Лабораторна робота №1

З АСД

Виконана студентом

Першого курсу ІПЗ-11-1

Шкітаком Нікітой Ярославовичем

# Умова завдання

Написати програму мовою C# з можливістю вибору різних алгоритмів пошуку. Продемонструвати роботу (ефективність, час виконання) програм на різних структурах даних (масив, лінійний зв’язаний список), з різними умовами, що забезпечують зменшення часу виконання. Навести аналіз отриманих результатів.

Реалізувати алгоритми:

• пошуку перебором елемента масиву, що дорівнює заданому значенню.

• пошуку з бар'єром елемента масиву, що дорівнює заданому значенню.

• бінарного пошуку елемента масиву рівного заданому значенню.

• бінарного пошуку елемента масиву, рівного заданому значенню, в якій нове значення індексу m визначалося б не як середнє значення між L і R, а згідно з правилом золотого перерізу.

# Аналіз завдання

Заради того , спрощення алгоритму програми, будемо працювати виключно з цілими числами ( додатні та від’ємні ). Для кожного методу пошуку будемо створювати окремий статичний клас, методи якого будуть використані для пошуку певних елементів в масиві та в зв’язаному списку. Також для роботи зі зв’язаним списком нам потрібен сам список, який буде як окремий клас. Також потрібен метод який буде перетворювати масив чисел в зв’язаний список

# Структура основних вхідних та вихідних даних

**Вхідні дані :** Масив чисел , який буде переводитися також в формат зв’язаного списку та елемент, який буде шуканий, за декількох методів у програмі з наданого масиву чисел.

**Вихідні дані :** Буде отримана найближча позиція елементу з початку (відлік починається з 0) , а також час , який був затрачений на пошук цього елементу.

# Різні методи розв’язання (Алгоритм)

**Лінійний пошук :**

**ДЛЯ МАСИВУ :**

bool Found = false;

int i = 0, size = array.Length, position = 0;

while (i < size && !Found)

{

if (array[i] == key)

{

position = i;

Found = true;

}

i++;

}

if (Found)

Console.WriteLine(position.ToString());

else

Console.WriteLine("The object is not found");

**ДЛЯ СПИСКУ :**

bool Found = false;

int i = 0, position = 0;

var iterator = list;

while (iterator != null && !Found)

{

if (iterator.Data == key)

{

position = i;

Found = true;

}

i++;

iterator = iterator.Next;

}

if (Found)

Console.WriteLine(position.ToString());

else

Console.WriteLine("The object is not found" );

**Пошук з бар’єром :**

**ДЛЯ МАСИВУ :**

int i = 0;

var newArray = new int[array.Length + 1];

while (i < array.Length)

{

newArray[i] = array[i];

i++;

}

newArray[i] = key;

i = 0;

while (newArray[i] != key )

i++;

if (i < array.Length)

Console.WriteLine(i);

else

Console.WriteLine("The object is not found");

**ДЛЯ СПИСКУ :**

int i = 0;

var iterator = list;

var newLast = new LinkedList(key, null);

while (iterator.Next != null)

iterator = iterator.Next;

iterator.Next = newLast;

iterator = list;

while (iterator.Data != key)

{

i++;

iterator = iterator.Next;

}

if (iterator.Next != null)

Console.WriteLine(i);

else

Console.WriteLine("The object is not found");

**Бінарний пошук (2 випадки : стандартий та зі золотим перерізом):**

**ДЛЯ МАСИВУ :**

int min = 0, max = array.Length - 1, mid;

double adder = (Math.Sqrt(5) + 1) / 2;

double devider = ratio ? 1 + adder : 2;

double multiplayer = ratio ? adder : 1;

while (min <= max)

{

mid = (int)((min + multiplayer \* max) / devider);

if (key < array[mid])

max = mid - 1;

else

min = mid + 1;

}

if(array[max] == key)

Console.WriteLine(max);

else

Console.WriteLine("The object is not found");

**ДЛЯ СПИСКУ :**

int mid = 0, minIndex = 0, maxIndex = LinkedList.GetLength(list);

LinkedList maxList = null, midList = null, minList = list;

double adder = (Math.Sqrt(5) + 1) / 2;

double devider = ratio ? 1 + adder : 2;

double multiplayer = ratio ? adder : 1;

while (maxList == null || minList.Data != maxList.Data)

{

mid = (int)((minIndex + multiplayer \* maxIndex) / devider);

midList = LinkedList.GetElement(minList, mid - minIndex);

if (key < midList.Data)

{

maxList = midList;

maxIndex = mid;

}

else

{

minList = midList.Next;

minIndex = mid + 1;

}

}

if (midList.Data == key)

Console.WriteLine(mid);

else

Console.WriteLine("The object is not found");

# Посилання на GitHub з працюючою аплікацією

[**https://github.com/nickiway/Univ\_1.2/tree/main/ASD/Lab\_ASD\_Shkitak\_1**](https://github.com/nickiway/Univ_1.2/tree/main/ASD/Lab_ASD_Shkitak_1)

# Набори тестів

1. Шуканий елемент = 817 , позиція №32

Кількість елементів у масиві = 40

**Масив :** 1954 34 1386 1856 972 1645 791 210 606 1103 901 1585 1413 95 1437 1009 25 787 1200 342 537 739 1381 408 504 239 1619 1467 709 926 499 473 817 1966 944 1476 1439 1629 1676 445

1. Шуканий елемент = 30, позиція №82

Кількість елементів у масиві = 100

**Масив :** 1411 2532 1787 469 2173 95 2856 2508 2803 2653 1337 2335 469 205 1574 1456 885 2868 531 2918 2256 1053 2755 2249 1665 2002 238 186 1116 1472 1275 2006 1069 1874 2123 414 2834 1847 1413 2032 1977 1844 2472 1765 932 566 190 276 157 476 728 1056 2068 300 1567 359 2776 2537 1803 171 2856 2670 1392 1332 1379 2330 1267 1594 1080 1161 1620 2036 1455 1497 941 413 1579 990 1318 1100 1813 2960 30 641 744 219 2937 769 307 1764 637 298 45 1243 2175 2305 452 1141 387 1600

1. Шуканий елемент = 41, позиція №145

Кількість елементів у масиві = 1000

**Масив :** 35 5 62 37 55 27 45 64 62 98 43 3 57 54 28 62 50 85 33 53 82 76 17 36 22 37 76 85 14 48 35 21 94 76 2 53 80 97 83 38 13 17 60 53 66 28 84 38 23 43 37 5 93 23 72 59 87 26 15 18 51 60 50 15 63 22 9 81 53 75 82 53 3 94 40 66 16 64 70 94 53 38 63 71 11 31 27 72 4 42 65 54 82 65 86 53 66 46 89 44 19 4 85 47 61 83 3 98 82 7 43 24 56 20 65 14 97 79 4 33 94 37 77 11 10 28 70 37 9 87 99 70 74 26 75 49 52 16 10 27 28 29 89 96 99 41 98 63 99 73 56 87 42 56 49 82 26 74 38 34 14 66 90 95 7 72 50 31 44 16 61 81 69 76 6 66 47 80 12 69 30 8 37 35 31 13 29 18 76 92 24 78 78 76 14 61 38 94 90 28 91 67 19 84 13 74 18 66 51 83 36 46 63 68 4 7 29 56 64 94 44 32 54 52 92 83 4 92 63 44 29 53 90 83 77 62 88 53 22 57 95 63 67 93 57 77 15 10 73 7 33 82 30 96 83 60 14 66 91 30 70 25 3 7 55 83 55 79 91 42 18 76 34 8 98 81 65 61 95 15 68 94 20 56 12 46 23 94 1 18 49 22 62 92 86 70 60 60 38 74 23 35 19 31 88 56 48 22 97 3 95 52 71 76 61 75 6 47 95 9 60 61 79 73 92 96 15 42 22 29 21 5 24 21 92 34 76 88 68 55 98 2 97 97 23 97 50 85 32 11 65 14 65 30 14 62 56 45 39 92 42 7 1 68 81 90 29 51 84 28 99 17 78 40 10 59 63 81 51 95 46 29 10 11 64 8 40 93 7 31 78 32 50 75 13 92 1 29 17 32 69 99 2 5 12 48 35 25 20 55 98 74 94 43 46 14 96 89 5 73 51 35 44 54 50 67 67 3 28 18 58 34 65 19 27 46 27 6 99 16 73 15 74 52 32 4 38 7 29 99 96 11 25 43 81 34 56 48 55 44 81 32 97 2 97 64 9 75 17 1 86 90 90 89 1 36 60 92 22 46 30 60 73 28 22 47 10 23 37 92 90 79 51 55 34 41 18 72 54 67 95 62 90 29 13 6 20 36 42 46 74 64 26 10 15 22 58 69 80 50 55 85 39 24 10 8 40 36 55 59 18 88 38 55 50 68 70 45 16 17 27 74 80 91 19 26 53 42 41 13 83 49 91 4 12 10 24 66 19 6 65 84 80 83 28 85 26 71 59 47 51 14 54 64 23 81 5 47 6 83 55 98 95 42 76 69 96 34 43 40 44 5 26 11 51 43 93 97 64 33 1 81 82 93 61 69 94 28 80 31 6 76 61 35 50 67 89 38 8 58 89 92 28 19 3 47 88 44 14 80 88 8 83 72 83 73 16 2 81 7 74 69 53 13 35 67 44 90 60 84 54 55 90 6 43 9 54 64 91 15 81 51 14 1 42 97 93 88 52 77 51 87 57 1 83 21 39 16 84 35 4 97 84 29 27 34 18 77 29 75 20 52 11 66 26 24 55 53 70 37 74 55 39 82 33 66 97 89 22 23 69 38 80 56 12 83 67 84 73 9 80 17 59 76 58 98 77 91 75 60 66 86 45 47 61 48 59 2 96 60 36 28 55 7 7 81 72 55 55 42 56 70 79 61 56 58 95 62 24 35 89 7 14 62 56 83 92 33 94 63 25 65 88 14 73 53 61 53 70 77 27 22 37 34 4 96 7 83 90 4 52 96 77 61 71 30 14 93 50 23 88 39 60 91 18 90 81 66 88 4 96 43 92 47 8 67 70 80 58 60 76 10 43 90 11 28 88 27 43 43 39 68 3 47 88 88 62 45 16 13 16 41 95 86 7 10 5 30 54 4 60 33 35 90 46 79 95 69 80 89 3 54 24 45 65 29 40 45 59 20 6 25 64 46 45 36 14 57 84 2 24 58 73 39 83 6 33 13 1 10 92 92 64 26 11 61 18 50 22 78 69 59 71 35 97 35 69 89 46 11 65 39 85 79 66 45 80 84 62 59 95 28 44 49 27 14 60 38 35 83 19 64 35 7 32 64 86 39 42 49 37 44 66 90 44 54 7 85 59 65 81 33 88 62 82 42 15 22 8 84

# Результати тестування

Тест №1

**Лінійний пошук** Зображення, що містить текст

Автоматично згенерований опис

**Бар’єрний пошук**

Зображення, що містить текст

Автоматично згенерований опис

**Бінарний пошук (стандартний)**

Зображення, що містить текст

Автоматично згенерований опис

**Бінарний пошук (золотий переріз)**

Зображення, що містить текст

Автоматично згенерований опис

Тест №2

**Лінійний пошук**

Зображення, що містить текст

Автоматично згенерований опис

**Бар’єрний пошук**

Зображення, що містить текст

Автоматично згенерований опис

**Бінарний пошук (стандартний)**

Зображення, що містить текст

Автоматично згенерований опис

**Бінарний пошук (золотий переріз)**

Зображення, що містить текст

Автоматично згенерований опис

Тест №3

**Лінійний пошук**

Зображення, що містить текст

Автоматично згенерований опис

**Бар’єрний пошук**

Зображення, що містить текст

Автоматично згенерований опис

**Бінарний пошук (стандартний)**

Зображення, що містить текст

Автоматично згенерований опис

**Бінарний пошук (золотий переріз)**

Зображення, що містить текст

Автоматично згенерований опис

# Аналіз помилок, допущенних при виконанні роботи

Як ми можемо спостерігати з вище наданих скріншотів, помилок в фінальній програмі немає. Спочатку була проблема з розумінням алгоритму золотого перерізу, тому на ранніх стадіях проекту , даний алгоритм працював некоректно. Також була проблема з розумінням як саме працює зв’язаний список, що також ускладнювало роботу. Також проблемою з золотим перерізом було те , що в формулі було забуте відношення ділення в знаменнику, і ділення відбувалося виключно на 2 , через що програма не знаходила елементи за алгоритмом бінарного пошуку.

- формула , яка використовується для золотого перерізу, була забута у знаменнику

# Висновок

Виконано завдання першої лабораторної роботи. Реалізовано різні алгоритми пошуку на різних структурах даних : масив та зв’язаний список. Знаходження найбільш ефективного методу для кожного типу даних залежало від аналізу 3 результатів за часом, який був отриманий при викоританні вище зазначених тестів. В результаті : найефективнішим алгоритмом для масиву виявився бінарний пошук з золотим перерізом, а для зв’язного списку найбільш ефективними методома є також бінарний пошук з золотим перерізом , а також лінійний пошук (коли кількість елементів невелика – більш ефективним є лінійний, коли велика кількість – бінарний пошук з золотим перерізом).