Київський національний університет імені Тараса Шевченка

Кафедра програмних систем і технологій

Звіт про виконання лабораторної роботи з дисципліни

"Структури даних, аналіз і алгоритми комп'ютерної обробки інформації"

Виконав: студент 1 курсу ФІТ

Група ІПЗ-14

Бойко Костянтин Богданович

Викладач: Бичков Олексій Сергійович

Київ-2022

1. Умова задачі

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.

2. Аналіз задачі

Проаналізувавши умову задачі, ми вирішуємо використовувати знання та алгоритми, наведених у лекціях та завданнях на практику. Для двох типів даних буде ідентичний алгоритм пошуку, щоб побачити різницю у виконанні. Також для наглядності будемо використовувати таймер, щоб побачити за скільки часу буде виконуватись певний алгоритм.

3. Структура основних вхідних та вихідних даних

На вхід ми будемо подавати цілочисельні типи даних. Для масивів та лінійних зв’язних списків будемо вводити кількість елементів та діапазон значень цих елементів. А для алгоритмів пошуку будуть надходити готовий масив/лінійний зв’язний список та число, яке потрібно перевірити на наявність та було введене з клавіатури. На виході ми отримуємо інформацію на наявність елемента, а для массива ще і його індекс. Також буде виводитись час виконання алгоритму пошуку.

4. Алгоритм розв’язання задачі

Пошук перебором. Для реалізації цього алгоритму, нам потрібно реалізувати грубу силу, а саме - повний перебір масиву чи лінійного зв’язного списка. Переглядаємо кожен елемент, порівнюємо його із шуканим значенням і, якщо елемент дорівнює шуканому значенню, то ми зберігаємо індекс цього елемента. Алгоритм завершується тоді, коли ми знайшли шуканий елемент, або, коли весь масив/список пройдено і збігу не виявлено.

Пошук із бар’єром. Для цього алгоритму ми удосконалюємо попередній пошук перебором. А саме, ми спрощуємо логічну умову, зробивши бар’єм додатковий елемент, який буде мати шукане значення. Алгоритм буде швидшим, якщо існує гарантія , що співпадіння рано чи пізно відбудеться.

Бінарний пошук. Для цього алгоритму у нас буде додаткова інформація, а саме впорядковані дані. Сам алгоритм заключається в тому, щоб розділити елементи навпіл і, якщо середній елемент буде дорівнювати шуканому значенню, то пошук закінчується. Якщо ж ні, то порівнюємо це середнє значення із шуканим і, якщо середнє значення менше, то воно стає лівою границею, а якщо більше, то стає правою границею. Ці дії повторюються доти, поки не буде знайдено шуканий елемент, або коли цього значення не буде знайдено.

Бінарний пошук згідно з правилом золотого перерізу. Цей алгоритм відрізняється від стандартного бінарного пошуку тільки тим, що ми не ділимо елементи навпіл, а ділимо на золоте число, яке має значення $(1+\sqrt{5})/2$, що приблизно дорівнює ⅝. Алгоритм теж завершиться, коли число, отримане при ділені, буде дорівнювати шуканому, або коли цього значення не буде знайдено.

5. Текст програми

Текст готової програми буде викладений на GitHub.

Посилання: https://github.com/KostiaBoiko/Labs-ASD

6. Набір тестів

Для початку, проведемо тестування для масиву.

Згенеруємо один масив і задамо для кожного алгоритму той самий елемент.

```
Масив
Введіть розмір масиву
500
Введіть діапазон значень елементів
100
Масив згенеровано
```

Пошук перебором.

```
Пошук перебором
Введіть шуканий елемент
56
Елемент знайдено з індексом 145
Витрачено часу: 00:00:00.0001892
Витрачено часу в мілісекундах: 189208
```

Пошук із бар’єром.

```
Пошук з бар’єром
Введіть шуканий елемент
56
Елемент знайдено з індексом 145
Витрачено часу: 00:00:00.0000282
Витрачено часу в мілісекундах: 28291
```

Бінарний пошук.

```
Бінарний пошук
Масив відсортували.

Введіть шуканий елемент
56
Елемент знайдено з індексом 296
Витрачено часу: 00:00:00.0000722
Витрачено часу в мілісекундах: 72250
```

Бінарний пошук згідно з правилом золотого перерізу.

```
Бінарний пошук за золотим перерізом
Масив відсортували.

Введіть шуканий елемент
56
Елемент знайдено з індексом 291
Витрачено часу: 00:00:00.0000370
Витрачено часу в мілісекундах: 37042
```

Збільшимо кількість елементів у масиві та діапазон значень.

```
Масив
Введіть розмір масиву
100000
Введіть діапазон значень елементів
5000
Масив згенеровано
```

Пошук перебором.

```
Пошук перебором
Введіть шуканий елемент
3678
Елемент знайдено з індексом 4367
Витрачено часу: 00:00:00.0001547
Витрачено часу в мілісекундах: 154708
```

Пошук із бар’єром.

```
Пошук з бар’єром
Введіть шуканий елемент
3678
Елемент знайдено з індексом 4367
Витрачено часу: 00:00:00.0000437
Витрачено часу в мілісекундах: 43791
```

Бінарний пошук.

```
Бінарний пошук
Масив відсортували.

Введіть шуканий елемент
3678
Елемент знайдено з індексом 73656
Витрачено часу: 00:00:00.0000363
Витрачено часу в мілісекундах: 36375
```

Бінарний пошук згідно з правилом золотого перерізу.

```
Бінарний пошук за золотим перерізом
Масив відсортували.

Введіть шуканий елемент
3678
Елемент знайдено з індексом 73661
Витрачено часу: 00:00:00.0000746
Витрачено часу в мілісекундах: 74625
```

Тепер проведемо тестування для лінійних зв'язних списків. Тут також згенеруємо список з елементами та будемо шукати одне і те ж саме значення.

```
Лінійний зв'язний список

Ініціалізація списку
Введіть кількість елементів списку
500
Введіть діапазон значень елементів
100

Список згенерований
```

Пошук перебором.

```
Пошук перебором

Введіть шуканий елемент
69
Елемент знайдено
Витрачено часу: 00:00:00.0001817
Витрачено часу в мілісекундах: 181792
```

Пошук із бар'єром.

```
Пошук з барєром

Введіть шуканий елемент
69
Елемент знайдено
Витрачено часу: 00:00:00.0011019
Витрачено часу в мілісекундах: 1101959
```

Бінарний пошук.

```
Бінарний пошук

Введіть шуканий елемент
69
Елемент знайдено
Витрачено часу: 00:00:00.0010319
Витрачено часу в мілісекундах: 1031917
```

Бінарний пошук згідно з правилом золотого перерізу.

```
Бінарний пошук за золотим перерізом

Введіть шуканий елемент
69
Елемент знайдено
Витрачено часу: 00:00:00.0012602
Витрачено часу в мілісекундах: 1260250
```

Збільшимо кількість елементів та діапазон значень

```
Лінійний зв’язний список

Ініціалізація списку
Введіть кількість елементів списку
100000
Введіть діапазон значень елементів
5000

Список згенерований
```

Пошук перебором.

```
Пошук перебором

Введіть шуканий елемент
2679
Елемент знайдено
Витрачено часу: 00:00:00.0002268
Витрачено часу в мілісекундах: 226875
```

Пошук із бар’єром.

```
Пошук з барєром

Введіть шуканий елемент
2679
Елемент знайдено
Витрачено часу: 00:00:00.0000723
Витрачено часу в мілісекундах: 72375
```

Бінарний пошук.

```
Бінарний пошук

Введіть шуканий елемент
2679
Елемент знайдено
Витрачено часу: 00:00:00.0032543
Витрачено часу в мілісекундах: 3254375
```

Бінарний пошук згідно з правилом золотого перерізу.

```
Бінарний пошук за золотим перерізом

Введіть шуканий елемент
2679
Елемент знайдено
Витрачено часу: 00:00:00.0047842
Витрачено часу в мілісекундах: 4784209
```

7. Результати тестування програми та аналіз отриманих помилок.

Виконавши тестування для двох типів даних ми можемо зробити висновок, що алгоритми пошуку для масивів є значно швидшими, в порівнянні з лінійними зв’язними списками.

Провівши тестування для масивів ми бачимо, що для малої кількості елементів найшвидшим є пошук з бар’єром, а на другому місці - бінарний пошук за золотим перерізом. А коли збільшили кількості елементів то бачимо, що на першому місці у нас звичайний бінарний пошук, хоча в минулому випадку він був на третьому місці, а на другому - пошук з бар’єром. Отже, можна вважати що для малої кількості елементів

найкращим буде алгоритм пошуку з бар’єром, а для великої кількості елементів - алгоритм стандартного бінарного пошуку.

Тепер переходимо до лінійних зв’язних списків. Спостерігаємо, що для малої кількості елементів у нас найшвидшим є стандартний бінарний пошук. Хоча, різниця в часі між пошуком з бар’єром та бінарним за золотим перерізом є невеликою, всі значення дуже близькі між собою. Але, коли збільшили кількість елементів то бачимо, що на першому місці, з великим відривом, стоїть пошук з бар’єром, а на другому місці - пошук перебором. Робимо висновок, що для малої кількості елементів найбільш ефективним є бінарний пошук, а для великої - пошук з бар’єром.

При тестуванні програми жодних помилок не було виявлено.