

Objectives

- Search strategies

Mar 31, 2010

Sprenkle - CSCI111

1

Any Questions About Lab 10?

Mar 31, 2010

Sprenkle - CSCI111

2

Need 5 Volunteers

- No one will get hurt ...

Mar 31, 2010

Sprenkle - CSCI111

3

Find the Card in Your Deck

- Reminder to me: take out the jokers
- Challenge: who can find the card first
 - (Most efficient algorithm)
- Need rest of class to keep searchers honest and help me determine who found the card first

Mar 31, 2010

Sprenkle - CSCI111

4

The Race is On!

- 3 of Hearts
- 2 of Diamonds
- 4 of Clubs
- Queen of Spades
- King of Queens

Mar 31, 2010

Sprenkle - CSCI111

5

Searching for a Playing Card

- Given a deck of cards and a card to find, describe the algorithm for how you would find that card.
 - Present several algorithms (naïve ones too!)
 - Discuss the strengths and weaknesses of each

Mar 31, 2010

Sprenkle - CSCI111

6

Search Using `in` Review

- Iterates through a list, checking if the element is found
- Known as **linear search**
- **Implementation:**

```
def linearSearch(searchlist, key):  
    for elem in searchlist:  
        if elem == key:  
            return True  
    return False
```

value	8	5	3	7
pos	0	1	2	3

What are the strengths and weaknesses of implementing search this way?

Mar 31, 2010

Sprengle - CSCI111

search.py

7

Search Using `in` Review

- **Overview:** Iterates through a list, checking if the element is found
- Known as **linear search**
- **Benefits:**
 - Works on *any* list
- **Drawbacks:**
 - Does not tell us where in the list it is
 - What if wanted to do something to that element?
 - Could implement our own version that returns the position
 - Slow -- needs to check every element of list if the item we're searching for is not in the list

Mar 31, 2010

Sprengle - CSCI111

8

High-Low Game

- I'm thinking of a number between 1-100
- You want to guess the number as quickly as possible (in fewest guesses)
- For every number you guess, I'll tell you if you got it right. If you didn't, I'll tell you whether you're too high or too low

Reminder: write down guesses

Mar 31, 2010

Sprengle - CSCI111

9

High-Low Game/TPIR Clock Game

- I'm thinking of a number between 0-100
- You want to guess the number as quickly as possible (in fewest guesses)
- For every number you guess, I'll tell you if you got it right. If you didn't, I'll tell you whether you're too high or too low

→What is your best guessing strategy?

Mar 31, 2010

Sprengle - CSCI111

10

Strategy: Eliminate Half the Possibilities

- Repeat until find value or looked through all values
 - Guess middle value of possibilities
 - If match, found!
 - Otherwise, find out too high or too low
 - Modify your possibilities
 - Eliminate the possibilities from your number and higher/lower, as appropriate
- Known as **Binary Search**

Mar 31, 2010

Sprengle - CSCI111

11

Searching...

value	-3	0	0	1	2	7	8	9
pos	0	1	2	3	4	5	6	7

Use algorithm to search for key = 8

Mar 31, 2010

Sprengle - CSCI111

12

Searching for 8

-3	0	0	1	2	7	8	9
0	1	2	3	4	5	6	7

- Find the middle of the list
 - Positions: 0-7, so mid position is $((7+0)/2) = 3$
- Check if the key equals the value at mid (1)
 - If so, report the location
- Check if the key is higher or lower than value at mid
 - Search the appropriate half of the list

8 > 1, so look in upper half

low mid high

Mar 31, 2010 Sprenkle - CSCI111 13

Searching for 8

- mid is 5 $((7+4)/2)$, list[5] is 7

8 > 7, so look in upper half

low mid high

Mar 31, 2010 Sprenkle - CSCI111 14

Searching for 8

- mid is 5 $((7+4)/2)$, list[5] is 7
 - 8 > 7, so look in upper half
- mid is 6 $((7+6)/2)$, list[6] is 8
 - 8 == 8, FOUND IT at position 6!

What if searched for 6 instead of 8?

Mar 31, 2010 Sprenkle - CSCI111 15

Searching for 6

-3	0	0	1	2	7	8	9
0	1	2	3	4	5	6	7

- Will follow same program flow, but 6 is not in the list
- mid is 6, list[5] is 7
 - 6 < 7, so will try to look in lower half of the list
- mid is 4, list[4] is 2
 - 6 > 2, so will try to look in upper half of the list, but we've already determined it's not there.
 - How do we know to stop looking?**

Mar 31, 2010 Sprenkle - CSCI111 16

Implementation Group Work

```
def search(searchlist, key):
    """Pre: searchlist is a list of integers in sorted order. Returns the position of key (an integer) in the list of integers (searchlist) or -1 if not found"""
```

- Trace through your program using examples
 - Start simple (small lists)
 - Do what the program says *exactly*, not what you *think* the program says

Mar 31, 2010 Sprenkle - CSCI111 17

One Solution

```
def search(searchlist, key):
    low=0
    high = len(searchlist)-1
    while low <= high :
        mid = (low+high)/2
        if searchlist[mid] == key:
            return mid # return True
        elif key > searchlist[mid]:
            low = mid+1
        else:
            high = mid-1
    return -1 # return False
```

If you just want to know if it's in the list

search2.py

Mar 31, 2010 Sprenkle - CSCI111 18

Binary Search

- Example of a **Divide and Conquer** algorithm
 - Break into smaller pieces that you can solve
- Benefits:
 - Faster to find elements (especially with larger lists)
- Limitations:
 - Requires that data can be compared
 - `__cmp__` method implemented by the class
 - List **must** be sorted before searching
 - Takes time to search

Mar 31, 2010

Sprenkle - CSCI111

19

Empirical Study of Search Techniques

Goal: Determine which technique is better under various circumstances

- How long does it take to find various keys?
 - **Measure** by the number of comparisons
 - Vary the size of the list and the keys
 - What are good tests for the lists and the keys?

Mar 31, 2010

Sprenkle - CSCI111 [search_compare.py](#) 20

Empirical Study of Search Techniques

- Analyzing Results ...
 - By how much did the number of comparisons for *linear search* vary?
 - By how much did the number of comparisons for *binary search* vary?
- What conclusions can you draw from these results?

Mar 31, 2010

Sprenkle - CSCI111 [search_compare.py](#) 21

For Friday

- Broader Issue
 - One of 3 social networking articles
- Lab 10

Mar 31, 2010

Sprenkle - CSCI111

22