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 5

Виконав студент ІП-11 Гіжицький Даниїл Олександрович

(шифр, прізвище, ім'я, по батькові)

Перевірив Мартинова Оксана Петрівна

( прізвище, ім'я, по батькові)

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

**Варіант 5**

**Завдання:**

Text

Description automatically generated



**Розв’язок:**

**1 Постановка задачі**

**Результатом виконання задачі є кількість елементів, коди яких меньші за 115**

**2 Побудова математичної моделі**

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім'я** | **Призначення** |
| Перший масив | List(chr) | Arr1 | Проміжні дані |
| Другий масив | List(chr) | Arr2 | Проміжні дані |
| Третій масив | List(chr) | Arr3 | Проміжні дані |
| Кількість елементів | chr | Max\_chr | Вихідні данні |

**За формулами 120-i та 110+i заповнюємо перший та другий масиви, а третій заповнюємо елементами які знаходятся і в першому і в другому масиві**

|  |  |  |  |
| --- | --- | --- | --- |
| **Функція** | **Аргументи** | **Ім'я** | **Тип поверненго значення** |
| Виведення кількості елементів | Int n  (кількість елементів масива)  Arr3  (Третій масив заповнений спільними елементами двох попередніх) | count | int (l) |
| Переріз двох масивів | List(chr) arr1  List(chr) arr2 | iners | List(chr) |
| функція залежності елемента від індекса (перший масив) | Int i | faf | chr |
| функція залежності елемента від індекса (другий масив) | Int i | saf | chr |

**Розв’язання**

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

Крок 1. Визначимо дії основної програми.

Крок 2. Деталізуємо функцію init\_array

Крок 3. Деталізуємо функцію iners

Крок 4. Деталізуємо функцію count

Крок 5. Деталізуємо функції fsf і saf

**Псевдокод**

*Крок 1*

Основна програма

**Початок**

Arr1 = init\_array(10, fsf)

Arr2 = init\_array(10, saf)

Arr3 = iners(arr1, arr2)

Count= count(arr3)

Виввід Count

**Кінець**

*Крок 2*

Функція init\_array(int n, char func)

**Початок**

Інніціалізація list arr[n]

Повторити для i від 0 до n

Arr[i] = func(i)

Кінець повторити

Вивід arr

**Кінець**

*Крок 3*

Функція iners(list arr1, list arr2)

**Початок**

Інніціалізація list arr[]

Інніціалізація l=0

Повторити для i в arr1

Повторити для j в arr2

Якщо I == j то

L++

Кінець повторити

Кінець повторити

Вивід arr

**Кінець**

*Крок 4*

Функція count(int n, char\* arr)

**Початок**

Інніціалізація int count =0

Повторити для i в arr

Якщо arr[i]!=0 && arr[i]<115 то

Count++

Кінець повторити

Вивід Count

**Кінець**

*Крок 5*

Функція faf(int i)

**Початок**

Вивід 120 - i

**Кінець**

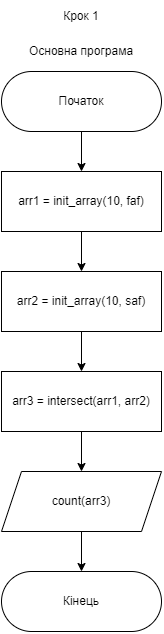
Функція saf(int i)

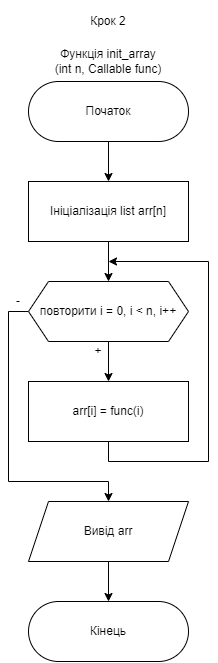
**Початок**

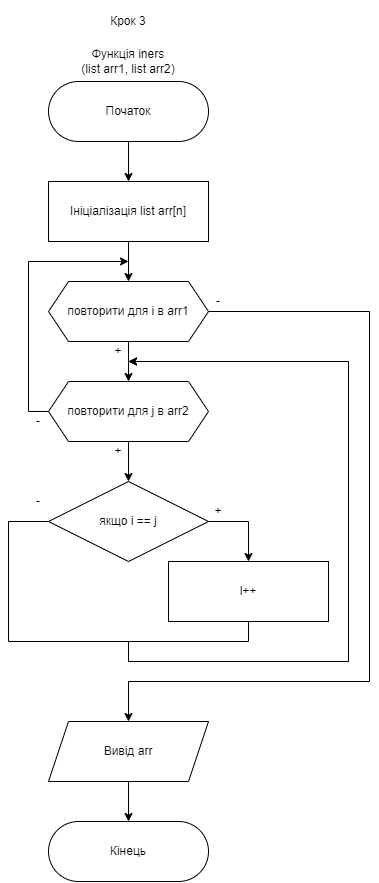
Вивід 110 + i

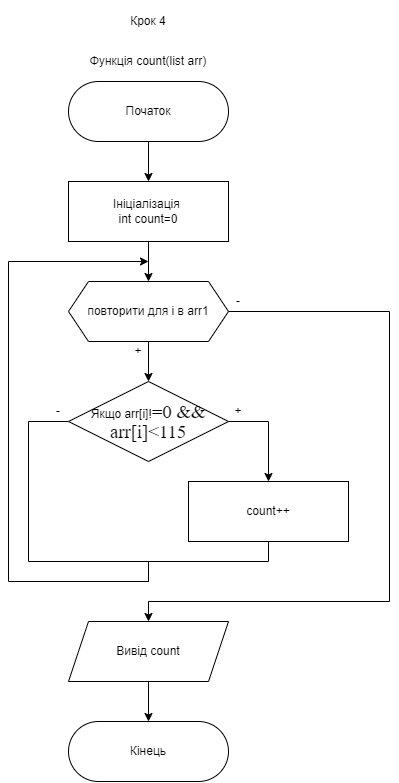
**Кінець**

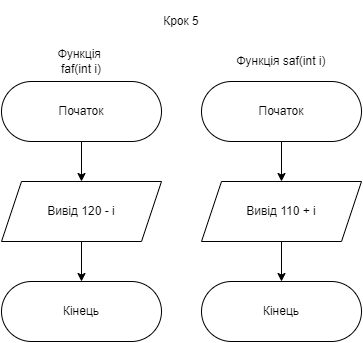
**Блок схеми**











**Код програми (c++)**

#include <iostream>

using namespace std;

char\* init\_array(int n, char func (int)) {

char\* arr = new char[n];

for (int i = 0; i < n; i++) {

arr[i] = func(i);

}

return arr;

}char\* iners(int n, char\* arr1, char\* arr2) {

char\* arr = new char[n] {};

int l = 0;

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

if (arr1[i] ==arr2[j])

{

arr[l] = arr1[i];

l++;

}

}

}

return arr;

}

int count(int n, char\* arr) {

int count = 0;

for (int i = 0; i < n; i++)

{

if (arr[i] !=0 && arr[i] < 115)

{

count++;

}

}

return count;

}

char faf(int i) {

return 120 - i;

}

char saf(int i) {

return 110 + i;

}

int main()

{

const int n = 10;

char\* arr1 = init\_array(n, faf);

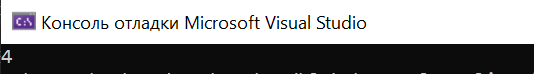
char\* arr2 = init\_array(n, saf);

char\* arr3 = iners(n, arr1, arr2);

cout << count(n, arr3);

}

**Тестування**



**count = 4**

**Висновок**

Ми дослідили методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набули практичних навичок їх використання під час складання програмних специфікацій, створили алгоритм виконання та написали програму результатом якої є 4, що відповідає дійсності.