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 № 1 з дисципліни
«Алгоритми та структури даних-1.
Основи алгоритмізації»

«Дослідження лінійних алгоритмів»

Варіант 12

Виконав студент Дойчев Костянтин Миколайович
(шифр, прізвище, ім'я, по батькові)

Перевірила Вєчерковська Анастасія Сергіївна
(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота №7

Тема: 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 12

1) Постановка 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 (табл. 1).

3. Ініціювання третьої змінної рівними значеннями двох попередніх змінних.

4. Обробки третьої змінної згідно з варіантом.

Вираз для обчислення 1 масиву: 95 + i

Вираз для обчислення 2 масиву: 105 - i

Знайти: суму елементів, коди яких більше 101 (в 3 масиві)

Розв'язання

Крок 1. Визначимо основні дії.

Крок 2. Створимо функції для перебору масиву

Крок 3. Визначимо перший масив

Крок 4. Визначимо другий масив

Крок 5. Визначимо третій масив

Крок 6. Отримаємо суму елементів, коди яких більше 101

2) Побудова математичної моделі:

Таблиця імен змінних

Змінна	Тип	Ім’я	Призначення
Константа, яка визначає довжину масивів	Цілий	ARRAY_LENGTH	Проміжні дані
Константа, яка визначає мінімальний код елемента для обчислення суми	Цілий	MIN_COMMON_VALUE	Проміжні дані
Перший масив	Символ	array1	Проміжні дані
Другий масив	Символ	array2	Проміжні дані
Третій масив	Символ	array3	Проміжні дані
Покажчик(лічильник) заповнення третього масиву	Цілий	filledIndex	Проміжні дані
Відфільтрований масив з кодами більше 101	Символ	filteredArray	Проміжні дані

Новий масив, який поверне ф-ція filter	Символ	newArray	Проміжні дані
Сума елементів відфільтрованого масиву	Цілий	sum	Вихідні дані
Функція перебору масиву	void	forEach(array, callback)	Проміжні дані
Функція фільтрації масиву	Символ	filter(array, callback)	Проміжні дані
Функція (колбек) заповнення першого масиву	Цілий	calcFirstArray(item, index)	Проміжні дані
Функція (колбек) заповнення другого масиву	Цілий	calcSecondArray(item, index)	Проміжні дані
Функція (колбек) фільтрації третього масиву	Булевий	filterThirdArray(item, index)	Проміжні дані

Таким чином, математичне формулювання задачі зводиться до знаходження елементів масивів за заданими формулами. Та вирахування суми елементів відфільтрованого масиву (**filteredArray**), що матиме спільні елементи 1 та 2 масивів, які *строго* більше 101(**MIN_COMMON_VALUE**). Для цього варто ввести певні функції для перебору масивів, ці ф-ції(**filter**, **forEach**), будуть приймати в себе масив(**array**) та ф-цію колбек(**callback**), яку викликатимуть на кожній ітерації і передаватимуть в неї значення конкретного елемента(**item**) та його індекс(**index**). Це потрібно для того, аби універсалізувати ці функції під різні маніпуляції з масивами. Різниця між **filter** та **forEach** в тому, що **forEach** змінює вхідний масив, а **filter** повертає новий, відфільтрований масив(**newArray**), не змінюючи вхідний(**array**).

Також вартий уваги той факт, що в новий масив(**newArray**) в ф-ції **filter** запишеться значення тільки якщо колбек ф-ція(**callback**) повернула true. Самі колбек функції (**calcFirstArray**, **calcSecondArray**, **filterThirdArray**) виконуватимуть більш специфічні операції щодо кожного масиву.

Для обрахування елементів 1 і 2 масивів (**array1, array2**) знадобляться вище зазначені формули, а саме (95 + i), та (105 - i) відповідно, де i - індекс елементу масиву.

Отже, після обрахування 1 та 2 масивів(**array1, array2**) буде виконаний перебір елементів першого(**array1**) та другого(**array2**) масивів задля знаходження однакових ASCII кодів, при умові, що елементи однакові, один екземпляр буде додано до 3 масиву(**array3**). Також ми будемо користуватися змінною **filledIndex** задля того, щоб елементи 3 масиву були розташовані підряд.

Після цього кроку ми викличимо ф-цію **filter,** в яку передамо 3 масив та функцію фільтрації. Результат виконанная ф-ції **filter** запишемо у змінну **filteredArray,** суму елементів якої ми запишемо у змінну **sum** і виведемо її в консоль

3) Псевдокод алгоритму

Крок 1:

Підпрограма
    forEach(array, callback)
        для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**
            array[i] = callback(array[i], i);
        все повторити
Все підпрограма

Підпрограма
    filter(array, callback)
        newArray[10] = {};

```
            filledIndex = 0
            для i від 0 до ARRAY_LENGTH з кроком 1 повторити
                res = callback(array[i], i);
                якщо res == true
                    то
                        newArray[filledIndex] = array[i]
                        filledIndex++;
                все якщо
            все повторити
        повернути newArray
Все підпрограма

Підпрограма
        calcFirstArray(item, index)
            повернути 95 + index;
Все підпрограма

Підпрограма
        calcSecondArray(item, index)
            повернути 105 - index;
Все підпрограма

Підпрограма
        filterThirdArray(item, index)
            повернути item > MIN_COMMON_VALUE;
Все підпрограма

Початок
        Ініціалізація змінних
        Визначення першого масиву
        Визначення другого масиву
        Визначення третього масиву
        Фільтрація третього масиву
        Обчислення суми відфільтрованого масиву
        Виведення sum
Кінець
```

Крок 2:

```
Підпрограма
    forEach(array, callback)
        для i від 0 до ARRAY_LENGTH з кроком 1 повторити
            array[i] = callback(array[i], i);
        все повторити
Все підпрограма

Підпрограма
    filter(array, callback)
        newArray[10] = {};
        filledIndex = 0
        для i від 0 до ARRAY_LENGTH з кроком 1 повторити
            res = callback(array[i], i);
            якщо res == true
                то
                    newArray[filledIndex] = array[i]
                    filledIndex++;
            все якщо
        все повторити
    повернути newArray
Все підпрограма

Підпрограма
    calcFirstArray(item, index)
        повернути 95 + index;
Все підпрограма

Підпрограма
    calcSecondArray(item, index)
        повернути 105 - index;
Все підпрограма

Підпрограма
    filterThirdArray(item, index)
```

повернути item > MIN_COMMON_VALUE;

Все підпрограма

Початок

Ініціалізація змінних

forEach(array1, calcFirstArray)

Визначення другого масиву

Визначення третього масиву

Фільтрація третього масиву

Обчислення суми відфільтрованого масиву

Виведення sum

Кінець

Крок 3:

Підпрограма

forEach(array, callback)

для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**

array[i] = callback(array[i], i);

все повторити

Все підпрограма

Підпрограма

filter(array, callback)

newArray[10] = {};

filledIndex = 0

для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**

res = callback(array[i], i);

якщо res == true

то

newArray[filledIndex] = array[i]

filledIndex++;

все якщо

все повторити

повернути newArray

Все підпрограма

Підпрограма

calcFirstArray(item, index)
повернути 95 + index;
Все підпрограма

Підпрограма
calcSecondArray(item, index)
повернути 105 - index;
Все підпрограма

Підпрограма
filterThirdArray(item, index)
повернути item > MIN_COMMON_VALUE;
Все підпрограма

Початок
Ініціалізація змінних
forEach(array1, calcFirstArray)
forEach(array2, calcSecondArray)
Визначення третього масиву
Фільтрація третього масиву
Обчислення суми відфільтрованого масиву
Виведення sum
Кінець

Крок 4:

Підпрограма
forEach(array, callback)
для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**
array[i] = callback(array[i], i);
все повторити
Все підпрограма

Підпрограма
filter(array, callback)
newArray[10] = {};
filledIndex = 0
для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**

```
                res = callback(array[i], i);
                якщо res == true
                        то
                                newArray[filledIndex] = array[i]
                                filledIndex= filledIndex + 1;
                все якщо
        все повторити
    повернути newArray
Все підпрограма

Підпрограма
    calcFirstArray(item, index)
        повернути 95 + index;
Все підпрограма

Підпрограма
    calcSecondArray(item, index)
        повернути 105 - index;
Все підпрограма

Підпрограма
    filterThirdArray(item, index)
        повернути item > MIN_COMMON_VALUE;
Все підпрограма

Початок
    Ініціалізація змінних
    forEach(array1, calcFirstArray)
    forEach(array2, calcSecondArray)
    для i від 0 до ARRAY_LENGTH з кроком 1 повторити
        для j від 0 до ARRAY_LENGTH  з кроком 1 повторити
            якщо array[i] == array[2]
                    то
                            array[filledIndex] = array1[i];
                            filledIndex= filledIndex + 1;
            все якщо
        все повторити
```

все повторити

Фільтрація третього масиву

Обчислення суми відфільтрованого масиву

Виведення sum

Кінець

Крок 5:

Підпрограма

forEach(array, callback)

для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**

array[i] = callback(array[i], i);

все повторити

Все підпрограма

Підпрограма

filter(array, callback)

newArray[10] = {};

filledIndex = 0

для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**

res = callback(array[i], i);

якщо res == true

то

newArray[filledIndex] = array[i]

filledIndex= filledIndex + 1;

все якщо

все повторити

повернути newArray

Все підпрограма

Підпрограма

calcFirstArray(item, index)

повернути 95 + index;

Все підпрограма

Підпрограма

calcSecondArray(item, index)

повернути 105 - index;

Все підпрограма

Підпрограма
filterThirdArray(item, index)
повернути item > MIN_COMMON_VALUE;
Все підпрограма

Початок
Ініціалізація змінних
forEach(array1, calcFirstArray)
forEach(array2, calcSecondArray)
для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**
для j **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**
якщо array[i] == array[2]
то
array[filledIndex] = array1[i];
filledIndex= filledIndex + 1;
все якщо
все повторити
все повторити
filteredArray = filter(array3, filterThirdArray);
Обчислення суми відфільтрованого масиву
Виведення sum
Кінець

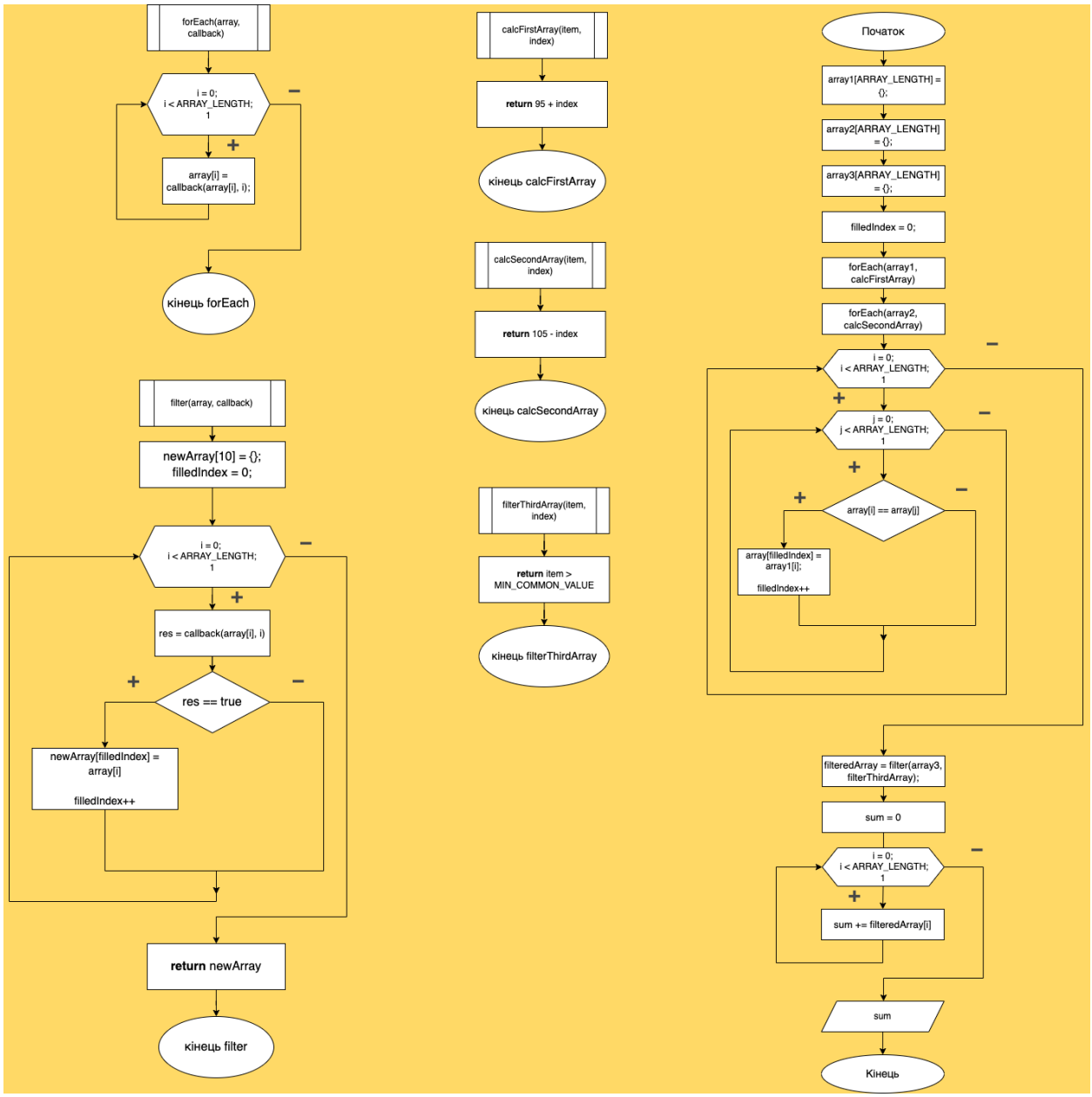
Крок 6:
Підпрограма
forEach(array, callback)
для i **від** 0 **до** ARRAY_LENGTH **з кроком** 1 **повторити**
array[i] = callback(array[i], i);
все повторити
Все підпрограма

Підпрограма
filter(array, callback)
newArray[10] = {};
filledIndex = 0

```
            для i від 0 до ARRAY_LENGTH з кроком 1 повторити
                res = callback(array[i], i);
                якщо res == true
                    то
                        newArray[filledIndex] = array[i]
                        filledIndex= filledIndex + 1;
                все якщо
            все повторити
        повернути newArray
Все підпрограма

Підпрограма
        calcFirstArray(item, index)
            повернути 95 + index;
Все підпрограма

Підпрограма
        calcSecondArray(item, index)
            повернути 105 - index;
Все підпрограма

Підпрограма
        filterThirdArray(item, index)
            повернути item > MIN_COMMON_VALUE;
Все підпрограма

Початок
        Ініціалізація змінних
        forEach(array1, calcFirstArray)
        forEach(array2, calcSecondArray)
        для i від 0 до ARRAY_LENGTH з кроком 1 повторити
            для j від 0 до ARRAY_LENGTH  з кроком 1 повторити
                якщо array[i] == array[2]
                    то
                        array[filledIndex] = array1[i];
                        filledIndex= filledIndex + 1;
                все якщо
```

```
            все повторити
        все повторити
        filteredArray = filter(array3, filterThirdArray);
        sum = 0;
        для i від 0 до ARRAY_LENGTH з кроком 1 повторити
            sum = sum + filteredArray[i];
        все повторити
        Виведення sum
Кінець
```

4) Блок схема алгоритму



5) Код програми

main.cpp

```cpp
#include <iostream>

void forEach(char array[], int (*callback)(int, int));

char *filter(char array[], bool (*callback)(int, int));

int calcFirstArray(int item, int index);

int calcSecondArray(int item, int index);

bool filterThirdArray(int item, int index);

using namespace std;

const int ARRAY_LENGTH = 10;
const int MIN_COMMON_VALUE = 101;

int main() {
    char array1[ARRAY_LENGTH] = {};
    char array2[ARRAY_LENGTH] = {};
    char array3[ARRAY_LENGTH] = {};
    int filledIndex = 0;

    forEach(array1, calcFirstArray);
    forEach(array2, calcSecondArray);

    for (int i = 0; i < ARRAY_LENGTH; i++) {
        for (int j = 0; j < ARRAY_LENGTH; j++) {
            if ((int) array1[i] == (int) array2[j]) {
                array3[filledIndex] = array1[i];
```

main

main.cpp

```
26
27      for (int i = 0; i < ARRAY_LENGTH; i++) {
28          for (int j = 0; j < ARRAY_LENGTH; j++) {
29              if ((int) array1[i] == (int) array2[j]) {
30                  array3[filledIndex] = array1[i];
31                  filledIndex++;
32              }
33          }
34      }
35
36      char *filteredArray = filter(array3, filterThirdArray);
37      int sum = 0;
38      for (int i = 0; i < ARRAY_LENGTH; i++) {
39          sum += (int) filteredArray[i];
40      }
41
42      cout << "The sum is: " << sum << endl;
43
44      return 0;
45  }
46
47  void forEach(char array[], int (*callback)(int, int)) {
48      for (int i = 0; i < ARRAY_LENGTH; i++) {
49          array[i] = (char) callback( (int) array[i], i);
50      }
51  }
52
53  char *filter(char array[], bool (*callback)(int, int)) {
54      char newArray[ARRAY_LENGTH] = {};
55      int filledIndex = 0;
```

main

main.cpp

```cpp

char *filter(char array[], bool (*callback)(int, int)) {
    char newArray[ARRAY_LENGTH] = {};
    int filledIndex = 0;
    for (int i = 0; i < ARRAY_LENGTH; i++) {
        bool res = callback( (int) array[i], i);
        if (res) {
            newArray[filledIndex] = array[i];
            filledIndex++;
        }
    }
    return newArray;
}

int calcFirstArray(int item, int index) {
    return 95 + index;
}

int calcSecondArray(int item, int index) {
    return 105 - index;
}

bool filterThirdArray(int item, int index) {
    return item > MIN_COMMON_VALUE;
}

```

6) Виновки:

Дослідив методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набув практичних навичок їх використання під час складання програмних специфікацій. Розв’язав задачу, побудував мат. модель, блок схему, написав псевдокод і код на мові С++