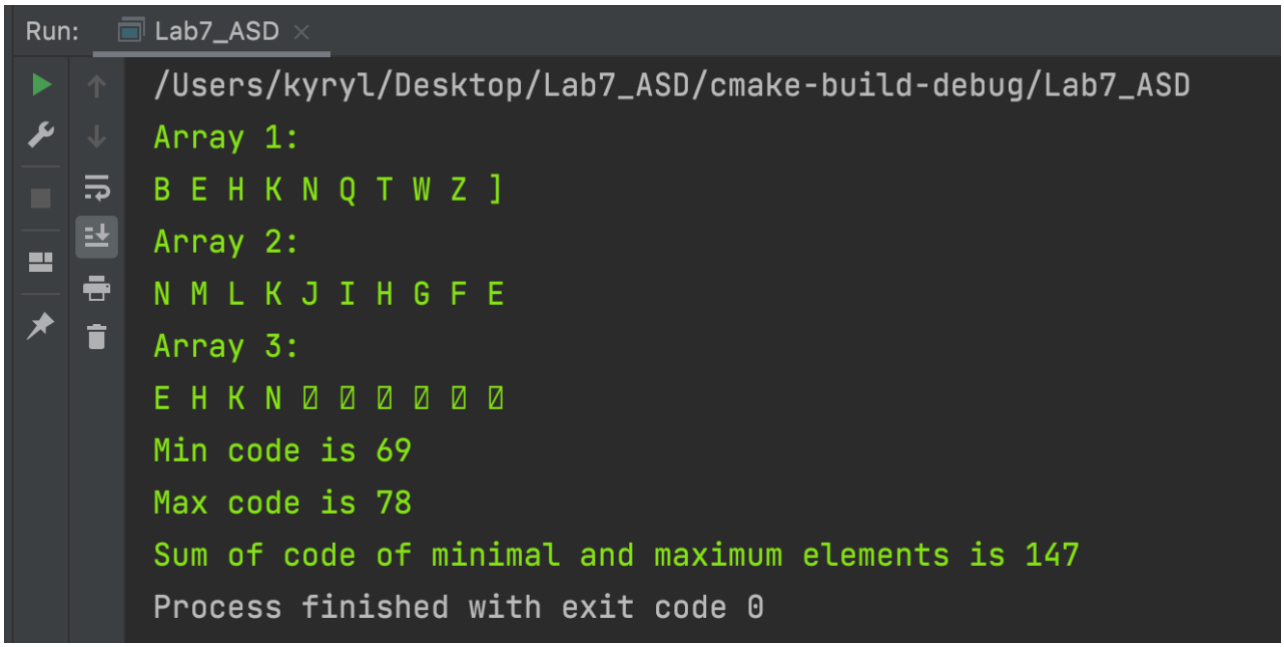

find_min_code(arr, length)
i := 1
temp_code := arr[0]
arr[i] != 0 і і < length
arr[i] < temp_code
temp_code := arr[i]
i += 1
повернути temp_code
Кінець find_min_code


find_max_code(arr, length)
i := 1
temp_code := arr[0]
arr[i] != 0 і і < length
arr[i] > temp_code
temp_code := arr[i]
i += 1
повернути temp_code
Кінець find_max_code

Програма на C++

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

void fill_arrays(char [], char[], int);
void display_array(char [], int);
void fill_third_array(char[], char[], char[], int);
int find_max_code(char [], int);
int find_min_code(char [], int);

int main() {
    const int n = 10;
    char arr1[n], arr2[n], arr3[n];
    int sum, min_code, max_code;
    fill_arrays(arr1, arr2, n);
    cout << "Array 1:\n";
    display_array(arr1, n);
    cout << "Array 2:\n";
    display_array(arr2, n);
    fill_third_array(arr1, arr2, arr3, n);
    cout << "Array 3:\n";
    display_array(arr3, n);
    min_code = find_min_code(arr3, n);
    cout << "Min code is " << min_code << '\n';
    max_code = find_max_code(arr3, n);
    cout << "Max code is " << max_code << '\n';
    sum = min_code + max_code;
    cout << "Sum of code of minimal and maximum elements is " <<
sum;
    return 0;
}

void fill_arrays(char arr1[], char arr2[], int length) {
    for (int i = 0; i < length; i++) {
        arr1[i] = (char)(66 + 3 * i);
        arr2[i]= (char)(78 - i);
    }
}

void display_array(char arr[], int length) {
    for (int i = 0; i < length; i++) {
        cout << arr[i] << " ";
    }
    cout << '\n';
}

void fill_third_array(char arr1[], char arr2[], char arr3[], int
length) {
    int k = 0;
    for (int i = 0; i < length; i++) {
        for (int j = 0; j < length; j++) {
            if (arr1[i] == arr2[j]) {
                arr3[k] = arr1[i];
                k++;
            }
```

```
        }
    }
}

int find_min_code(char arr[], int length) {
    int i = 1;
    int temp_code = (int)arr[0];
    while((int)arr[i] != 0 && i < length) {
        if ((int)arr[i] < temp_code) {
            temp_code = (int)arr[i];
        }
        i++;
    }
    return temp_code;
}

int find_max_code(char arr[], int length) {
    int i = 1;
    int temp_code = (int)arr[0];
    while ((int)arr[i] != 0 && i < length) {
        if ((int)arr[i] > temp_code) {
            temp_code = (int)arr[i];
        }
        i++;
    }
    return temp_code;
}
```



Випробування алгоритму

Блок	Дія
	Початок
1	n = 10
2	Виведення: Array 1: B E H K N Q T W Z]
3	Виведення: Array 2: N M L K J I H G F E
4	Виведення: Array 3: E H K N
5	Виведення: Min code is 69
6	Виведення: Max code is 78
7	Виведення: Sum of code of minimal and maximum elements is 147
	Кінець

Висновок

Отже, я дослідив методи послідовного пошуку в послідовностях та набув практичних навичок їх використання, створивши алгоритм для заповнення двох масивів за заданою умовою, пошуку спільних елементів для цих масивів та пошуку найменшого та найбільшого елемента в третьому масиві, заповненого цими спільними елементами. Порахувавши їх суму, я отримав коректний результат.