

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 ІІ-12, Кириченко Владислав Сергійович
(шифр, прізвище, ім'я, по батькові)

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

Лабораторна робота № 7

Назва роботи: 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 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 елементів масиву.

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

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

Побудова математичної моделі:

Складемо таблицю змінних:

Змінна	Тип	Ім'я	Призначення
Розмір масивів	цілочисельний	<i>size</i>	Початкове значення
Масив з формулою елемента 43 - i	індексований	<i>arrA</i>	Проміжкове значення
Масив з формулою елемента 37 + i	індексований	<i>arrB</i>	Проміжкове значення
Масив рівних елементів перший двох масивів	індексований	<i>arrC</i>	Проміжкове значення
Лічильник	цілочисельний	<i>i</i>	Проміжкове значення
Лічильник	цілочисельний	<i>j</i>	Проміжкове значення

формальний параметр(перший масив)	індексований	<i>arr1</i>	Проміжкове значення
формальний параметр(другий масив)	індексований	<i>arr2</i>	Проміжкове значення
формальний параметр(третій масив)	індексований	<i>newArr</i>	Проміжкове значення
результат роботи підпрограми	цілочисельний	<i>prod</i>	Результат(підпрограма)
результат роботи програми	цілочисельний	<i>product</i>	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 заповнення 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*

заповнення 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]

заповнення 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 = calculateProduct(arrC)

виведення *product*

кінець

Псевдокод(підпрограма - *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(підпрограма - *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(підпрограма - *calculuteProduct*):

Крок 1.

функція *calculuteProduct* (*arr3[]*,*size*)

обрахування значення змінної **prod**

перевірка чи були знайдені елементи, що задовольняють умову задачі
повернути **prod**

кінець

Крок 2.

функція *calculuteProduct* (*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 *calculuteProduct* (*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

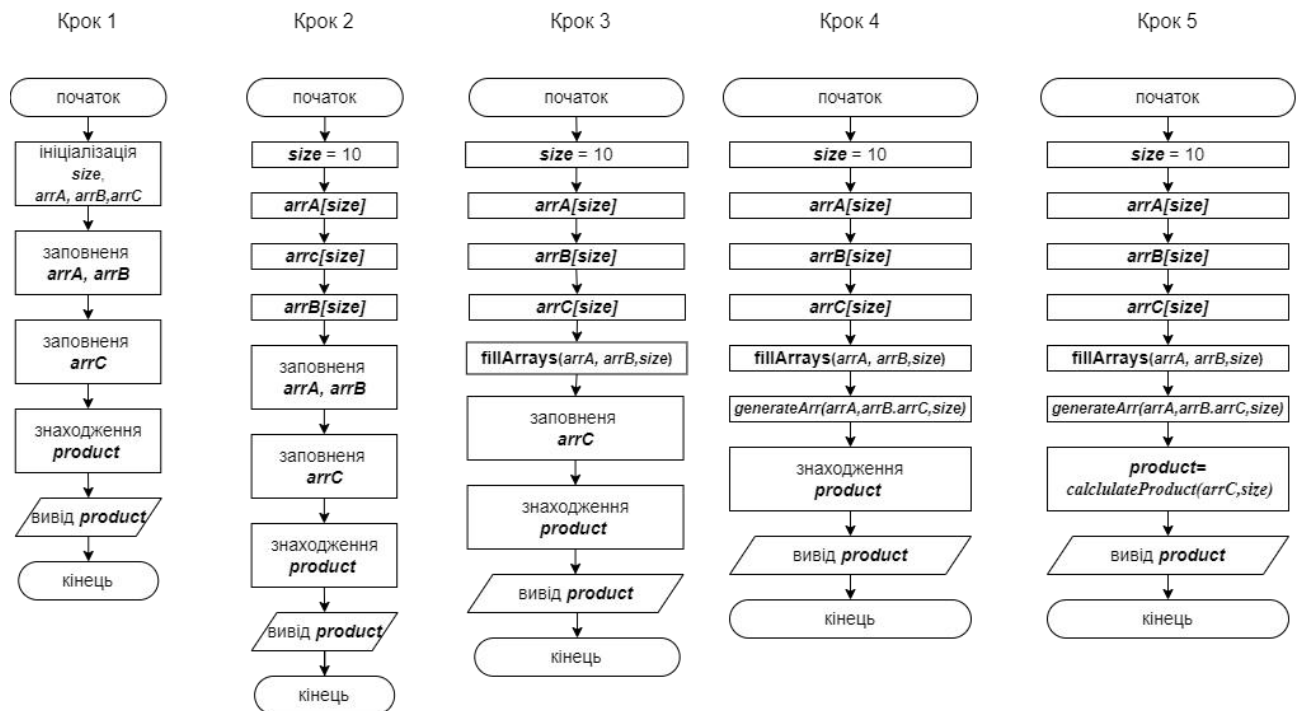
все якщо

поверненути **prod**

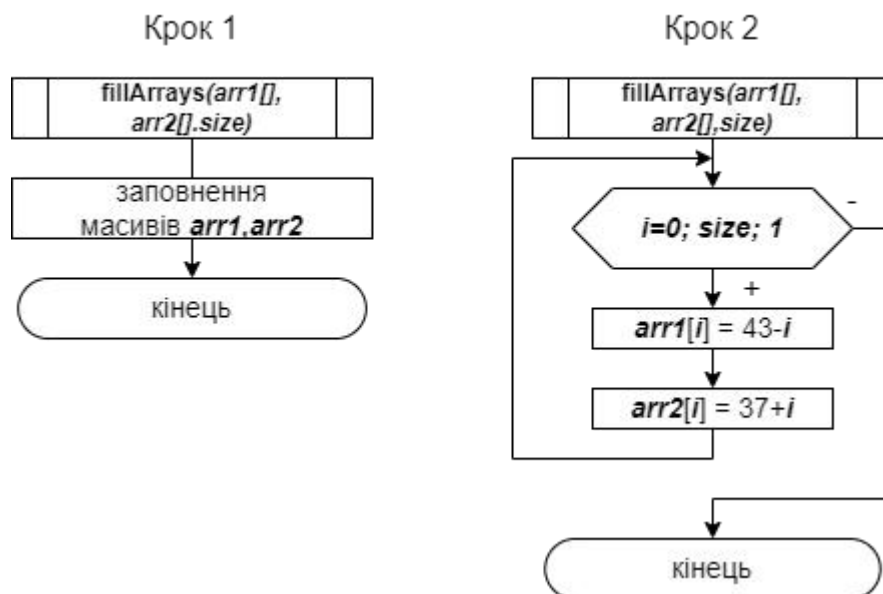
кінець

Блок схема:

Основна програма



Підпрограма(fillArrays)

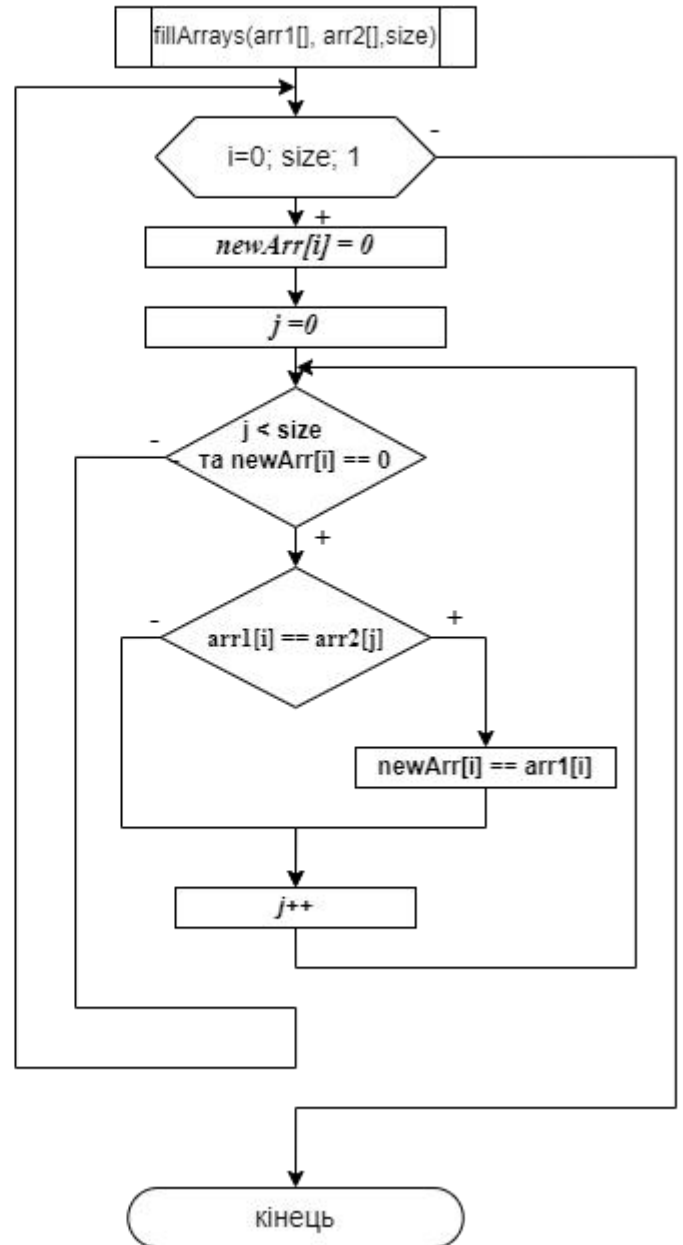


Підпрограма(*generateArr*)

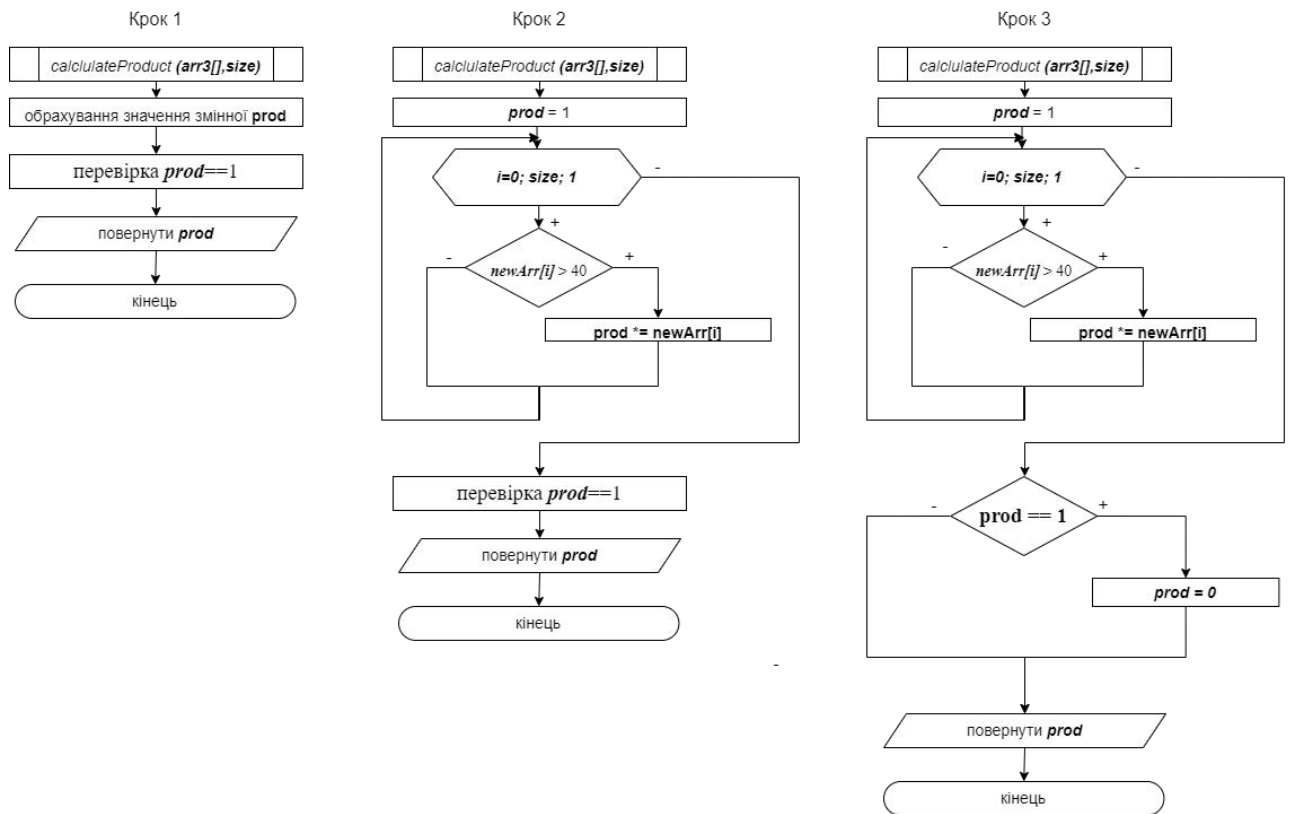
Крок 1



Крок 2



Підпрограма(*calculateProduct*)



4. Код програми(C++)

```

1 #include <iostream>
2 #define SIZE 10
3 using namespace std;
4
5 void fillArrays(char arr1[], char arr2[]);
6 void generateArr(char arr1[], char arr2[], char newArr[]);
7 int calculateProduct(char arr[]);
8
9 void showArr(char arr[], string message);
10
11 int main() {
12     //Initializing values
13     char arrA[SIZE];
14     char arrB[SIZE];
15     char arrC[SIZE];
16     int product;
17     //Processing data
18     fillArrays(arrA, arrB);
19     generateArr(arrA, arrB, arrC);
20     product = calculateProduct(arrC);
21
22     // Displaying result
23     showArr(arrA, "Елементи масиву А: ");
24     showArr(arrB, "Елементи масиву В: ");
25     showArr(arrC, "Елементи масиву С: ");
26     cout << "Добуток елементів масиву С, які задовольняють умову задачі: " << endl << product;
27
28     return 0;
29 }
30
31 void fillArrays(char arr1[], char arr2[]) {
32     for (int i = 0; i < SIZE; i++) {
33         arr1[i] = 43 - i;
34         arr2[i] = 37 + i;
35     }
36 }
37
  
```

```
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 calculateProduct(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;
}
```

```
E:\files\kpi\asd\lab7>g++ lab7.cpp

E:\files\kpi\asd\lab7>a
Элементы массива A:
+ * ) ( ' & % $ # "

Элементы массива B:
% & ' ( ) * + , - .

Элементы массива C:
+ * ) ( ' & %

Добуток елементів масиву C, які задовольняють умову задачі:
74046
E:\files\kpi\asd\lab7>
```

Висновок - Було досліджено методи послідовного пошуку у впорядкованих і неупорядкованих 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 двох
3. Знаходження добутку усіх кодів елементів третього масива, що більше за 40.