

Основи програмування – 1. Алгоритми та структури даних

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 13

Виконав студент

ІП-15 Конденко Іван Ігорович

(шифр, прізвище, ім'я, по батькові)

Перевірів(-ла)

Вечерковська Анастасія Сергіївна

(прізвище, ім'я, по батькові)

Київ 2021

Лабораторна робота 7

Дослідження лінійного пошуку в послідовностях

Мета — дослідити методи послідовного пошуку у впорядкованих і неупорядкованих послідовностях та набуті практичних навичок їх використання під час складання програмних специфікацій.

Індивідуальне завдання

Варіант 13

Постановка задачі

Завдання

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису трьох змінних індексованого типу з 10 символічних 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 елемента		Знайти
	1-го масиву	2-го масиву	
13	$62 + 3 * i$	$74 - i$	Суму кодів мінімального та максимального елементів

Математична модель

Змінна	Тип	Ім'я	Призначення
Перший масив	Символьний[10]	mass1	Вхідні дані
Другий масив	Символьний[10]	mass2	Вхідні дані
Третій масив	Символьний[10]	mass3	Проміжні дані
Розмірність третього масиву	Цілочисельний	k	Проміжні дані
Мінімум	Цілочисельний	min	Проміжні дані
Максимум	Цілочисельний	max	Проміжні дані
Сума мінімального та максимального кодів елементів масиву	Цілочисельний	s	Вихідні дані
Функція для побудови першого та другого масиву	Пустий	bud_mass(char mass1[10], char mass2[10])	Проміжні дані
Функція для побудови третього масиву та обчислення суми кодів мінімального та максимального елементів	Цілочисельний	bud_mass3(char mass1[10], char	Проміжні дані

Основи програмування – 1. Алгоритми та структури даних

		mass2[10], mass3[10])	
Ітератор	Цілочис льний	i	Проміжні дані
Ітератор	Цілочис льний	j	Проміжні дані

Перший та другий масив будується за допомогою ф-ії `bud_mass` в якій знаходиться цикл. Побудова третього масиву, знаходження мінімального та максимального кодів елементів і їх суми буде виконуватися за допомогою функції `bud_mass3`. Для побудови третього масиву використовується перебір елементів двох інших масивів і якщо вони співпадають, то елемент записується у третій масив. Кожне проходження циклу додає 1 до довжини третього масиву. Хоч і всі три масиви відсортовані через особливість їх заповнення, для наочності та універсальності програми, ми знайдемо максимальний та мінімальний елемент за допомогою циклу який порівнює минуле значення з наступним і перезаписує його, в залежності від умови.

Розв'язання

Псевдокод

Початок

Введення `mass1, mass2, mass3`

`bud_mass(mass1, mass2)`

`s = bud_mass3(mass1, mass2, mass3)`

Виведення `s`

Кінець

Функція `bud_mass(char mass1[10], char mass2[10])`

повторити для `i` від 0 до `i < 10`

`mass1[i] = 62 + 3*i`

`mass2[i] = 74 - i`

все повторити

Функція `bud_mass3(char mass1[10], char mass2[10], char mass3[10])`

`k=0`

повторити для `i` від 0 до `i < 10`

повторити для `j` від 0 до `j < 10`

якщо `mass1[i] == mass2[j]` **то**

`mass3[k] = mass1[i]`

`k++`

все якщо

все повторити

все повторити

$\min = \text{mass3}[0]$

$\max = \text{mass3}[0]$

повторити для i від 0 до $i < k$

якщо $\text{mass3}[i] > \max$ то

$\max = \text{mass3}[i]$

все якщо

якщо $\text{mass3}[i] < \min$ то

$\min = \text{mass3}[i]$

все якщо

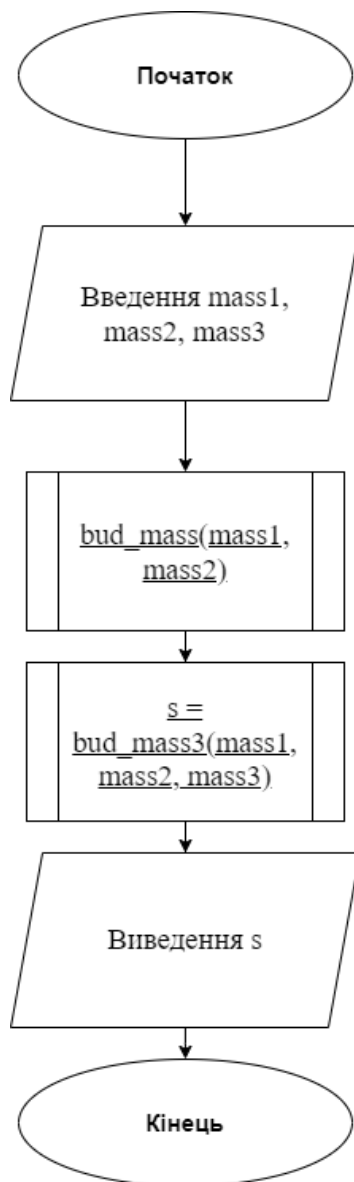
все повторити

$s = \min + \max$

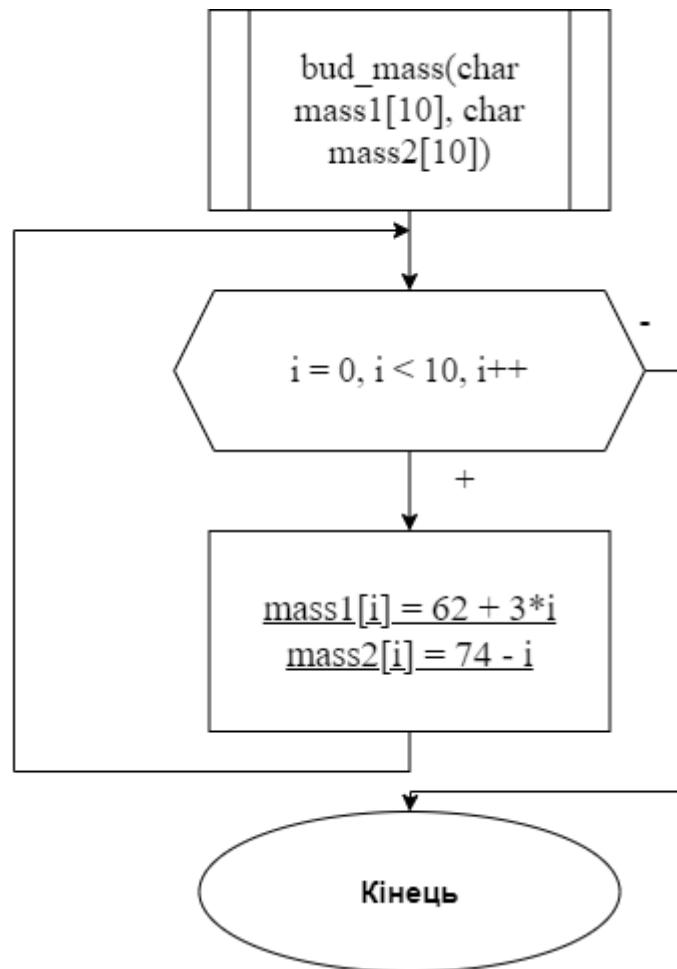
повернути s

Блок схема

Основна програма



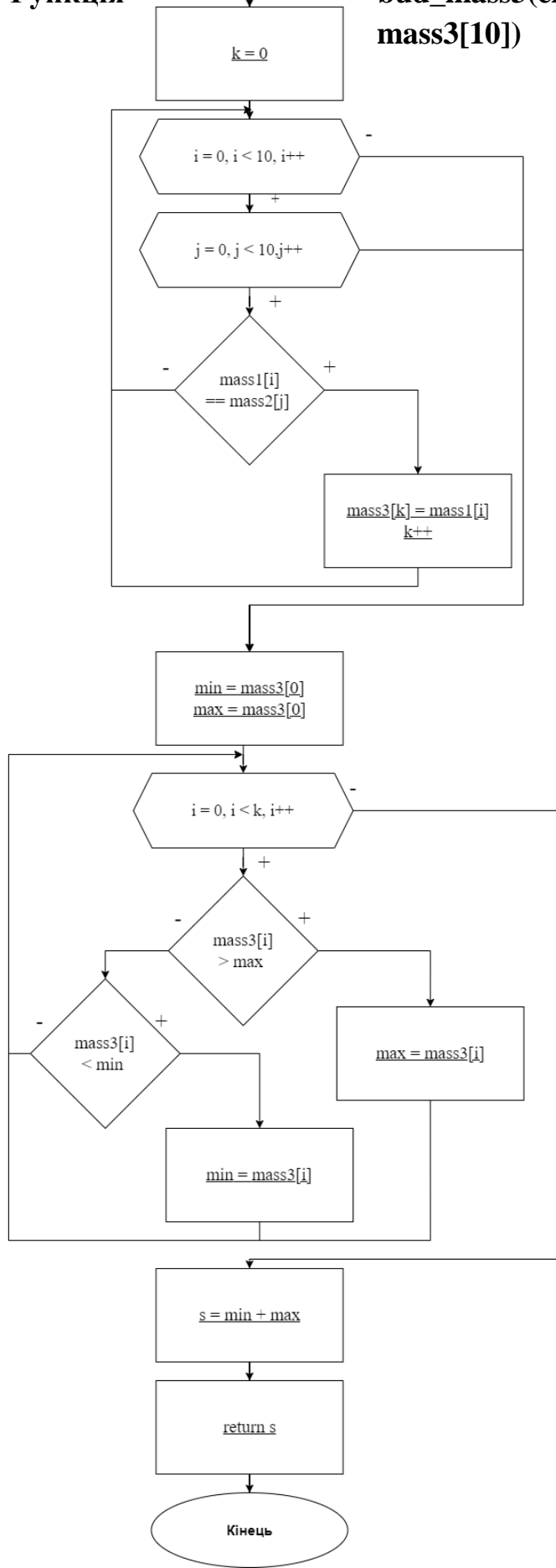
Функція `bud_mass(char mass1[10], char mass2[10])`



Функція

```
bud_mass3(char  
mass1[10], char  
mass2[10], char  
mass3[10])
```

bud_mass3(char mass1[10], char mass2[10], char mass3[10])



Код програми

```
#include <iostream>

using namespace std;

void bud_mass(char mass1[10], char mass2[10]) {
    for (int i = 0; i < 10; i++) {
        mass1[i] = 62 + 3 * i;
        mass2[i] = 74 - i;
    }
}

int bud_mass3(char mass1[10], char mass2[10], char mass3[10]) {
    int k = 0;

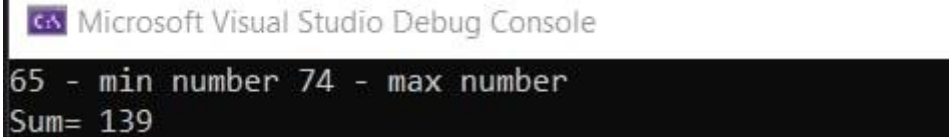
    for (int i = 0; i < 10; i++) {
        for (int j = 0; j < 10; j++){
            if (mass1[i] == mass2[j]) {
                mass3[k] = mass1[i];
                k++;
            }
        }
    }

    int min = mass3[0];
    int max = mass3[0];
    for (int i = 0; i < k; i++)
    {
        if (mass3[i] > max){
            max = mass3[i];
        }
        if (mass3[i] < min){
            min = mass3[i];
        }
    }
    cout << min << " - min number " << max << " - max number " << endl;
    int s = min + max;

    return s;
}

int main()
{
    char mass1[10];
    char mass2[10];
    char mass3[10];
    bud_mass(mass1, mass2);
    int s = bud_mass3(mass1, mass2, mass3);
    cout << "Sum= " << s;
}
```

Випробування програми



```
Microsoft Visual Studio Debug Console  
65 - min number 74 - max number  
Sum= 139
```

Висновки

Ми дослідили методи послідовного пошуку у впорядкованих і невлпорядкованих послідовностях та набули практичних навичок їх використання під час складання програмних специфікацій. У результаті ми отримали програму, яка знаходить спільні елементи двох масивів та об'єднує їх в третій, знаходить мінімальну та максимальне значення кодів елементів та їх суму.