Searching

CSE 142, Summer 2003

Computer Programming 1

http://www.cs.washington.edu/education/courses/142/03su/

Readings and References

- Reading
  - » Sections 13.1 through 13.3, *Intro to Programming and Object-Oriented Design Using Java*, Niño and Hosch

Searching a List

- Assume that we've got a list, and some collection of strings has been added to the list

```
ArrayList names = new ArrayList( );
names.add("frog");
names.add("rabbit");
names.add("aardvark");
```

- Problem: Look for a name in the list
  - » If found, report its position
  - » If not found, report -1

Linear Search

- Locate a string in a list
- We can do this!
  - » how can we look at each element in turn?
  - » how do we check if it's what we want?
  - » what do we do when we get it?

Linear Search implementation

Can we do better?

- How much work does linear search do?
- Can we do it faster?
  - » No, if we don't know anything about the order of elements in the list
  - » Yes, if the list is sorted

Binary Search – Informal

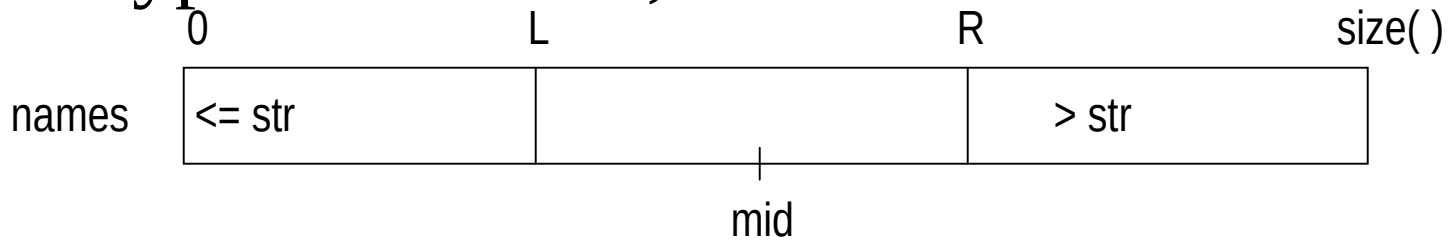
- If the list is sorted, we can use that knowledge to speed up how we search
  - » think about the phonebook - do you do a linear search when looking up a name?
- Idea
  - » Look in the middle of the list
  - » If we haven’t found what we’re looking for, we can ignore half of the list and look at the other half

Binary Search – Goal

- Goal (more formally)
  - » Want to find the point in the list such that everything to the left is <= the string we're searching for and everything to the right is >
- Picture:

Binary Search – Strategy

- On a typical iteration, we have



- Idea:
  - » Let mid = (L+R)/2
  - » If names.get(mid) <= str, move L
  - » If names.get(mid) > str, move R

String Comparisons

- We need to compare Strings to determine ordering, not just equality
  - » Can't use <, <=, etc. on objects
- Solution: method compareTo in class String

  `s.compareTo(t)`

  returns

  negative integer if **`s`** is before **`t`** alphabetically

  zero if **`s`** is equal to **`t`** alphabetically

  positive integer if **`s`** is after **`t`** alphabetically

A binary search implementation

```
public Object findItem(Comparable key) {
    int low = 0;
    int high = theList.size()-1;

    while (low <= high) {
        int mid = (low + high) / 2;
        Object midVal = theList.get(mid);
        int cmp = ((Comparable)midVal).compareTo(key);

        if (cmp < 0)
            low = mid + 1;
        else if (cmp > 0)
            high = mid - 1;
        else
            return midVal; // key found
    }
    return null;  // key not found
   }
```

Binary Search – Performance

- Is the extra complexity worth it?
- How much work is done to search a list of a given size?
- or, How big a list can be searched with *n* comparisons?

Binary & Linear Search Compared

- Linear search: work ~ size
- Binary search: work ~ $\log_2$ size
  - » This is a *fundamental* difference – not just a constant speedup.
- Graph: