Додаток 1

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 15

Виконав студент ІП-12, Кириченко Владислав Сергійович
(шифр, прізвище, ім'я, по батькові)

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

Київ 2021

Лабораторна робота № 7

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 15

Умова задачі:

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 (**43 - i; 37 + i**).
3. Ініціювання третьої змінної рівними значеннями двох попередніх змінних.
4. Обробки третьої змінної згідно з варіантом(**Добуток елементів, коди яких більше 40**).

Постановка задачі:

Початкові дані **-** із початкових даних маємо лише кількість елементов масиву.

За даними вимогами до формування елементів сгенерувати 3 массиви. Перші два масива - сгенерувати за даними формулами, а третій - з рівних елементів перших двох массивів.Потів знайти добуток усіх елеменіт третього масива, коди елементів яких більше за 40.

Результат - число, що дорівнює добутку всіх елеметів третього масива, коди елементів яких більше за 40.

Побудова математичної моделі:

Складемо таблицю змінних:

Змінна	Тип	Ім’я	Призначення
Розмір масивів	цілочисельний	***size***	Початкове значення
Масив з формулою елемента **43 - i**	індексований	***arrA***	Проміжкове значення
Масив з формулою елемента **37 + i**	індексований	***arrB***	Проміжкове значення
Масив рівних елементів перший двух масивів	індексований	***arrC***	Проміжкове значення
Лічильник	цілочисельний	***i***	Проміжкове значення
Лічильник	цілочисельний	***j***	Проміжкове значення

формальний параметр(перший масив)	індексований	***arr1***	Проміжкове значення
формальний параметр(другий масив)	індексований	***arr2***	Проміжкове значення
формальний параметр(третій масив)	індексований	***newArr***	Проміжкове значення
результат роботи підпрогрмамми	цілочисельний	***prod***	Результат(підпрограмма)
результат роботи прогрмамми	цілочисельний	***product***	Результат

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

Крок 1. Визначимо основні дії.
Крок 2.Деталізація ініціалізації промжкових змінних ***size***, ***arrA, arrB,arrC***
Крок 3.Деталізація заповненя масивів ***arrA, arrB*** елементами
Крок 4.Деталізація заповнення масива ***arrC*** рівними елементами масивів ***arrA, arrB***
Крок 5.Деталізація знаходження значення змінної ***product***

Псевдокод(основна програма):

Крок 1.

початок
введення
ініціалізація промжкових змінних ***size***, ***arrA, arrB,arrC***
заповненя масивів ***arrA, arrB*** елементами
заповнення масива ***arrC*** рівними елементами масивів ***arrA, arrB***
знаходження значення змінної ***product***
виведення ***product***
кінець

Крок 2.

початок
введення
size=10
arrA[size]
arrB[size]
arrC[size]

заповненя масивів ***arrA, arrB*** елементами
заповнення масива ***arrC*** рівними елементами масивів ***arrA, arrB***
знаходження значення змінної ***product***
виведення ***product***
кінець

Крок 3.

початок
введення
size=10
arrA[size]
arrB[size]
arrC[size]
fillArrays(arrA, arrB)
заповнення масива ***arrC*** рівними елементами масивів ***arrA, arrB***
знаходження значення змінної ***product***
виведення ***product***
кінець

Крок 4.

початок
введення
size=10
arrA[size]
arrB[size]
arrC[size]
fillArrays(arrA, arrB)
generateArr(arrA, arrB, arrC)
знаходження значення змінної ***product***
виведення ***product***
кінець

Крок 5.

початок
введення
size=10
arrA[size]
arrB[size]
arrC[size]
fillArrays(arrA, arrB)
generateArr(arrA, arrB, arrC)
product = calclulateProduct(arrC)
виведення ***product***
кінець

Псевдокод(підпрогрма - fillArrays):

Крок 1.

функція fillArrays(arr1[], arr2[])
заповнення порожніх масивів елементами
кінець

Крок 2.

функція fillArrays(arr1[], arr2[])
повторити
для *i* від 0 **до *size* із кроком** 1
arr1[i] = 43 - ***i***
arr2[i] = 37 + ***i***
все повторити
кінець

Псевдокод(підпрогрма - generateArr):

Крок 1.
функція *generateArr(arr1, arr2,newArr,size)*
заповнення масива ***arrC*** рівними елементами масивів ***arrA, arrB***
кінець

Крок 2.
функція *generateArr(arr1, arr2,newArr,size)*
повторити
для *i* від 0 **до *size* із кроком** 1

```
            newArr[i] = 0
            j = 0
            повторити
                    поки j < size та newArr[i] == 0
                    якщо arr1[i] == arr2[j]
                            то
                            newArr[i] = arr1[i]
                    все якщо
                    j ++
            все повторити
   все повторити
кінець
```

Псевдокод(підпрогрма - calclulateProduct):
Крок 1.

функція *calclulateProduct* ***(arr3[],size)***
   обрахування значення змінної **prod**
   перевірка чи були знайдені елементи, що задовольняють умову задачі
   повернути **prod**
кінець

Крок 2.

функція *calclulateProduct* ***(newArr[],size)***
   prod = 1
   повторити
            для ***i*** **від** 0 **до** ***size*** **із кроком** 1
            якщо ***newArr[i]*** > 40
                    то
                    prod *= newArr[i]
            все якщо
   все повторити
   перевірка чи були знайдені елементи, що задовольняють умову задачі
   повернути **prod**
кінець

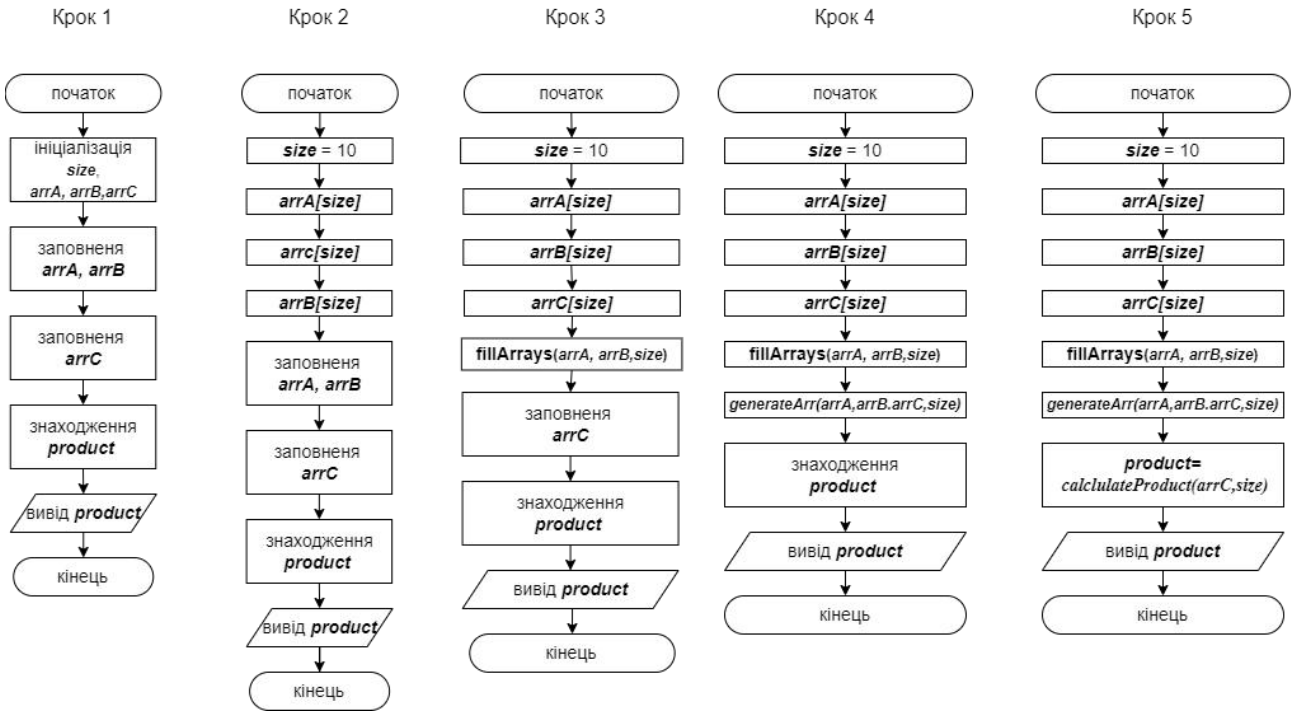
Крок 3.

функція *calclulateProduct* ***(newArr[],size)***
   prod = 1
   повторити
            для ***i*** **від** 0 **до** ***size*** **із кроком** 1
            якщо ***newArr[i]*** > 40
                    то
                    prod *= newArr[i]
            все якщо
   все повторити
   якщо prod == 1
            то
              prod =0

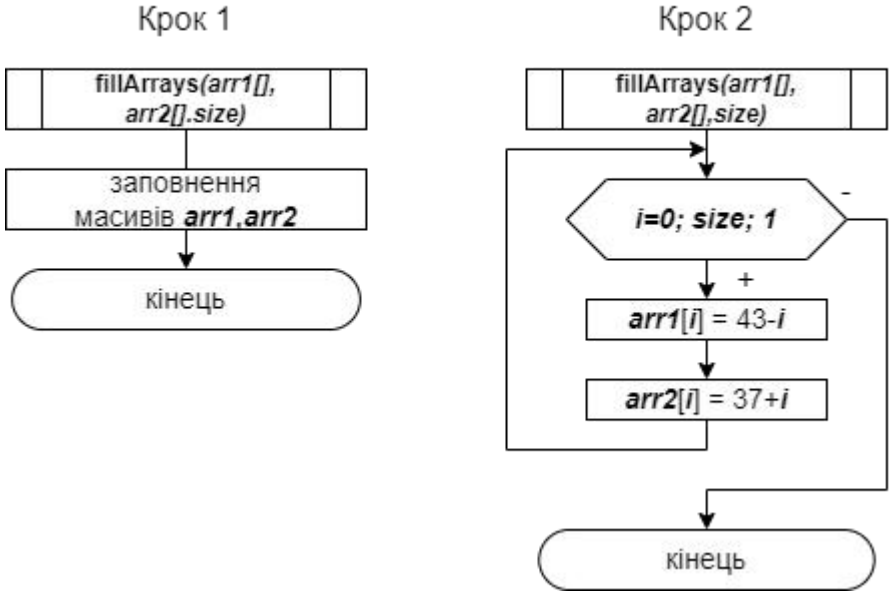
все якщо
повернути **prod**
кінець

Блок схема:

Основна програма



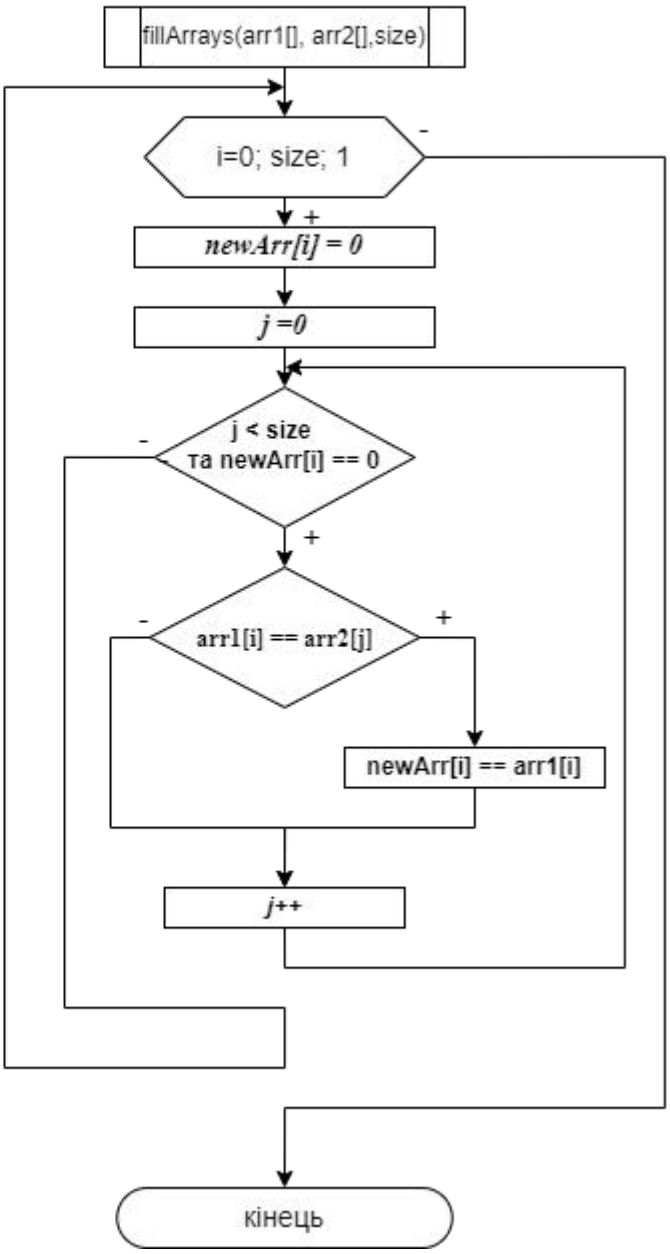
*Підпрограма(*fillArrays*)*



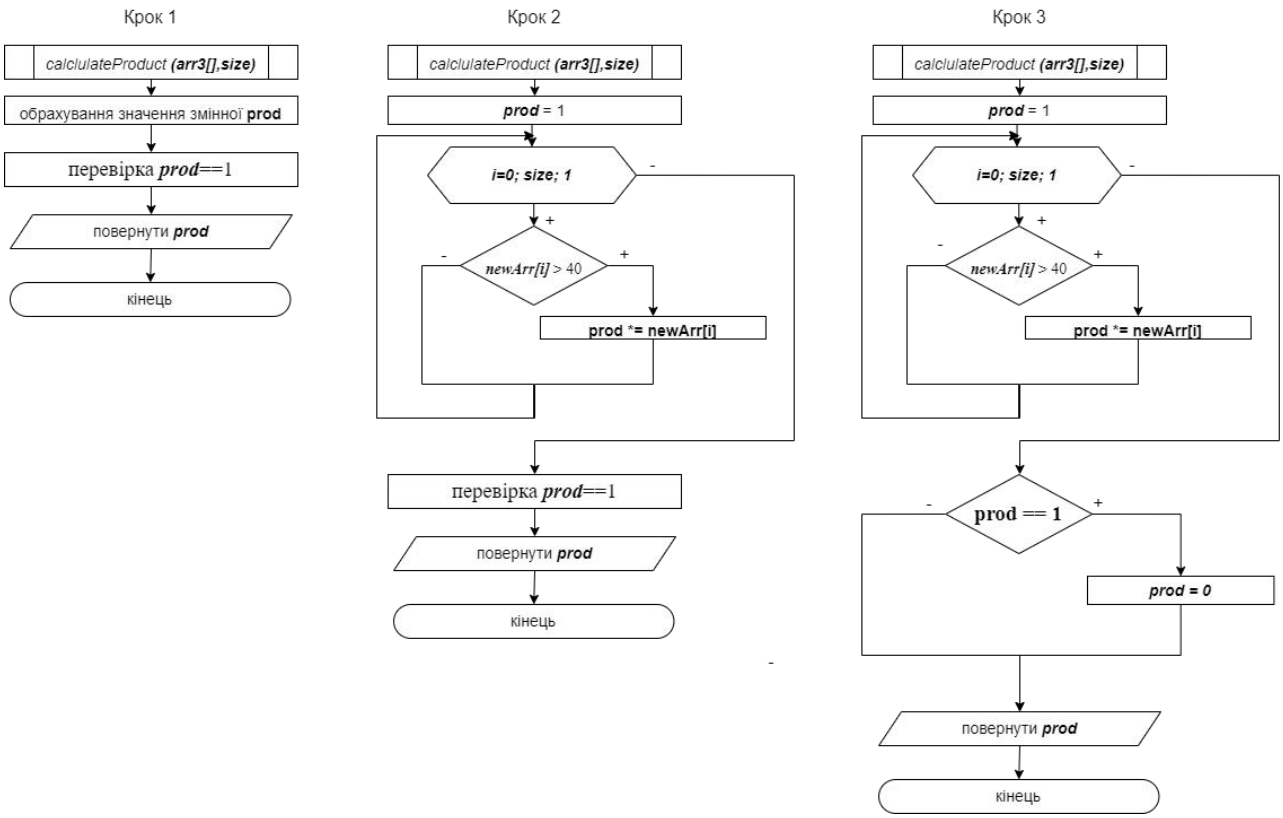
*Підпрограма(*generateArr*)*

Крок 1

Крок 2

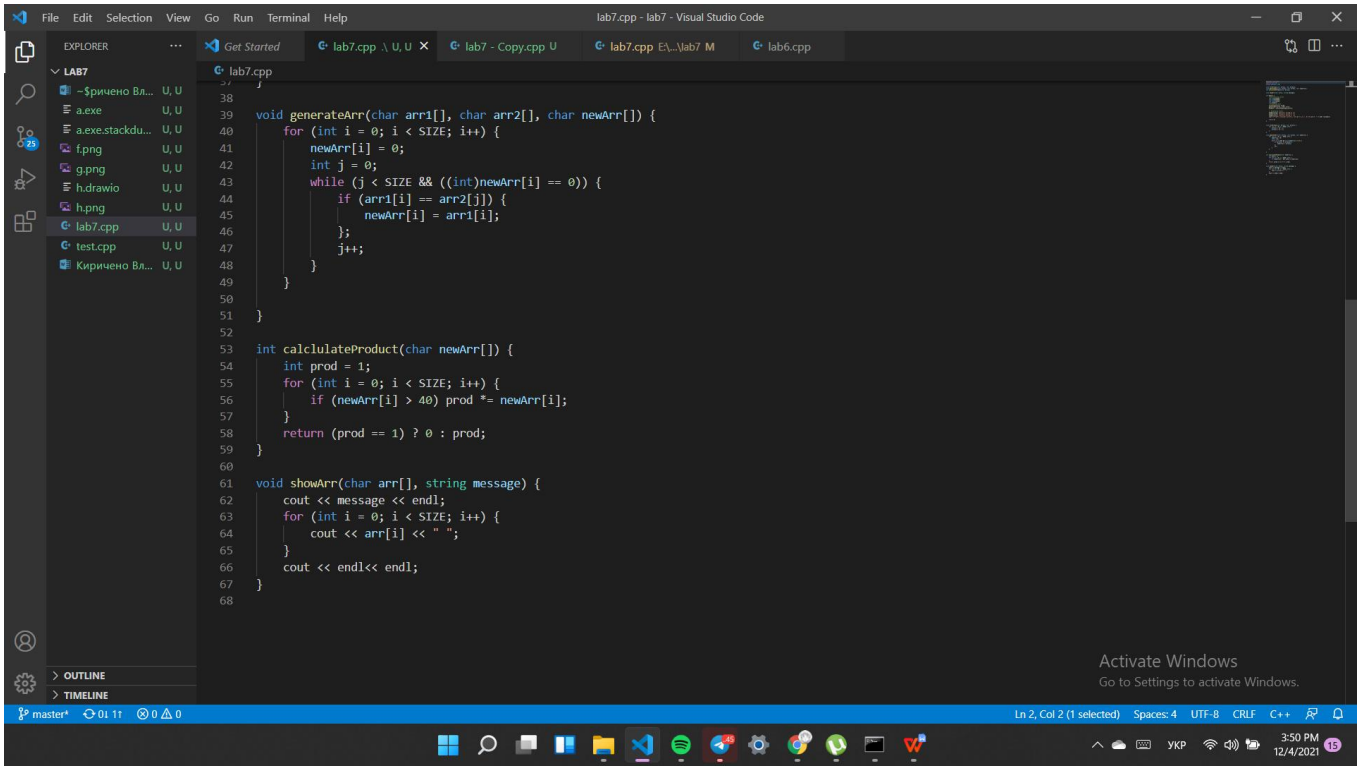


Підпрограма(calclulateProduct)

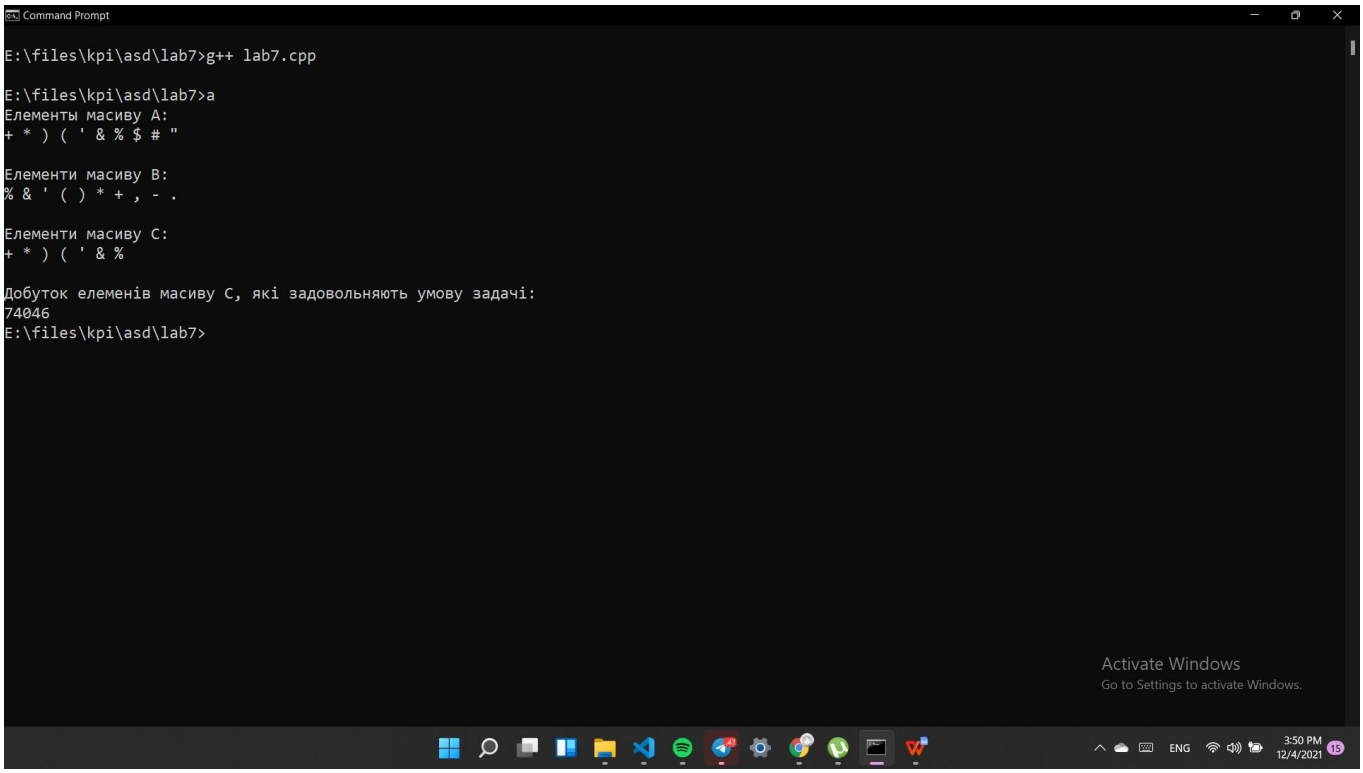


4. Код програми(C++)

```cpp
#include <iostream>
#define SIZE 10
using namespace std;

void fillArrays(char arr1[], char arr2[]);
void generateArr(char arr1[], char arr2[], char newArr[]);
int calclulateProduct(char arr[]);

void showArr(char arr[], string message);

int main() {
    //Initializing values
    char arrA[SIZE];
    char arrB[SIZE];
    char arrC[SIZE];
    int product;
    //Processing data
    fillArrays(arrA, arrB);
    generateArr(arrA, arrB, arrC);
    product = calclulateProduct(arrC);

    // Displaying result
    showArr(arrA, "Елементы масиву A: ");
    showArr(arrB, "Елементи масиву B: ");
    showArr(arrC, "Елементи масиву C: ");
    cout <<"Добуток елеменів масиву C, які задовольняють умову задачі: " << endl <<product;

    return 0;
}


void fillArrays(char arr1[], char arr2[]) {
    for (int i = 0; i < SIZE; i++) {
        arr1[i] = 43 - i;
        arr2[i] = 37 + i;
    }
}
```




```
void generateArr(char arr1[], char arr2[], char newArr[]) {
    for (int i = 0; i < SIZE; i++) {
        newArr[i] = 0;
        int j = 0;
        while (j < SIZE && ((int)newArr[i] == 0)) {
            if (arr1[i] == arr2[j]) {
                newArr[i] = arr1[i];
            };
            j++;
        }
    }

}

int calclulateProduct(char newArr[]) {
    int prod = 1;
    for (int i = 0; i < SIZE; i++) {
        if (newArr[i] > 40) prod *= newArr[i];
    }
    return (prod == 1) ? 0 : prod;
}

void showArr(char arr[], string message) {
    cout << message << endl;
    for (int i = 0; i < SIZE; i++) {
        cout << arr[i] << " ";
    }
    cout << endl<< endl;
}
```



Висновок - Було досліджено методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набуто практичних навичок їх використання під час складання програмних специфікацій.

Декомпозовано задачу на 3 етапи:

1. Генерація перших двох масивів.
2. Генерація третього масиву з рівних елементами перших двух
3. Знаходження добутку усіх кодів елементів третього масива, що більше за 40.