**Линейно търсене(Linear search)**

### От

### Симеон М.

### 10А

## 16.04.2023г.

## Линейно търсене

## Въведение:

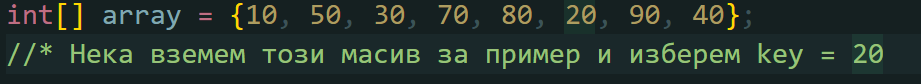
1. Определение:

Линейното търсене, наричан още алгоритъм за последователно търсене. Той е най-лесният алгоритъм за търсене. В него, ние просто обхождаме от началото до самия края целия лист и търсим дадения елемент, докато не го намерим в листа.

1. Как работи алгоритъма ?

* Започва с индекс 0 и сравнява всеки елемент с желания такъв, и връща индекса му. Ако елемента не е намерен, return -1

1. Илюстрация на алгоритъма
2. Diagram

   Description automatically generatedЗапочваме с индекс 0( i = 0) и проверяваме за **key** с arr[0].

7

6

5

4

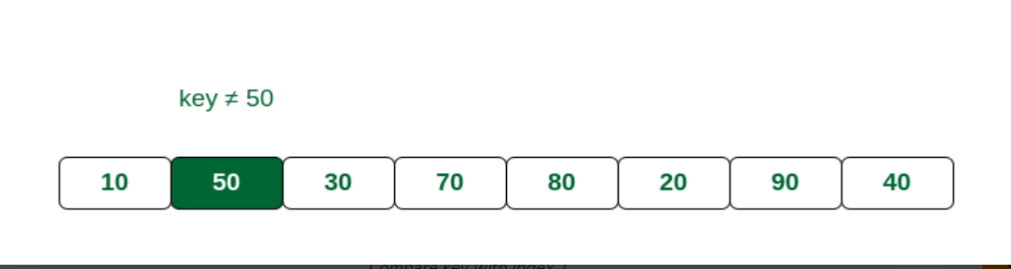
3

2

1

0

1. Key и arr[0] не съвпадат, така че увеличаваме индекса с 1(i= 1) и проверяваме за **key**, като arr[1].



1

2

3

4

5

6

7

0

1. Имаме отново несъвпадение. Увеличаваме индекса(i = 2) и сравняваме(arr[2]).

Diagram

Description automatically generated

7

6

5

4

3

2

1

0

1. arr[2] е различно от key следователно увеличаваме индекса(i = 3) и сравняваме(arr[3]).

Timeline

Description automatically generated

1

2

3

4

5

6

7

0

1. Повтаряме същата операция, както в по гореспоменатите стъпки.

Timeline, waterfall chart

Description automatically generated

5

6

7

3

4

1

0

2

1. Diagram

   Description automatically generatedkey и arr[4] са различни. Увеличаваме индекса(i = 5) и сравняваме(arr[5]).

2

4

5

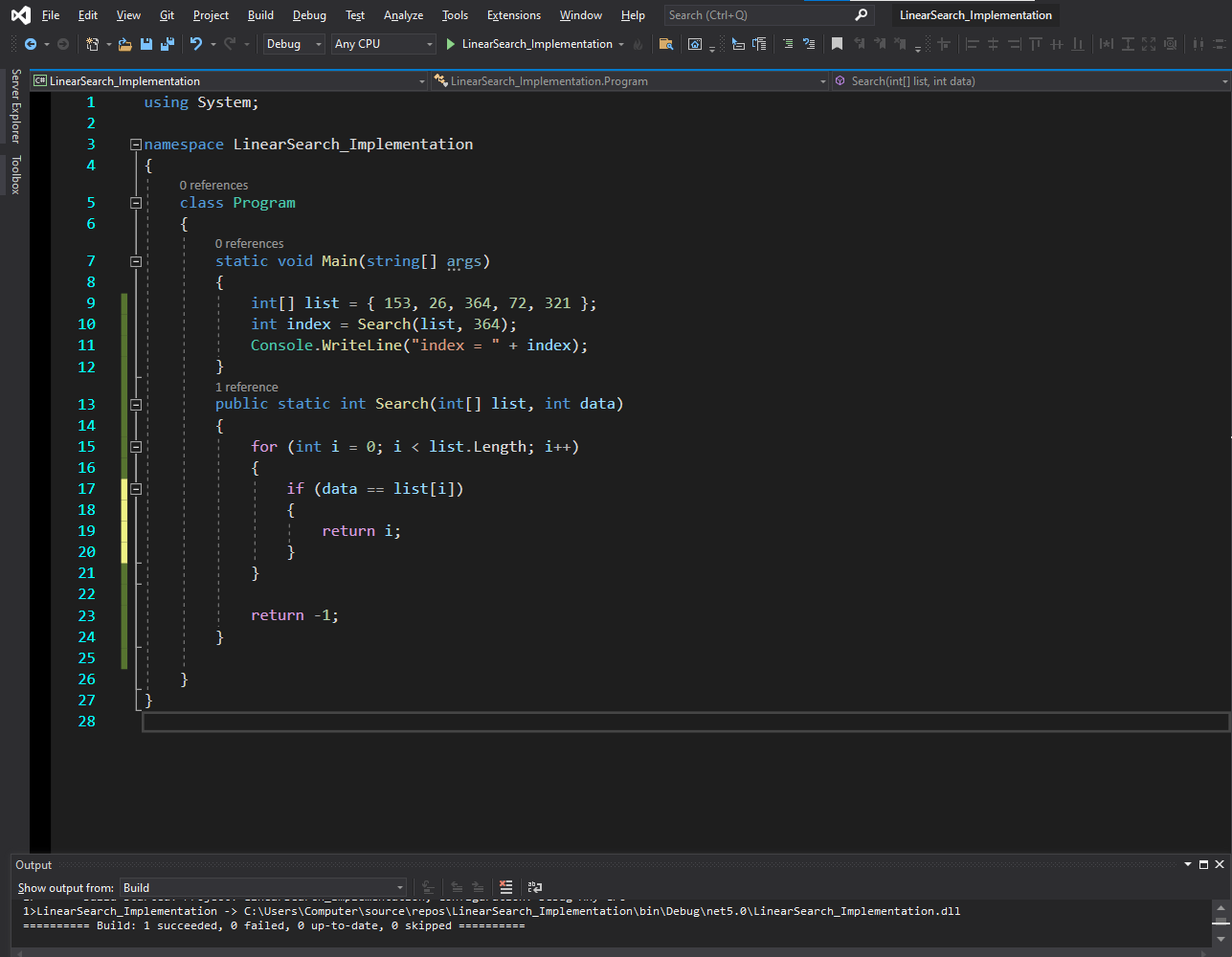
6

7

3

1

0

1. Най-накрая открихме, че key се намира на позиция arr[5].
2. Имплементация на алгоритъма(C#)
   * + Резултат:

Text

Description automatically generated

3 Времева сложност(Time complexity)

1. **Най-добър случай** – имаме, когато елемента, който търсим се намира на първия индекс или в средата. Затова имаме времева сложност от O(1).
2. **Най-лош случай** – имаме, когото елемента се намира в последния индекс. Затова времевата сложност е O(n).
3. **Среден случай** – когато желания елемент се намира някъде в масива имаме O(n).

4 Памет(Space complexity)

1. **Паметта** е O(1), тъй като алгоритъма за търсене не изисква допълнително място.

**5** **Приложение на линейното търсене**

* + - * Алгоритъма е лесен за имплементация и ефективен когато имаме малък набор от елементи.
      * Последователното търсене е също ефективно, когато имаме единично търсене в несортиран масив.
      * Когато търсим точно определена стойност.

**6** **Предимства**

1. Лесно за имплементация и разбиране
2. Може да се използва върху масиви с всякакъв вид данни
3. Не изисква допълнително място
4. Подходящ е за несортиран масив.

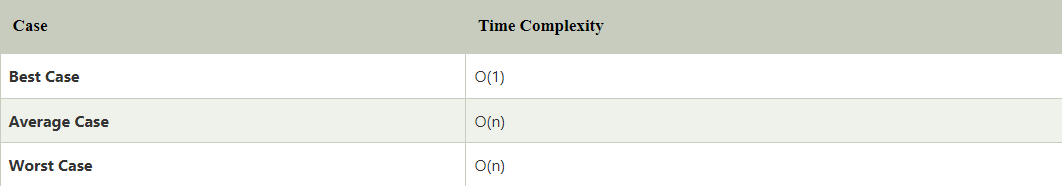
**7** **Недостатъци**

* 1. Отнема много време( в сравнение с другите алгоритми за търсене)
  2. Не е подходящ за голям диапазон от числа
  3. Не е препоръчителен за повтарящи се задачи
  4. Не е подходящ за използването му в приложения в реално време.

**8** **Подобряване на линейното търсене**

* Както се вижда по-горе, времето, необходимо за търсене на елемент с помощта на последователното търсене, е линейно. Това е недостатък при използването на линейно търсене за големи набори от данни. Наблюдава се, че при търсене на ключов елемент има възможност за търсене на същия ключов елемент отново и отново. Можем да подобрим ефективността на линейното търсене, използвайки това наблюдение. Целта е, ако същият елемент се търси отново, операцията да отнема по-малко време. Следователно в такъв случай линейното търсене може да бъде подобрено чрез използване на следните два метода:
* **Преместване(Transposition)**
* **Преместване отпред(Move to front)**

**9** **Обобщение(Summary)**

* Времева сложност и памет:



* Проверява всеки от елементите един по един, последователно, докато открием желания елемент.
* Подходящ за несортирани числа.
* Лесен за усвояване.

10 Препратки

1. <https://www.freecodecamp.org/news/search-algorithms-linear-and-binary-search-explained/>
2. <https://www.geeksforgeeks.org/linear-search/>
3. <https://www.javatpoint.com/linear-search>
4. <https://www.programmingalgorithms.com/algorithm/linear-search/>
5. <https://www.simplilearn.com/tutorials/data-structure-tutorial/linear-search-algorithm#:~:text=A%20linear%20search%20is%20the,the%20target%20element%20is%20located>.
6. <https://aspiringyouths.com/advantages-disadvantages/linear-search/>
7. <https://github.com/BG-IT-Edu/School-Programming/blob/main/Courses/Applied-Programmer/Algo-and-Data-Structures-Basics/05-%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B8-%D0%B7%D0%B0-%D1%82%D1%8A%D1%80%D1%81%D0%B5%D0%BD%D0%B5/05.1.%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B8-%D0%B7%D0%B0-%D1%82%D1%8A%D1%80%D1%81%D0%B5%D0%BD%D0%B5.pdf>