МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені Тараса Шевченка
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра програмних систем і технологій

Дисципліна
«Алгоритми і структури даних»

Лабораторна робота № 1
«Методи пошуку елементу у масиві даних»

Виконав:	Шевчук Максим Юрійович	**Перевірив**:	Бичков Олексій Сергійович
Група	ІПЗ-12	Дата перевірки	
Форма навчання	денна	Оцінка	
Спеціальність	121		

2022

Умова задачі

Реалізація алгоритму пошуку елемента в певному масиві даних(цілих чисел).
Алгоритми:

1. Лінійний
2. Лінійний з бар'єром
3. Бінарний
4. Бінарний з застосуванням правила золотого перерізу

Структури даних:

1. Масив
2. Двозв'язний список(LinkedList<>)

Аналіз задачі

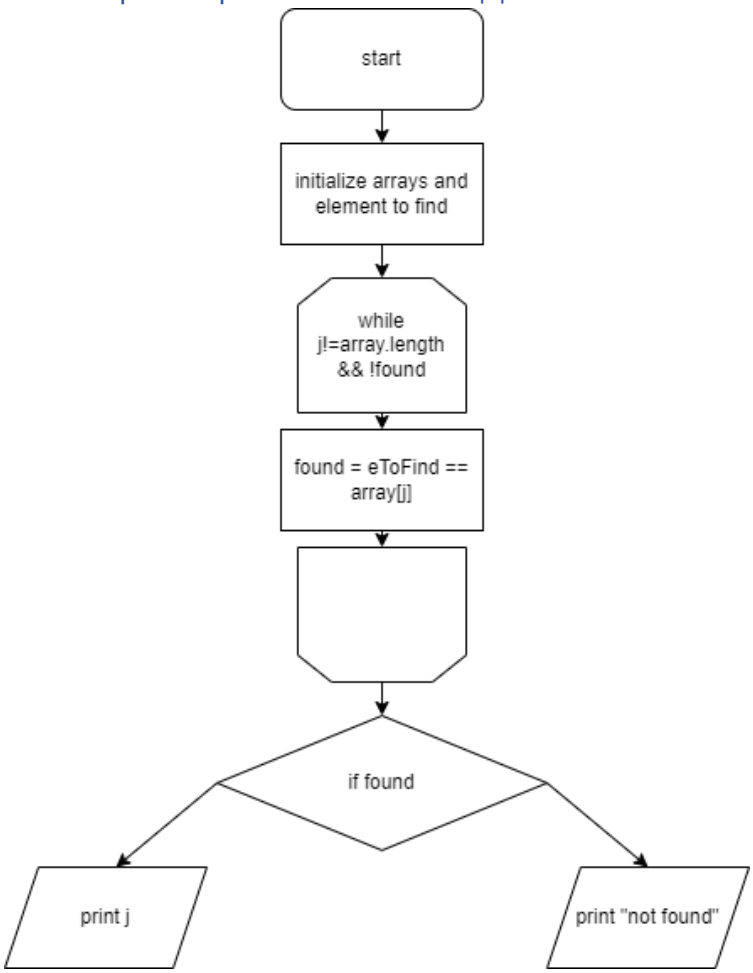
Для реалізації данних алгоритмів необхідно застосувати різні структури даних, а саме масив та зв'язний список, для того, щоб порівняти не тільки самі алгоритми, а й структури даних, при виконанні пошуку елемента.
Для генерації псевдо-випадкових чисел необхідно застосувати клас Random.
Для відстежування часу виконання алгоритму необхідно застосувати клас StopWatch.

Структура основних вхідних та вихідних даних

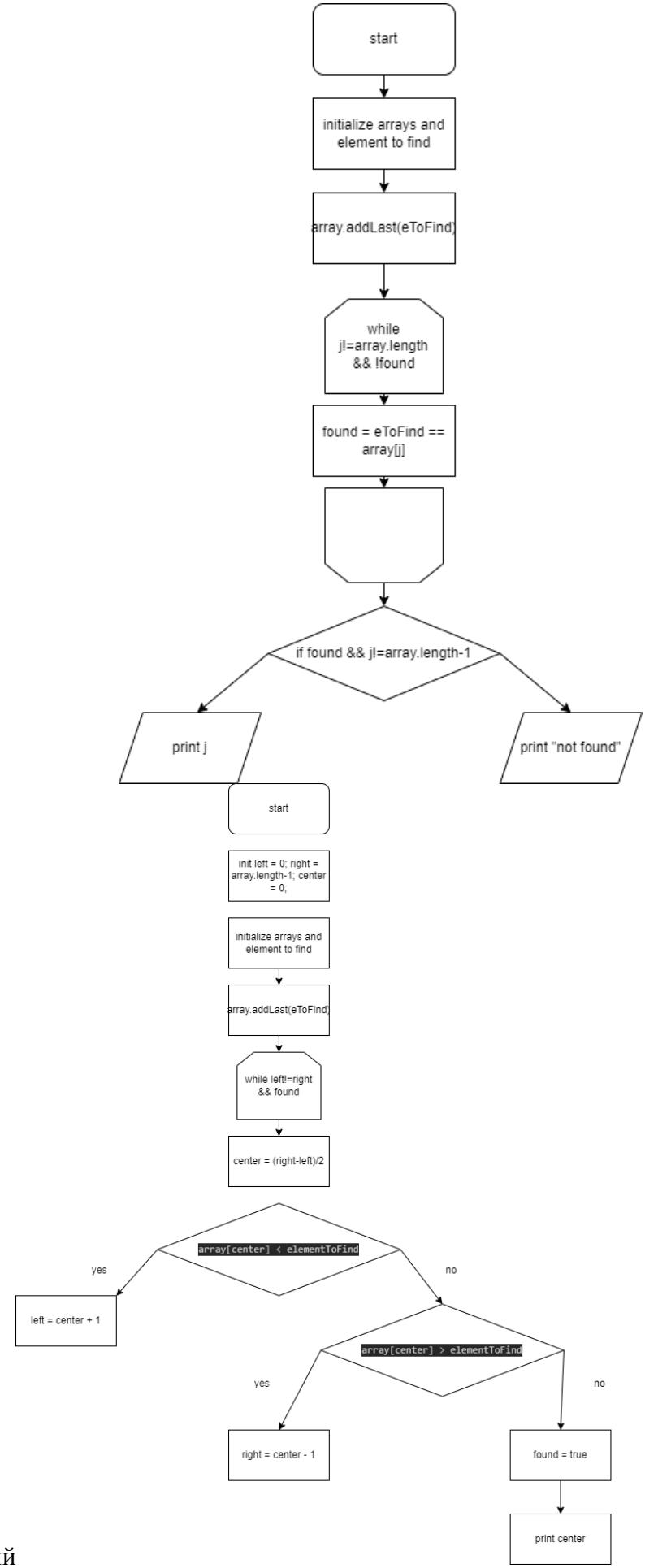
Для порівняння алгоритмів та структур, використовуються однакові вхідні дані. Це - цілі додатні числа. При порівнянні різних алгоритмів, буде використано як відсортовані, так і ні, вхідні дані. У випадку відсортованого масиву - це масив в якому зберігаються числа від 1 до 9_999_999. У невідсортованому масиві це: 1)Псевдовипадкові числа 2)Псевдовипадкові числа від 1 до 100.
Вихідними даними є результат пошуку(знайдено/не знайдено) та індекс знайденого елемента, якщо такий є.

Алгоритм розв’язання задачі

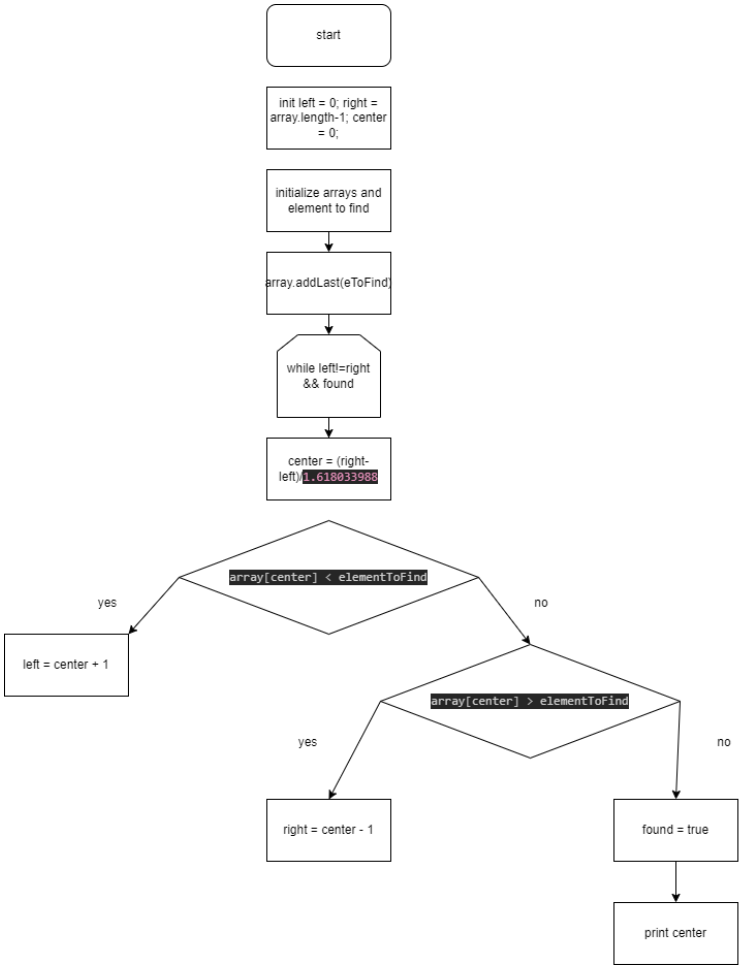
1. Лінійний пошук.



2. Лінійний пошук з бар’єром



3. Бінарний
4. Бінарний з застосуванням пропорції золотого перерізу


start
init left = 0; right = array.length-1; center = 0;
initialize arrays and element to find
array.addLast(eToFind)
while left!=right && found
center = (right-left)/1.618033988
array[center] < elementToFind
yes
no
left = center + 1
array[center] > elementToFind
yes
no
right = center - 1
found = true
print center

Текст програми знаходиться на GitHub!

Набір тестів:

```
var arrayLinBinBig = new int[9_999_999];
var linkedLinBig = new LinkedList<int>();
var arrayBarBig = new int[9_999_999];
var linkedBarBig = new LinkedList<int>();
for (var i = 0; i < arrayLinBinBig.Length; i++)
{
    arrayLinBinBig[i] = i;
    arrayBarBig[i] = i;
    linkedBarBig.AddLast(i);
    linkedLinBig.AddLast(i);
}

var arrayLinBigUnsorted = new int[9_999_999];
var linkedLinBigUnsorted = new LinkedList<int>();
var arrayBarBigUnsorted = new int[9_999_999];
var linkedBarBigUnsorted = new LinkedList<int>();

for (var i = 0; i < arrayLinBinBig.Length; i++)
{
    arrayLinBigUnsorted[i] = new Random().Next();
    linkedLinBigUnsorted.AddLast(arrayLinBigUnsorted[i]);
    arrayBarBigUnsorted[i] = arrayLinBigUnsorted[i];
    linkedBarBigUnsorted.AddLast(arrayLinBigUnsorted[i]);
}

var arrayLinSmall = new int[9_999_999];
var linkedLinSmall = new LinkedList<int>();
var arrayBarSmall = new int[9_999_999];
var linkedBarSmall = new LinkedList<int>();
```

```
var arrayLinSmall = new int[9_999_999];
var linkedLinSmall = new LinkedList<int>();
var arrayBarSmall = new int[9_999_999];
var linkedBarSmall = new LinkedList<int>();

for (var i = 0; i < arrayLinBinBig.Length; i++)
{
    arrayLinSmall[i] = new Random().Next(1, 100);
    linkedLinSmall.AddLast(arrayLinSmall[i]);
    arrayBarSmall[i] = arrayLinSmall[i];
    linkedBarSmall.AddLast(arrayLinSmall[i]);
}
```

Результат тестування програми та аналіз результатів

LINEAR BIG SORTED
Element to find is 4967689
Find element 4967689, index 4967689
Linear search in array used 00:00:00.1113431

Find element4967689, index 4967690
Linear search in linked list used 00:00:00.0459077

LINEAR BIG UNSORTED
Element to find is 80303384
Find element 80303384, index 7470545
Linear search in array used 00:00:00.1315518

Find element 80303384, index 7470545
Linear search in linked list used 00:00:00.0778303

LINEAR SMALL UNSORTED
Element to find is 52
Find element 52, index 113
Linear search in array used 00:00:00.0000008

Find element 52, index 113
Linear search in linked list used 00:00:00.0000025

BARRIER BIG SORTED
Element to find is 4967689
Find element 4967689, index 4967689
Linear search in array used 00:00:00.3602727

Find element 4967689, index 4967689
Linear search in linked list used 00:00:00.0419305

BARRIER BIG UNSORTED
Element to find is 80303384
Find element 80303384, index 7470545
Linear search in array used 00:00:01.0063014

Find element 80303384, index 7470545
Linear search in linked list used 00:00:00.0655483

BARRIER SMALL UNSORTED
Element to find is 52
Find element 52, index 113
Linear search in array used 00:00:00.0000007

Find element 52, index 113
Linear search in linked list used 00:00:00.0000014

BINARY USUAL
Element to find is 4967689
Element 4967689 found, index 4967689
Binary took 00:00:00.0015208 tim

BINARY GOLD
Element 4967689 found, index 4967689
Binary GOLD took 00:00:00.0404862 time

В результаті тестування були отримані наступні результати: найшидшим виявився алгоритм Бінарного Пошуку. Також була помічена тенденція, шо при пошуку елементу в масиві, шо складається з відносно невеликих чисел(1-100), всі алгоритми працюють значно швидше. Найповільніше спрацював лінійний пошук в невідсортованому масиві методом з застосуванням бар’єру. В процесі пошуку застосовувались окрім звичайних масивів також зв’язні списки, які не мали ніякої переваги перед звичайними масивами, і які в залежності від алгоритму працювали то швидше, то повільніше, що вказує на непередбачуваність швидкості роботи цієї структури, при пошуку елемента в ній.