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 рекурсивних алгоритмів» Варіант 2

Виконав студент ІП-14, Бабіч Денис Володимирович (шифр, прізвище, ім'я, по батькові)

Перевірила Мартинова Оксана Петрівна

( прізвище, ім'я, по батькові)

Київ 2021

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

**Варіант 2**

**Задача**:

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.

3. Ініціювання третьої змінної рівними значеннями двох попередніх змінних.

4. Обробки третьої змінної згідно з варіантом.

D:\Telegram Desktop\tdata\tdummy\image_2021-12-08_23-37-14.png

**Постановка задачі:**

У лабораторній роботі використовуються 3 масиви: **first[SIZE] –** заданий формулою **(5 \* і + 30), second[SIZE] –** заданий формулою **(60 – 5 \* і), result[SIZE] –** заповнюється лише елементами, які є як у першому, так і другом масивах**.** У створенні алгоритму використовується цілочисельна константа **SIZE** для зберігання довжини масиву, вона ініціалізована значенням **10**. Змінна **result\_multiply** використовується для зберігання результату від множення елементів третього масиву. **first\_element –** використовується для тимчасвого зберігання елемента масиву під-час порівняння першого та другого масивів. Змінна **index** використовується для зберігання індексу роглядаємого елемента в масиві **result[SIZE].**Аргумент **SIZE** вважати не включеним у тілі арифметичного циклу, всі ітератори у кожному арифметичному циклі у ході роботи програми інкрементуються. За допомогою конструкції **result[SIZE] := { 0 }** масив ініціалізується нулями.

**Побудова математичної моделі:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| Перший масив | Символьний | first | Масив значень, проміжне значення |
| Другий масив | Символьний | second | Масив значень, проміжне значення |
| Третій масив | Символьний | result | Масив значень, результуючий масив |
| Індекс третього масиву | Ціле число | index | Для ітерування по третьому масиву |
| Результат від множення | Ціле число | result\_multiply | Резултьтат роботи алгоритму |
| Зберігання поточного елемента першого масиву | Символьний | first\_element | Розглядаємий елемент третього масиву |
| Індекс у арифметичному циклі | Ціле число | i | Ітератор у циклі |
| Індекс у арифметичному циклі | Ціле число | k | Ітератор у циклі |
| Індекс у арифметичному циклі | Ціле число | j | Ітератор у циклі |

**Розв’язання**

Крок 1: Визначити основні дії

Крок 2: Ініціалізація початкових даних

Крок 3: Присвоєння значень обом масивам

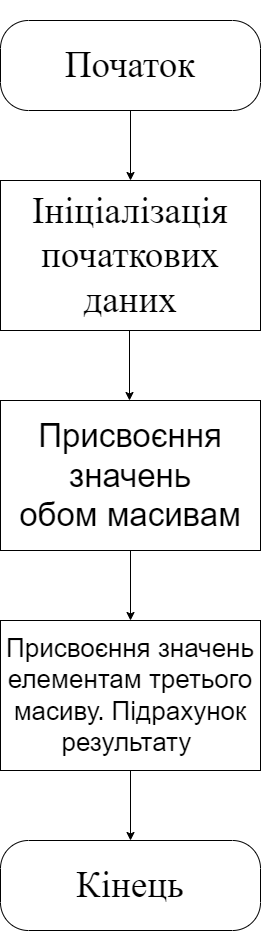
Крок 4: Присвоєння значень елементам третього масиву. Підрахунок результату

**Псевдокод**

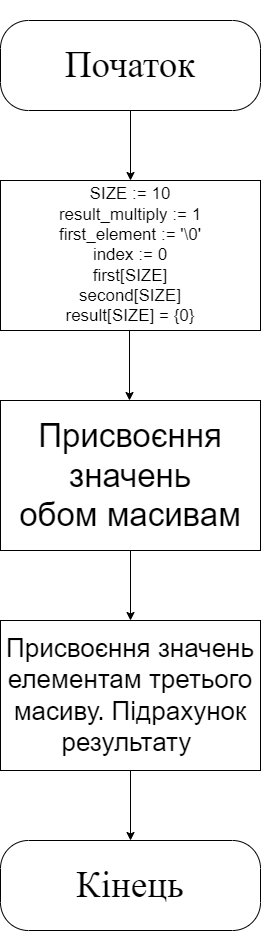
|  |  |
| --- | --- |
| Крок 1 | **Початок**  Визначити основні дії  Ініціалізація початкових даних  Присвоєння значень обом масивам  Присвоєння значень елементам третього масиву. Підрахунок результату  **Кінець** |
| Крок 2 | **Початок**  SIZE := 10  result\_multiply := 1  first\_element := '\0'  index := 0  first[SIZE]  second[SIZE]  result[SIZE] := { 0 }  Присвоєння значень обом масивам  Присвоєння значень елементам третього масиву. Підрахунок результату  **Кінець** |
| Крок 3 | **Початок**  SIZE := 10  result\_multiply := 1  first\_element := '\0'  index := 0  first[SIZE]  second[SIZE]  result[SIZE] := { 0 }  **повторити**  **для i від 0 до SIZE**  first[i] := 5 \* i + 30  second[i] := 60 - 5 \* i  **все повторити**  Присвоєння значень елементам третього масиву. Підрахунок результату  **Кінець** |
|  |  |
| Крок 4 | **Початок**  SIZE := 10  result\_multiply := 1  first\_element := '\0'  index := 0  first[SIZE]  second[SIZE]  result[SIZE] := { 0 }  **повторити**  **для i від 0 до SIZE**  first[i] := 5 \* i + 30  second[i] := 60 - 5 \* i  **все повторити**  **повторити**  **для k від 0 до SIZE**  first\_element := first[k]  **повторити**  **для j від 0 до SIZE**  **якщо** first\_element == second[j]  **то**  result[index] := first\_element  index := index + 1  **все якщо**    **якщо** first\_element < 40  **то**  result\_multiply = first\_element \* result\_multiply  **все якщо**  **все повторити**  **все повторити**  **Кінець** |
|  |  |

**Блок-схема алгоритму**

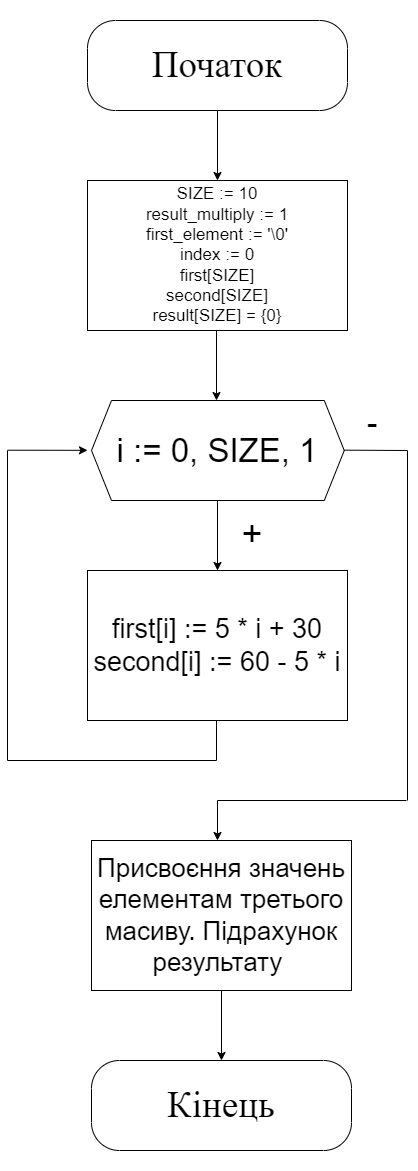
**Крок 1**

****

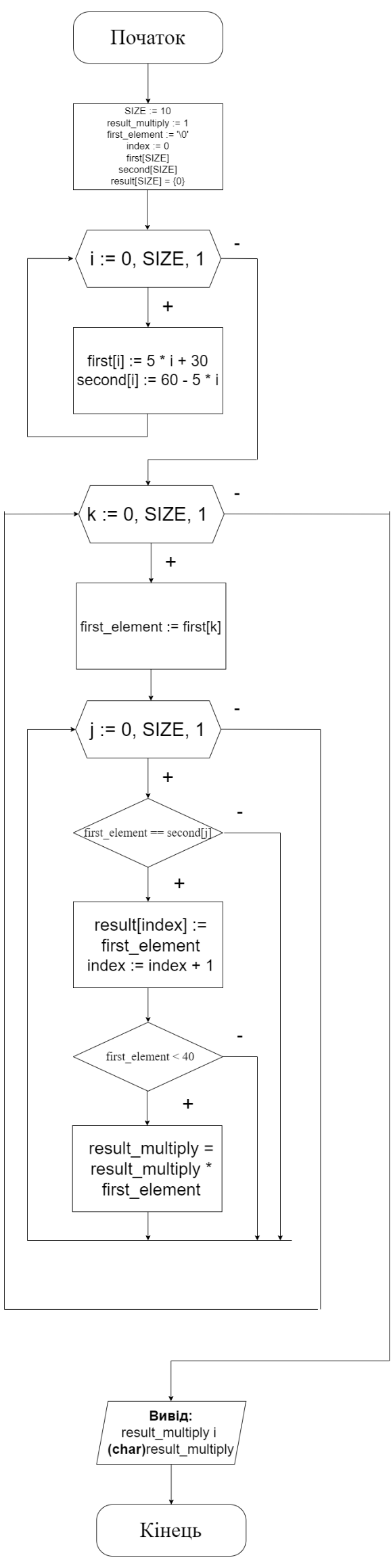
**Крок 2**

****

**Крок 3**

****

**Крок 4**

****

**Випробування алгоритму:**

**Код алгоритму**



**Випробування**

|  |  |
| --- | --- |
| Блок | Дія |
|  | **початок** |
| 1 | //first[i] ▲ 30 //second[i] < 60  //first[i] # 35 //second[i] 7 55  //first[i] ( 40 //second[i] 2 50  //first[i] – 45 //second[i] - 45  //first[i] 2 50 //second[i] ( 40  //first[i] 7 55 //second[i] # 35  //first[i] < 60 //second[i] ▲ 30  //first[i] A 65 //second[i] ↓ 25  //first[i] F 70 //second[i] ¶ 20  //first[i] K 75 //second[i] ☼ 15  // # ( - 2 7 < 0 0 0 – масив result |
| 2 | Вивід: Result (symbol): → Result (number): 1050 |
|  | **кінець** |



**Висновок:**

У результаті лабораторної роботи набув навичок у використані масивів, використовував оператори альтернативного вибору, арифметичні операції і арифметичні цикли. Дослідив метод послідовного пошуку у невпорядкованих послідовностях.