

Objectives

- Search strategies

Mar 28, 2012

Sprenkle - CSCI111

1

Lab 10

- Trying to solve a real problem
- Started with designing the solution from a vague specification
- Broke into smaller problems (different classes, different responsibilities)
- Implementing smaller components
- Building to large component

Mar 28, 2012

Sprenkle - CSCI111

2

Lab 10 Discussion

- What is the API for the Person class?
 - How much code did it require?
 - How complex was the code?
- How do the SocialNetwork class and Person class work together?

Mar 28, 2012

Sprenkle - CSCI111

3

APIs

Person

- Person(id)
- str(person)
- getName()
- getNetwork()
- getFriends()
- getNumberOfFriends()
- getId()
- setName(newName)
- setNetwork(newNetwork)
- addFriend(person)

SocialNetwork

- SocialNetwork()
- str(socialNetwork)
- getPerson(id)
- getPeople()
- getUserIds()
- printNetwork()
- addConnection(id1, id2)
- addConnections(filename)
- ...

Mar 28, 2012

Sprenkle - CSCI111

4

Need 5 Volunteers

- No one will get hurt ...

Mar 28, 2012

Sprenkle - CSCI111

5

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 28, 2012

Sprenkle - CSCI111

6

The Race is On!

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

Mar 28, 2012

Sprenkle - CSCI111

7

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 28, 2012

Sprenkle - CSCI111

8

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 28, 2012

Sprenkle - CSCI111

search.py

9

Linear Search

- **Overview:** Iterates through a list, checking if the element is found
- **Benefits:**
 - Works on *any* list
- **Drawbacks:**
 - Slow -- needs to check each element of list if the element is not in the list

Mar 28, 2012

Sprenkle - CSCI111

10

High-Low Game/TPIR Clock Game

- I'm thinking of a number between 1-100
- You want to guess the number as quickly as possible, i.e., 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 28, 2012

Sprenkle - CSCI111

11

High-Low Game/TPIR Clock Game

- I'm thinking of a number between 1-100
- You want to guess the number as quickly as possible, i.e., 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 28, 2012

Sprenkle - CSCI111

12

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 28, 2012

Sprenkle - CSCI1111

13

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 28, 2012

Sprenkle - CSCI1111

14

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

				2	7	8	9
				4	5	6	7

8 > 1, so look in upper half

Mar 28, 2012

Sprenkle - CSCI1111

15

Searching for 8

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

2	7	8	9
4	5	6	7

8 > 7, so look in upper half

Mar 28, 2012

Sprenkle - CSCI1111

16

Searching for 8

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

2	7	8	9
4	5	6	7

8 > 7, so look in upper half

- mid is 6 $((7+6)/2)$, list[6] is 8

8	9
6	7

8 == 8, FOUND IT at position 6!

What if searched for 6 instead of 8?

Mar 28, 2012

Sprenkle - CSCI1111

17

Searching for 6

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

- Will follow same execution flow, but 6 is not in the list

- mid is 6, list[5] is 7

2	7	8	9
4	5	6	7

6 < 7, so will try to look in lower half of the list

- mid is 4, list[4] is 2

2
4

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 28, 2012

Sprenkle - CSCI1111

18

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 28, 2012

Sprenkle - CSCI1111

19

One Solution

Cutting list in half
Discuss tradeoffs

```
def altBinarySearch(searchlist, key):
    # Base Case: ran out of elements in the list
    if len(searchlist) == 0:
        return NOT_FOUND

    low = 0
    high = len(searchlist)-1
    mid = (low+high)//2

    valueAtMid = searchlist[mid]
    if valueAtMid == key:
        return mid
    if low == high:
        return NOT_FOUND

    if searchlist[mid] < key: # search upper half
        return altBinarySearch(searchlist[mid+1:], key)
    else: # search lower half
        return altBinarySearch(searchlist[:mid], key)
```

Creating a new list
Unnecessary memory use

Mar 28, 2012

Sprenkle - CSCI1111 search_divide.py

20

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

Mar 28, 2012

Sprenkle - CSCI1111

search2.py

21

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
 - `__lt__`, `__eq__` methods implemented by the class
 - List **must** be sorted before searching
 - Takes time to sort beforehand

Mar 28, 2012

Sprenkle - CSCI1111

22

Exam 2 Results

	A	B	C	Total
Median	80	81	89	83
Average	76	77	83	81

- Common issues
 - Identifying data types (int, str, dictionary, list)
 - Tracing functions, describing what they do
 - Formal, actual parameters
 - What code outputs

Mar 28, 2012

Sprenkle - CSCI1111

23

Reading from a File

```
delFile = open("delegates.dat", "r")
total = 0
for delegates in delFile:
    total += delegates
delFile.close()
print("The total number of delegates earned is", total)
```

Mar 28, 2012

Sprenkle - CSCI1111

24

What is the pattern?

```
MAX = 8
x = 1
a = 0
while x <= MAX:
    print("x is", x)
    a += x
    x += 3
print("a is", a)
```

Mar 28, 2012

Sprenkle - CSC111

25

Sustainability Initiative

```
def main():
    numAttend = eval(input("Enter the number of attendees: "))

    numBuses = numAttend//75
    if numAttend % 75 > 0:
        numBuses += 1
    busPerPerson = numBuses * 200 / numAttend

    numCars = numAttend//4
    if numAttend % 4 > 0:
        numCars += 1
    carPerPerson = numCars * 20 / numAttend

    if busPerPerson < carPerPerson:
        print("The bus is more cost-effective")
    elif carPerPerson < busPerPerson:
        print("The car is more cost-effective")
    else:
        print("It's a toss up.")
```

Refactoring:
Identify functionality
for calculating the per-
person cost of a
vehicle

Mar 28, 2012

Sprenkle - CSC111

26

Sustainability Initiative

```
def main():
    numAttend = eval(input("Enter the number of attendees: "))

    busPerPerson = calcCost(numAttend, 75, 200)
    carPerPerson = calcCost(numAttend, 4, 20)

    if busPerPerson < carPerPerson:
        print("The bus is more cost-effective")
    elif carPerPerson < busPerPerson:
        print("The car is more cost-effective")
    else:
        print("It's a toss up.")

def calcCost(numAttend, capacity, cost):
    numVehicles = numAttend//capacity
    if numAttend % capacity > 0:
        numVehicles += 1
    perPerson = numVehicles * cost / numAttend
    return perPerson

main()
```

main()

For Friday

- Broader Issue
 - FB's NewsFeed
- Lab 10

Mar 28, 2012

Sprenkle - CSC111

28