# Алгоритми та структури даних. Основи алгоритмізації

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра автоматизованих систем обробки інформації і управління

Звіт

з лабораторної роботи №7 з дисципліни
«Алгоритми та структури даних.
Основи алгоритмізації»

«Дослідження лінійного пошуку в послідовностях»

Варіант___4__

Виконав студент _____Берлінський Ярослав Владленович_____
(шифр, прізвище, ім'я, по батькові)

Перевірив _____
( прізвище, ім'я, по батькові)

Київ 2020

**Назва роботи:** дослідження лінійного пошуку в послідовностях.

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

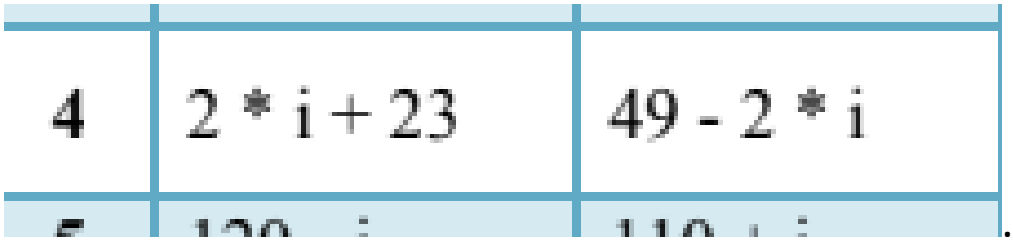
складання програмних специфікацій.

**Варіант**: 4
**Умова задачі**:

4. Обчислити спільний дільник для двох цілих десяткових чисел

**Постановка задачі.**

Згенерувати 2 масиви за заданими правилами для формування 1-го та 2-го масиву відповідно:




Знайти суму мінімального та максимального елементів масиву, утвореного рівними елементами перших двох масивів.

**Побудова математичної моделі:** для більшої наочності складемо таблицю імен змінних.

| Змінна | Тип | Ім’я | Призначення |
|---|---|---|---|
| Перший масив | Символьний | array1 | Початкові дані |
| Другий масив | Символьний | array2 | Початкові дані |

| | | | |
|---|---|---|---|
| Третій масив | Символьний | array3 | Проміжні дані. Глобальна змінна |
| Параметр арифм. циклів. Лічильник | Цілий | i | Проміжні дані. Параметр циклу |
| Параметр вкладених арифм. циклів. Лічильник | Цілий | j | Проміжні дані. Параметр циклу |
| Кількість елементів двох перших масивів | Цілий | a | Початкові дані. Константа. Глобальна змінна |
| Кількість елементів третього масиву | Цілий | n | Проміжні дані. Глобальна змінна |
| Результат: сума мін. і макс. елементів третього масиву | Цілий | result | Результуюча змінна |
| Масив | Символьний | array | Локальна змінна підпрограми. Аргумент функції. Проміжні дані |
| Мін. елемент | Символьний | min | Локальна змінна підпрограми. Проміжні дані |
| Макс. елемент | Символьний | max | Локальна змінна підпрограми. Проміжні дані |

Фактично задана умова зводиться до:

1) послідовного циклічного знаходження усіх *a(a=10* за умовою) елементів першого та другого масивів за заданими правилами:

$$array1[i] = 2 * i + 23$$

$$array2[i] = 49 - 2 * i$$

2) порівняння елементів першого та другого масивів один з одним. Формування нового масиву такого, що включає лише спільні елементи обох попередніх;

3) порівняння елементів третього масиву. Знаходження найбільшого та найменшого елементів.

Щодо порівняння елементів масивів, то, порівнюючи дані символьного типу, ми порівнюємо їх коди у таблиці ASCII.

Отже, математична модель достатньо проста, тож алгоритм програми буде побудований саме на ній з використанням циклів для послідовного визначення кожного з елементів масиву.

Щодо формування масиву спільних елементів та розрахунку мінімального/максимального елементів, то доречним буде використання функцій.

Варто також відмітити, що деякі змінні(кількості елементів масивів та третій масив) будуть глобальними задля швидкого звертання до них з будь-якої частини програми та більшої загальності.

**Розв’язання.**

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

   *Крок 1. Визначимо основні дії.*

   *Крок 2. Деталізуємо формування двох початкових масивів.*

   *Крок 3. Деталізуємо дію утворення масиву спільних елементів двох масивів у підпрограмі.*

   *Крок 4. Деталізуємо дію знаходженння максимальних та мінімальних елементів отриманого масиву.*

## *Псевдокод*

# Крок №1

# Підпрограма(1)
# array3_formation(array1, array2):

**початок**
**Утворення масиву спільних елементів двох масивів у підпрограмі**
**кінець**

# Підпрограма(2)
# minimum(char* array):

**початок**
**min=array[0]**
**Знаходженння мінімального елементу отриманого масиву**
**повернути min**
**кінець**

# Підпрограма(3)
# minimum(char* array):

**початок**
**max=array[0]**
**Знаходженння максимального елементу отриманого масиву**
**повернути max**
**кінець**

# Програма:

**початок**

**int n=0**

**const int a = 10**

**char array3[10]**

**char array1[a]**

**char array2[a]**

**формування першого масиву**

**Формування другого масиву**

**array3_formation(array1, array2)**

**int result = maximum(array3)-minimum(array3)**

**вивести result**

**кінець**

# Крок №2

## Підпрограма(1)
## array3_formation(array1, array2):
**початок**
**Утворення масиву спільних елементів двох масивів у підпрограмі**
**кінець**

## Підпрограма(2)
## minimum(char* array):
**початок**
**min=array[0]**
**Знаходженння мінімального елементу отриманого масиву**
**повернути min**
**кінець**

## Підпрограма(3)
## minimum(char* array):
**початок**
**max=array[0]**
**Знаходженння максимального елементу отриманого масиву**
**повернути max**
**кінець**

## Програма:

```
початок

int n=0

const int a = 10

char array3[10]

char array1[a]

char array2[a]

повторити

        для і від 0 до a-1

                array1[i]=2*i+23

все повторити
```

**повторити**

**для і від 0 до a-1**

**array2[i]=49-2*i**

**все повторити**

**array3_formation(array1, array2)**

**int result = maximum(array3)-minimum(array3)**

**вивести result**

**кінець**

# Крок №3

# Підпрограма(1) array3_formation(array1, array2):

**початок**
**повторити**
**для і від 0 до a-1**
**повторити**
**для j від 0 до a-1**
**якщо array1[j]==array2[i] то**
**array3[n]=array1[j]**
**n++**
**все якщо**
**все повторити**
**все повторити**
**кінець**

# Підпрограма(2) minimum(char* array):

**початок**
**min=array[0]**
**Знаходженння мінімального елементу отриманого масиву**
**повернути min**
**кінець**

# Підпрограма(3) minimum(char* array):

**початок**
**max=array[0]**
**Знаходженння максимального елементу отриманого масиву**
**повернути max**

**кінець**

# Програма:

```
початок
int n=0
const int a = 10
char array3[10]
char array1[a]
char array2[a]
повторити
        для і від 0 до a-1
                array1[i]=2*i+23
все повторити
повторити
        для і від 0 до a-1
                array2[i]=49-2*i
все повторити
array3_formation(array1, array2)
int result = maximum(array3)-minimum(array3)
вивести result
кінець
```

# Крок №4

## Підпрограма(1) array3_formation(array1, array2):

```
початок
повторити
        для і від 0 до a-1
                повторити
                        для j від 0 до a-1
                                якщо array1[j]==array2[i] то
                                        array3[n]=array1[j]
                                        n++
                                все якщо
                повторити
все повторити
кінець
```

## Підпрограма(2) minimum(char* array):

```
початок
min=array[0]
повторити
        для і від 0 до n-1
                якщо array[i]<min то
                        min = array3[i]
                все якщо
все повторити
повернути min
кінець
```

## Підпрограма(3) minimum(char* array):

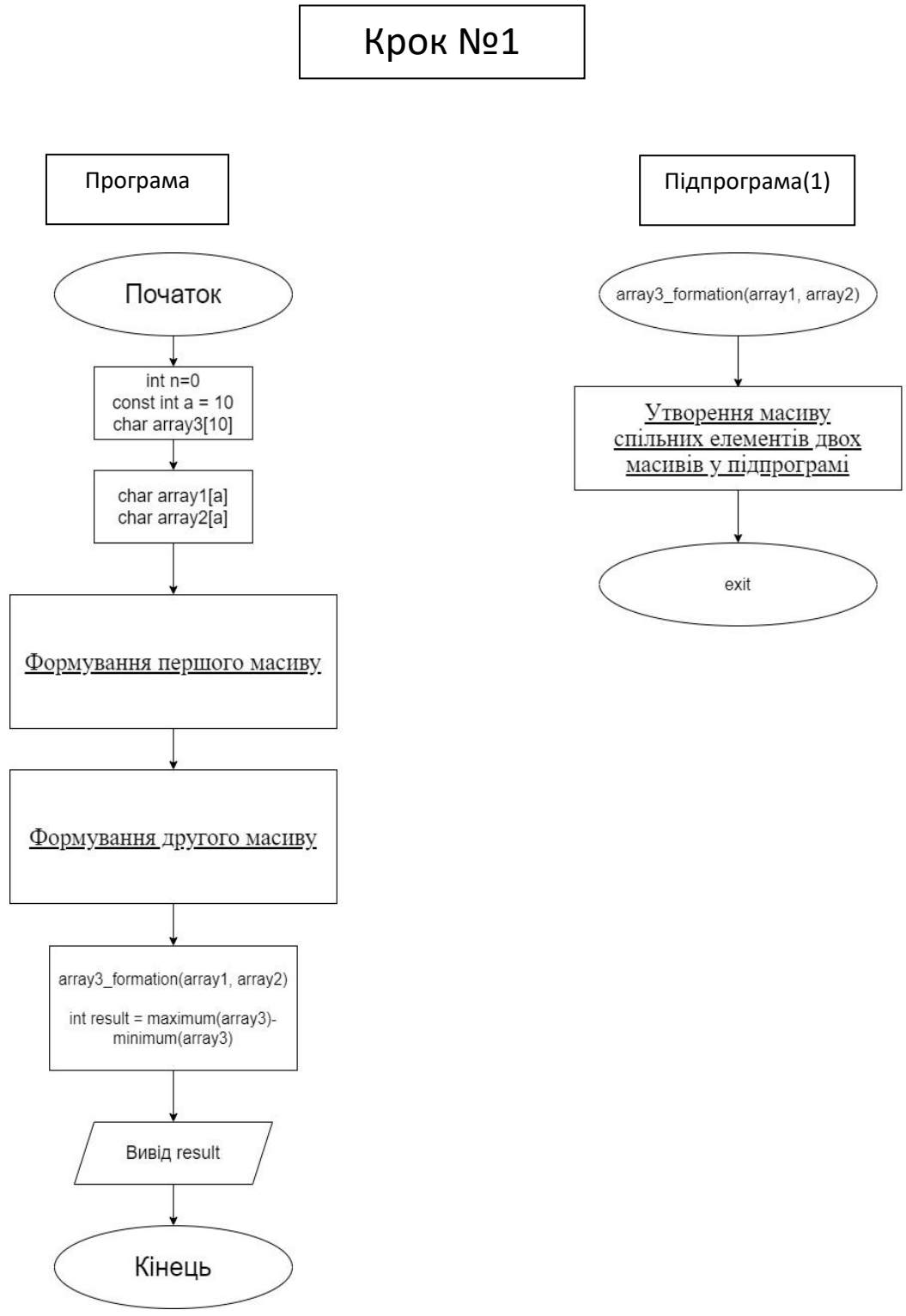
```
початок
max=array[0]
повторити
        для і від 0 до n-1
                якщо array[i]>max то
                        max = array3[i]
                все якщо
все повторити
повернути max
кінець
```

# Програма:

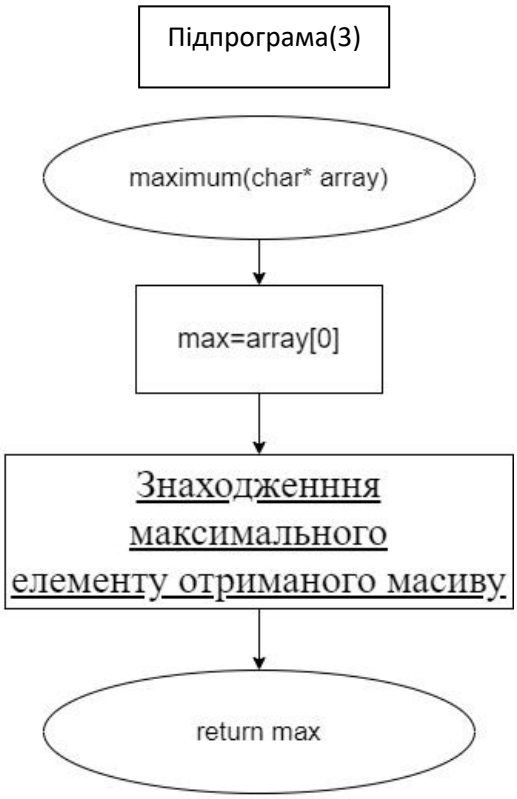
```
початок
int n=0
const int a = 10
char array3[10]
char array1[a]
char array2[a]
повторити
        для і від 0 до a-1
                array1[i]=2*i+23
все повторити
повторити
        для і від 0 до a-1
                array2[i]=49-2*i
все повторити
array3_formation(array1, array2)
int result = maximum(array3)-minimum(array3)
вивести result
кінець
```

Утворивши псевдокод, побудуємо блок-схему алгоритму.

## Блок-схема

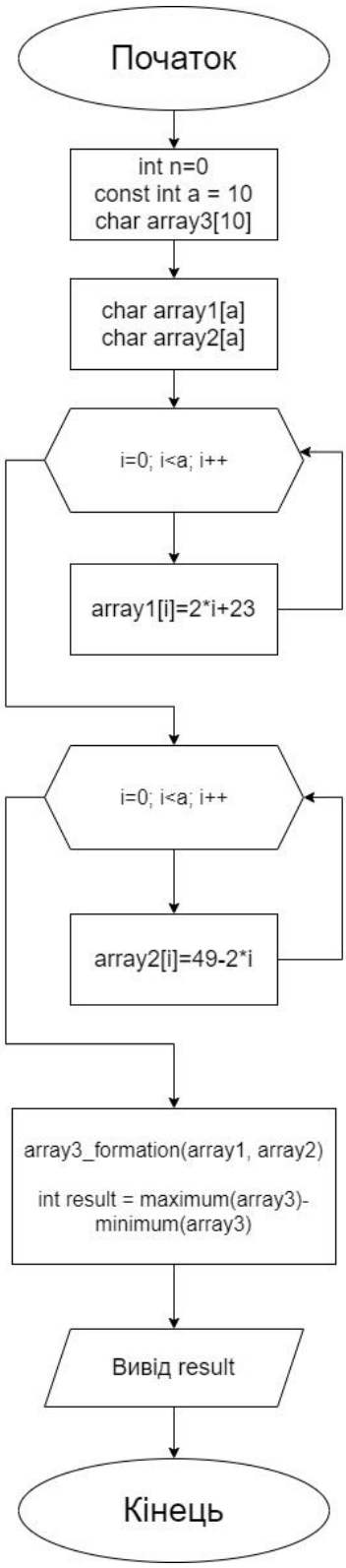



Підпрограма(2)
minimum(char* array)
min=array[0]
Знаходженння мінімального елементу отриманого масиву
return min

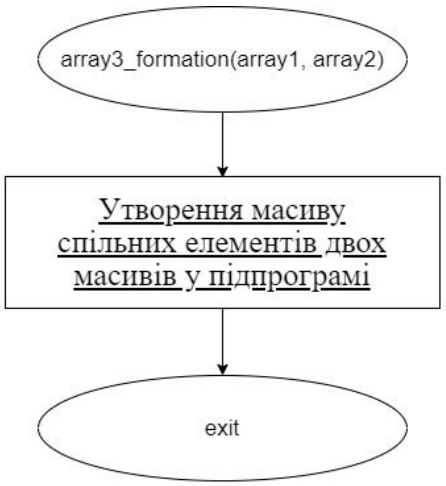

Підпрограма(3)
maximum(char* array)
max=array[0]
Знаходженння максимального елементу отриманого масиву
return max

# Крок №2

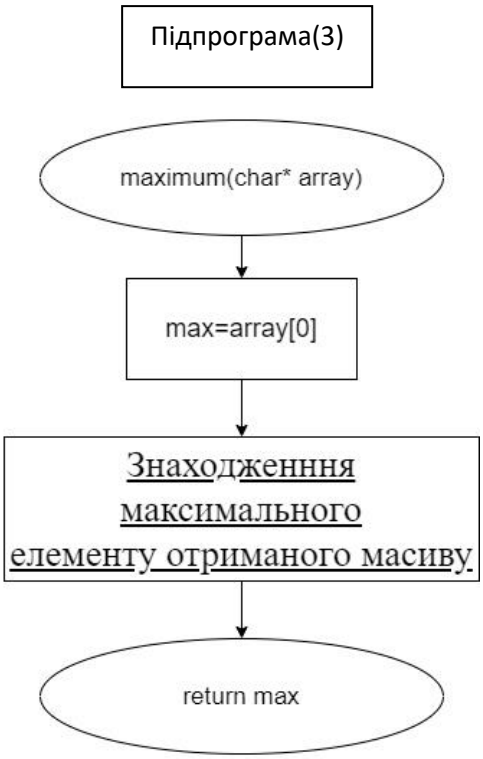
Програма



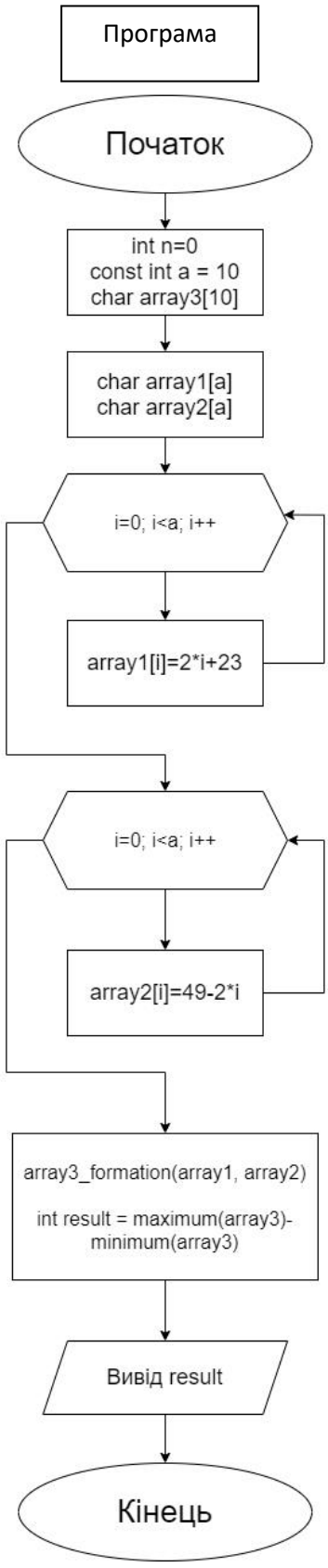

Підпрограма(1)

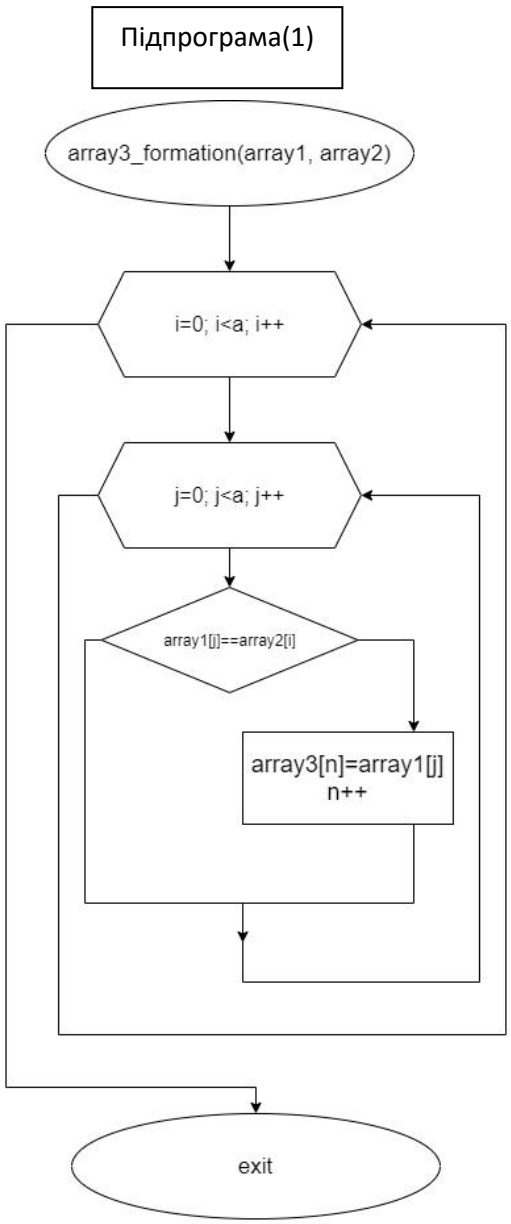



Підпрограма(2)
minimum(char* array)
min=array[0]
Знаходженння мінімального елементу отриманого масиву
return min

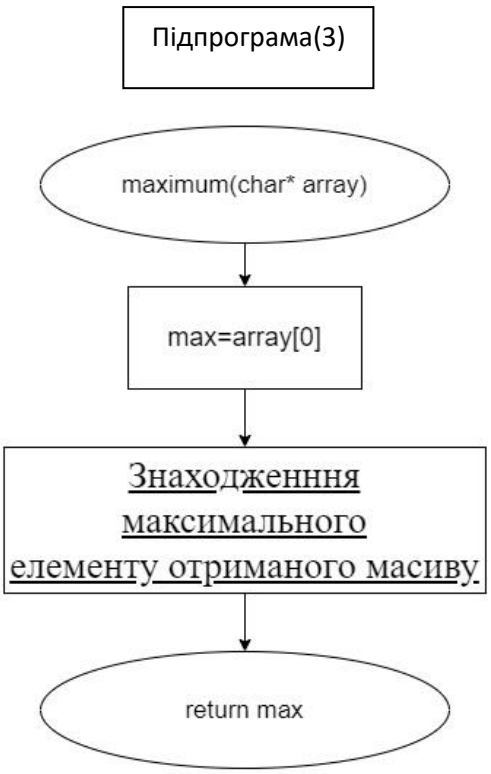

Підпрограма(3)
maximum(char* array)
max=array[0]
Знаходженння максимального елементу отриманого масиву
return max

# Крок №3

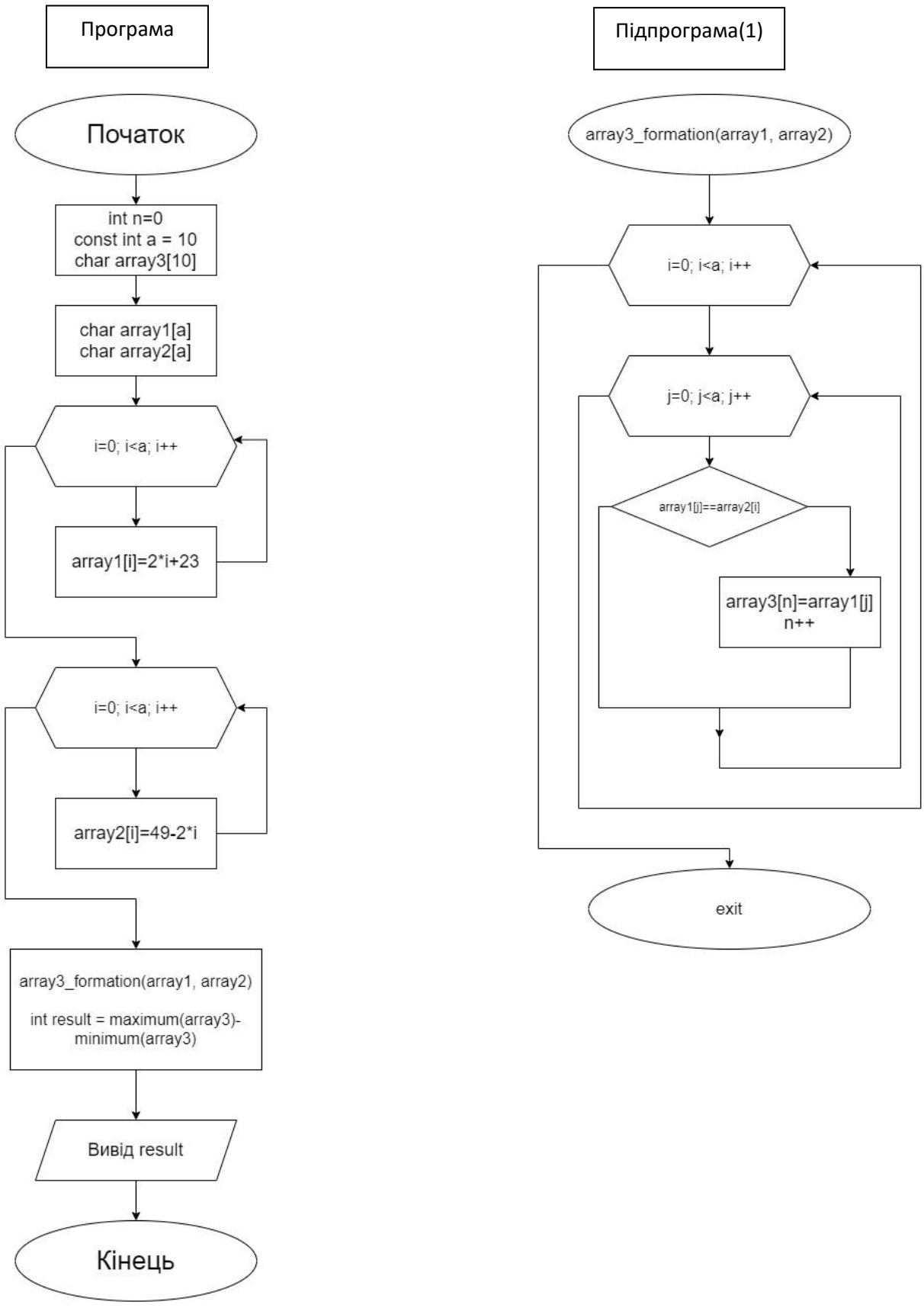

Програма
Початок
int n=0
const int a = 10
char array3[10]
char array1[a]
char array2[a]
i=0; i<a; i++
array1[i]=2*i+23
i=0; i<a; i++
array2[i]=49-2*i
array3_formation(array1, array2)
int result = maximum(array3)-
minimum(array3)
Вивід result
Кінець

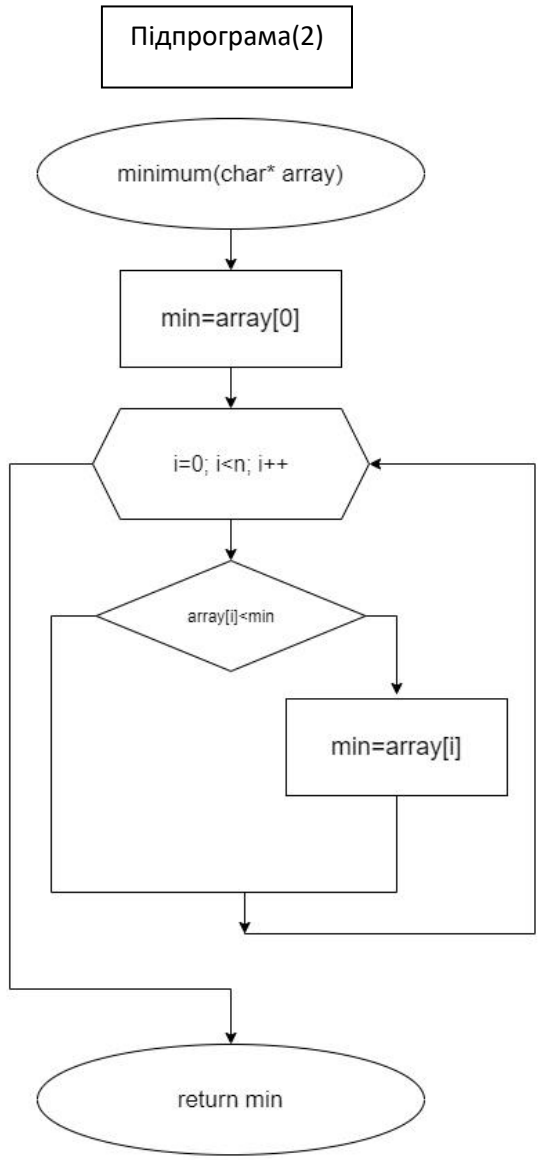

Підпрограма(1)
array3_formation(array1, array2)
i=0; i<a; i++
j=0; j<a; j++
array1[j]==array2[i]
array3[n]=array1[j]
n++
exit

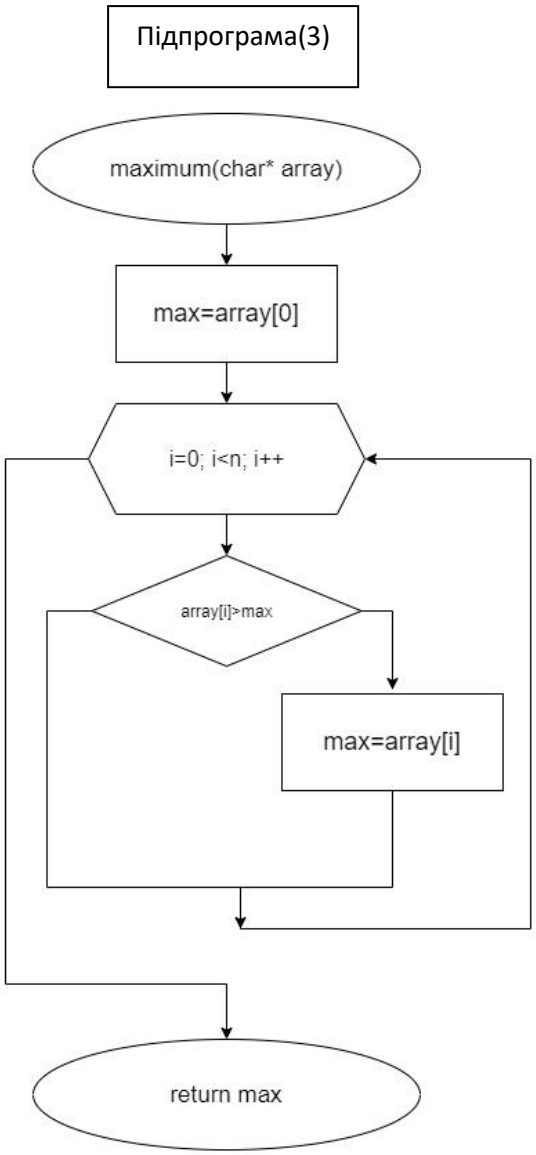

Підпрограма(2)
minimum(char* array)
min=array[0]
Знаходженння мінімального елементу отриманого масиву
return min


Підпрограма(3)
maximum(char* array)
max=array[0]
Знаходженння максимального елементу отриманого масиву
return max

# Крок №4


Програма
Початок
int n=0
const int a = 10
char array3[10]
char array1[a]
char array2[a]
i=0; i<a; i++
array1[i]=2*i+23
i=0; i<a; i++
array2[i]=49-2*i
array3_formation(array1, array2)
int result = maximum(array3)-minimum(array3)
Вивід result
Кінець
Підпрограма(1)
array3_formation(array1, array2)
i=0; i<a; i++
j=0; j<a; j++
array1[j]==array2[i]
array3[n]=array1[j]
n++
exit


Підпрограма(2)
minimum(char* array)
min=array[0]
i=0; i<n; i++
array[i]<min
min=array[i]
return min


Підпрограма(3)
maximum(char* array)
max=array[0]
i=0; i<n; i++
array[i]>max
max=array[i]
return max

## Випробування алгоритму

Перевіримо правильність алгоритму на довільних конкретних значеннях початкових даних.

### Тест №1(a=10)

| Блок | Дія |
|---|---|
| 1 | **Початок** |
| 2 | n = 0<br>a = 10<br>array3[10] |
| 3 | Арифметичний цикл. Заповнення першого масиву. array1[a]={23,25,27,29,31,33,35,37,39,41} |
| 4 | Арифметичний цикл. Заповнення другого масиву. array1[a]={49,47,45,43,41,39,37,35,33,31} |
| 5 | Ініціалізація підпрограми array3_formation(array1,array2). Арифметичний цикл з вкладенням. Формування третього масиву: array3[10]={41,39,37,35,33,31}. Кінець підпрограми |
| 6 | Ініціалізація підпрограми minimum(array3). Арифметичний цикл. Повернення min = 31 |
| 7 | Ініціалізація підпрограми maximum(array3). Арифметичний цикл. Повернення min = 41 |
| 8 | result = 41-31=10 |
| 9 | Виведення result |
| 10 | Кінець |

Більш детально можете ознайомитися з готовим кодом на GitHub

# Код програми:

```
//  Created by Берлинский Ярослав Владленович on 11.11.2020.


#include <iostream>
using namespace std;

int n=0;
const int a=10;
char array3[10];

void array3_formation(char[], char[]);
char maximum(char[]);
char minimum(char[]);

int main()
{
    char array1[a];
    char array2[a];
    for (int i=0; i<a; i++){
        array1[i]=2*i+23;
        printf("Елемент[%d] 1-го массиву: '%c' - код ASCII: %d\n", i, array1[i],
            int(array1[i]));
    }
    cout<<endl;
    for (int i=0; i<a; i++){
        array2[i]=49-2*i;
        printf("Елемент[%d] 2-го массиву: '%c' - код ASCII: %d\n", i, array2[i],
            int(array2[i]));
    }
    cout<<endl;

    array3_formation(array1, array2);
    cout<<endl<<endl;
    for (int i=0; i<n; i++){
        printf("Елемент[%d] 3-го массиву: '%c' - код ASCII: %d\n", i, array3[i],
            int(array3[i]));
    }
    cout<<endl;
    cout<<"Код максимального: "<<int(maximum(array3))<<endl;
    cout<<"Код мінімального: "<<int(minimum(array3))<<endl;
    int result = maximum(array3)-minimum(array3);
    printf("Різниця макс. і мін. елементів: %d(символ - %c)\n", result, result);
    return 0;
}

void array3_formation(char* array1, char* array2)
{
    for (int i=0; i<a; i++){
        for (int j=0; j<a;j++){
        if (int(array1[j])==int(array2[i])){
            array3[n]=array1[j];
            cout<<n<<" - "<<int(array3[n])<<endl;
            n++;
        }
        }
    }
}
char maximum(char* array){
   char max=array[0];
   for (int i=0; i<n; i++){
       if (array[i]>max){
           max=array[i];
       }
   }
    return max;
}

char minimum(char* array){
   char min=array[0];
   for (int i=0; i<n; i++){
       if (array[i]<min){
           min=array[i];
       }
   }
    return min;
}
```

**Вивід(a=10):**

```
Елемент[4] 1-го массиву: '' - код ASCII: 31
Елемент[5] 1-го массиву: '!' - код ASCII: 33
Елемент[6] 1-го массиву: '#' - код ASCII: 35
Елемент[7] 1-го массиву: '%' - код ASCII: 37
Елемент[8] 1-го массиву: ''' - код ASCII: 39
Елемент[9] 1-го массиву: ')' - код ASCII: 41

Елемент[0] 2-го массиву: '1' - код ASCII: 49
Елемент[1] 2-го массиву: '/' - код ASCII: 47
Елемент[2] 2-го массиву: '-' - код ASCII: 45
Елемент[3] 2-го массиву: '+' - код ASCII: 43
Елемент[4] 2-го массиву: ')' - код ASCII: 41
Елемент[5] 2-го массиву: ''' - код ASCII: 39
Елемент[6] 2-го массиву: '%' - код ASCII: 37
Елемент[7] 2-го массиву: '#' - код ASCII: 35
Елемент[8] 2-го массиву: '!' - код ASCII: 33
Елемент[9] 2-го массиву: '' - код ASCII: 31



Елемент[0] 3-го массиву: ')' - код ASCII: 41
Елемент[1] 3-го массиву: ''' - код ASCII: 39
Елемент[2] 3-го массиву: '%' - код ASCII: 37
Елемент[3] 3-го массиву: '#' - код ASCII: 35
Елемент[4] 3-го массиву: '!' - код ASCII: 33
Елемент[5] 3-го массиву: '' - код ASCII: 31

Код максимального: 41
Код мінімального: 31
Різниця макс. і мін. елементів: 10(символ -
)
```

**Висновок:** отже, за допомогою масивів було знайдено набори рядів чисел, заданих правилами, порівняно їх між масивами та сформовано масив спільних елементів, з якого знайдено мінімальний та максимальний елементи.

На перший погляд, достатньо громіздкий для повного розуміння алгоритм був на простому рівні представлений за допомогою використання підпрограм, а саме: однієї процедури(утворення нового масиву спільних елементів двох даних масивів) та двох функцій(пошук мінімального та максимального елементів).

По-перше, це узагальнює рішення поставленої задачі для ряду випадків.

По-друге, код тіла основної функції полегшується для сприйняття.