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

«Дослідження лінійного пошуку в послідовностях»

Варіант 29

Виконав студент ІП-15 Рибалка Ілля Сергійович
(шифр, прізвище, ім'я, по батькові)

Перевірив Вєчерковська 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 невпорядкованих послідовностях та набути практичних навичок їх використання під час складання програмних специфікацій.

Індивідуальне завдання

Варіант 29

Завдання

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.

№	Вираз для обчислення елемента		Знайти
	1-го масиву	2-го масиву	
29	i + 59	64 - i	Елементи, які менші за середньоарифметичне

1. Постановка задачі

Створити два масиви, в арифметичному циклі з лічильником і, за заданими формулами. Третій масив є перетином двох попередніх, після цього необхідно обробити третій масив згідно завдання.

2. Побудова математичної моделі

Змінна	Тип	Ім’я	Призначення
Основна програма			
1 масив	Символьний	lst1	Проміжні дані
2 масив	Символьний	lst2	Проміжні дані
3 масив	Символьний	lst3	Проміжні дані / Результат
Довжина 3 масиву	Ціле	len	Проміжні дані
Середнє арифметичне	Дійсне	arthm	Проміжні дані
Створення 1 та 2 масивів	Підпрограма	crlst	Початкові дані
Створення 3 масиву	Підпрограма	interlst	Початкові дані
Середнє арифметичне	Підпрограма	arthmean	Початкові дані
Сума елементів 3 масиву	Ціле	sum	Проміжні дані
Результат (Середнє арифметичне)	Дійсне	res	Проміжні дані

Обробка 3 масиву	Підпрограма	finlst	Початкові дані
Вивід масиву	Підпрограма	lstout	Початкові дані
Довільний масив	Символьний	lst	Проміжні дані
Номер масиву	Ціле	n	Проміжні дані
Лічильник	Ціле	i	Проміжні дані
Лічильник	Ціле	j	Проміжні дані

Генеруємо перші 2 масиви, згідно завдання, за формулами lst1[i]=i + 59, lst2[i]=64 - i, цю дію буде виконувати підпрограма *crlst*.

Підпрограма *interlst* генерує 3 масив за рахунок перебору елементів 2 попередніх масивів в арифметичному циклі та вкладеному ітераційному циклі з постумовою, обидва цикли працюють від 0 до 9. Якщо програма знаходить 2 однакових елементи в масивах, то цей елемент додається в 3, і для оптимізації процесу вкладений цикл завершує роботу.

Для знаходження середнього арифметичного необхідна кількість ненульових елементів(*len*) в 3 масиві, для цього підпрограма *iterlst* додатково рахує, після присвоєння нового елемента збільшуючи на 1, і повертає ціле значення кількості елементів в 3 масиві.

Підпрограма *arthmean* знаходить середнє арифметичне, за рахунок суми елементів 3 масиву від 1 до *len*, і повертає його.

Підпрограма *finlst* обробляє 3 масив, присвоюючи елементам коди яких більші за середнє арифметичне значення 0.

Згідно вимог усі вхідні та отримані в ході програми значення будуть виводитися в консолі. Дію виведення довільного масиву виконуватиме підпрограма *lstout*, на вхід вона приймає масив та його номер, у разі відсутності номеру масив вважається результатом роботи програми.

Розв’язання

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.
Крок 1. Визначимо основні дії.

Крок 2. Деталізуємо дію створення 1 та 2 масивів за рахунок підпрограми crlst.

Крок 3. Деталізуємо дію виведення масиву за рахунок підпрограми lstout.

Крок 4. Деталізуємо дію знаходження 3 масиву за рахунок підпрограми interlst.

Крок 5. Деталізуємо дію знаходження середнього арифметичного за рахунок підпрограми arthmean.

Крок 6. Деталізуємо дію обробки 3 масиву за рахунок підпрограми finlst.

Псевдокод

Основна програма

крок 1
початок
crlst(lst1, lst2)
len = interlst(lst1, lst2, lst3)
lstout(lst1, 1)
lstout(lst2, 2)
lstout(lst3, 3)
Вивід len
arthm = arthmean(lst3, len)
Вивід arthm
finlst(lst3, arthm, len)
lstout(lst3)
кінець

Підпрограми

крок 2
crlst(lst1, lst2)
для i **від** 0 **до** 9 **повторити**
    lst1[i]=i+59
    lst2[i]=64-i
все повторити
кінець crlst

крок 3
lstout(lst, n=0)
якщо n!=0 **то**
    Вивід "Масив №n = ["
інакше
    Вивід "Результат = ["
все якщо
для i **від** 0 **до** 10 **повторити**
    якщо lst[i]!=0 **то**
        Вивід lst[i]
    все якщо
    якщо lst[i+1]!=0 && i!=9 **то**
        Вивід ","
    все якщо
все повторити
Вивід "]"
кінець lstout

крок 4

```
interlst(lst1, lst2, lst3)
len = 0
для i від 0 до 10 повторити
        j = 0
        повторити
                якщо lst1[i] == lst2[j]
                        lst3[len] = lst1[i]
                        len++
                все якщо
        поки (lst1[i] != lst2[j]) && ++j<10
        все повторити
все повторити
повернути len
кінець interlst
```

крок 5

```
arthmean(lst, len)
sum = 0
для i від 0 до len повторити
        sum += lst[i]
все повторити
res = sum/len
повернути res
кінець arthmean
```

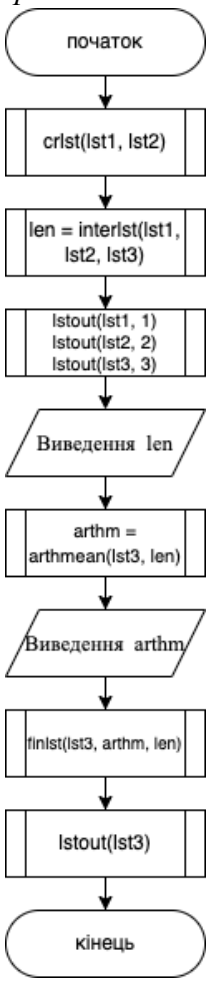
крок 6

```
finlst(lst, arthm, len)
для i від 0 до len повторити
        якщо lst[i]>arthm
                lst[i]=0
        все якщо
кінець finlst
```

Блок-Схема

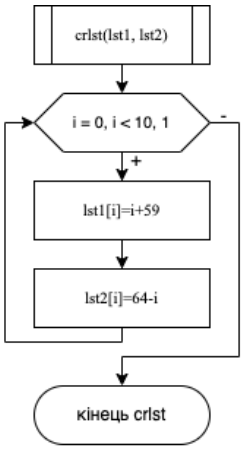
Основна програма

крок 1

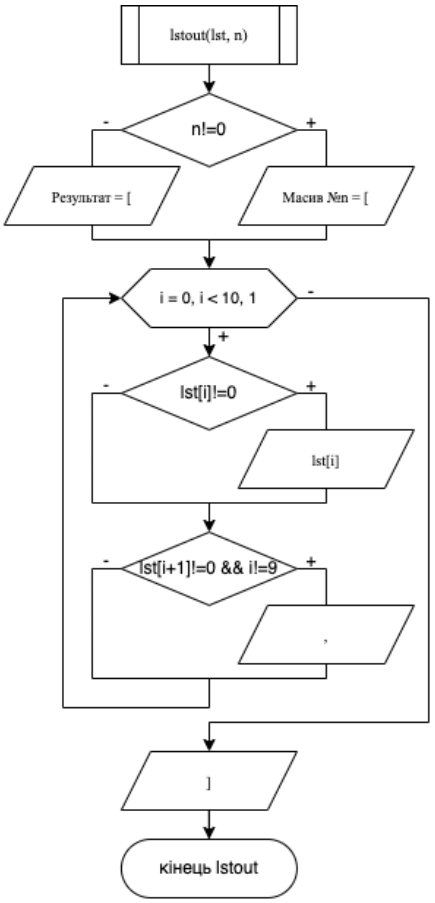


Підпрограми

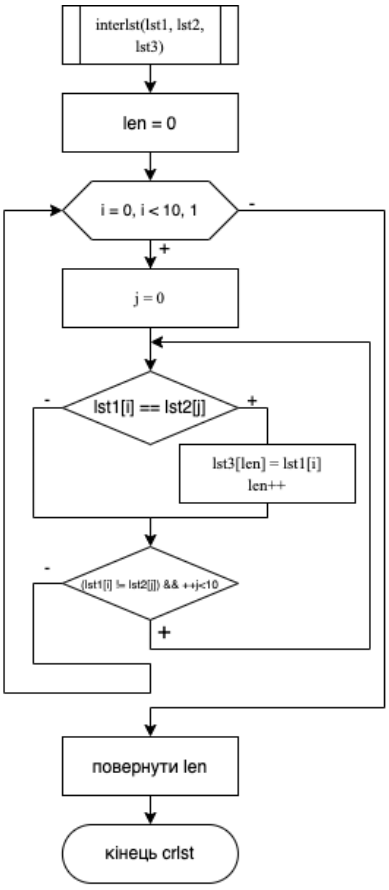
крок 2



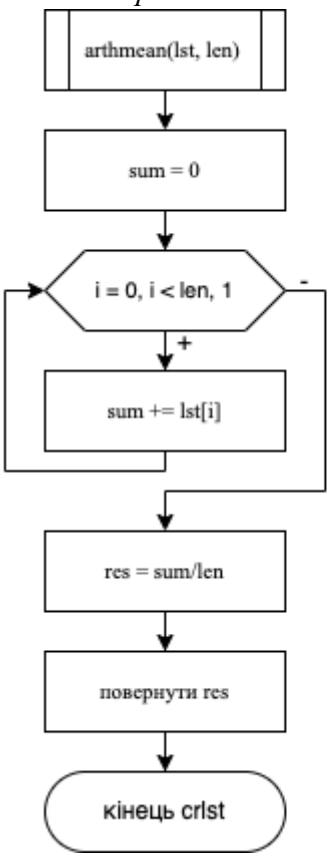
крок 3



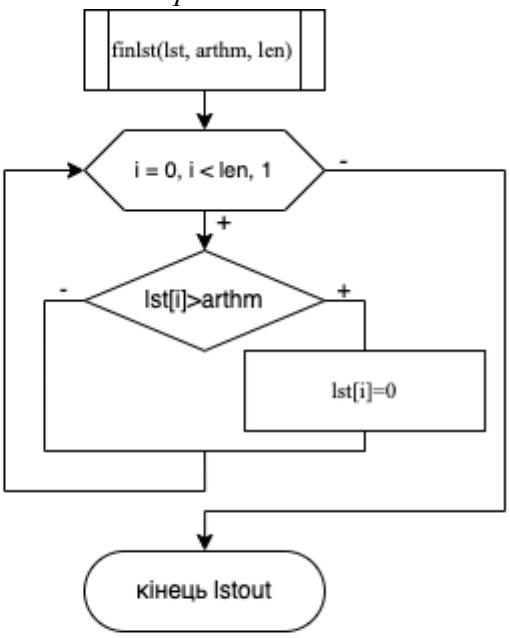
крок 4



крок 5



крок 6



Код

```
#include <iostream>
#include <iomanip>

void crlst(char [], char []); // Створення 1 та 2 масиву
void lstout(char [], int n = 0); // Вивід n-ого масиву
int interlst(char [], char [], char []); // Створення 3-ого масиву та знаходження його довжини
void finlst(char [], float, int); // Обробка 3-ого масиву
float arthmean(char [], int); // Знаходження середнього арифметичного

int main()
{
   char lst1[10], lst2[10], lst3[10];
   int len;
   float arthm;
   crlst(lst1, lst2);
   len = interlst(lst1, lst2, lst3);
   lstout(lst1, 1);
   lstout(lst2, 2);
   lstout(lst3, 3);
   std::cout << "Кількість спільних елементів: " << len << std::endl;
   arthm = arthmean(lst3, len);
   std::cout << "Середнє арифметичне: " << arthm << std::endl;
   finlst(lst3, arthm, len);
   lstout(lst3);
   return 0;
}

void crlst(char lst1[], char lst2[])
{
   for (int i=0; i<10; i++)
       {
           lst1[i]=i+59;
           lst2[i]=64-i;
       }
}

void lstout(char lst[], int n)
{
   if (n!=0) std::cout << "Масив №" << n << " = [";
   else std::cout << "Результат = [";
   for (int i=0; i<10; i++)
   {
       if (lst[i]!=0) std::cout << std::setw(2) << lst[i];
```

```
        if (lst[i+1]!=0 && i!=9) std::cout << ",";
    }
    std::cout << std::setw(2) << "]" << std::endl;
}

int interlst(char lst1[], char lst2[], char lst3[])
{
    int len = 0;
    for (int i = 0; i<10; i++)
    {
        int j = 0;
        do
        {
            if (lst1[i] == lst2[j])
            {
                lst3[len] = lst1[i];
                len++;
            }
        }
        while (lst1[i] != lst2[j] && ++j<10);
    }
    return len;
}

void finlst(char lst[], float arthm, int len)
{
    for (int i=0; i<len; i++)
    {
        if (lst[i]>arthm)
        {
            lst[i]=0;
        }
    }
}

float arthmean(char lst[], int len)
{
    float sum = 0, res;
    for (int i=0; i<len; i++)
    {
        sum += lst[i];
    }
    res = sum/len;
    return res;
}
```

Тестування

```
Масив №1 = [ ;, <, =, >, ?, @, A, B, C, D ]
Масив №2 = [ @, ?, >, =, <, ;, :, 9, 8, 7 ]
Масив №3 = [ ;, <, =, >, ?, @ ]
Кількість спільних елементів: 6
Середнє арифметичне: 61.5
Результат = [ ;, <, = ]
```

Висновок

Я дослідив методи послідовного пошуку у впорядкованих і невпорядкованих послідовностях та набув практичних навичок їх використання під час складання програмних специфікацій. В ході роботи було розроблено алгоритм обробки 3 одновимірних масивів символьного типу.