L16: Queues and Binary Trees
Alex Steiger
CompSci 201: Spring 2024
3/6/2024

Announcements, Coming up

- Today, Wednesday 3/6
  - APT 6 (sorting problems) due
  - Project P4: Autocomplete released
  - APT 7 out soon, due 3/29 (week after exam)

- Friday 3/8
  - Fill out the midsemester course survey
  - No discussion, enjoy spring break!

- Wednesday 3/20
  - Midterm 2
  - Practice exams available this evening on Canvas

Project 4 Autocomplete

- How to create something like:

![Autocomplete screenshot]

- All about two things:
  - Searching for all words that match on a prefix, and...
  - Sorting them by how common they are,
  - Return these words to show in the GUI above
Midterm 2

- 60 minutes, in-class
- Multiple choice + short answer
- 1 double-sided reference sheet (8.5"x11")
- Extra credit if >70% midsemester survey completion rate
- Grade replaced by Final Exam Part 2

- Lectures up to **Monday + Binary Search today**
  - Stacks/queues/trees not on exam
  - All projects and APTs through this week

Midsemester Survey

Today’s Agenda

1. Binary Search

2. Stack, Queue, PriorityQueue: API perspective
   - Stack/Queue we already know how to implement
   - PriorityQueue later

3. Binary (Search) Tree
Binary Search

- Given a *sorted list* of N elements and a *target* value, return:
  - Index i such that list.get(i) equals target, or
  - -1 if target not in list

- Example:
  - If we search for 'h', should return 4
  - If we search for 'c', should return -1

---

Java API Binary Search

*Arrays.binarySearch* (for arrays) and  
*Collections.binarySearch* (for Lists).

```java
String[] ar = {"ape", "bird", "cat", "dog", "elephant", "ferret", "gecko", "hippo"};
int index = Arrays.binarySearch(ar, "cat");  // Returns 2
```

Careful, assumes input is sorted (and does not verify)!

```java
String[] ar = {"cat", "ape", "bird"};
int index = Arrays.binarySearch(ar, "cat");  // Returns -4
```
Java API Binary Search with Comparator
Can pass a comparator $\text{comp}$, in which case:
1. Array/List should be sorted by that $\text{comp}$, and
2. Want an index $i$ with $i$th element $e_i$ has
   $\text{comp}.\text{compare}(e_i, \text{target})==0$.

```java
Comparator<String> comp = Comparator.comparing(String::length);
index = Arrays.binarySearch(ar, "dog", comp);
```

How is Binary Search $O(\log(N))$?
• How to find something in a list of $N$ elements without looping over the list?
• Let $\text{low}$ (initially 0) and $\text{high}$ (initially $N-1$) mark the limits of the active search space.
• Want to cut down the search space by half at each step:

<table>
<thead>
<tr>
<th>value</th>
<th>'a'</th>
<th>'b'</th>
<th>'d'</th>
<th>'g'</th>
<th>'h'</th>
<th>'j'</th>
<th>'k'</th>
<th>'m'</th>
<th>'p'</th>
</tr>
</thead>
<tbody>
<tr>
<td>index</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>low</th>
<th>high</th>
</tr>
</thead>
</table>

• Searching for 'd' in

<table>
<thead>
<tr>
<th>value</th>
<th>'a'</th>
<th>'b'</th>
<th>'d'</th>
<th>'g'</th>
<th>'h'</th>
<th>'j'</th>
<th>'k'</th>
<th>'m'</th>
<th>'p'</th>
</tr>
</thead>
<tbody>
<tr>
<td>index</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

• 'h' > 'd', so need to keep searching in the lower half.
• Set $\text{high} = \text{mid}-1$;
Binary Search in Pictures

• Searching for 'd' in

<table>
<thead>
<tr>
<th>value</th>
<th>'a'</th>
<th>'b'</th>
<th>'c'</th>
<th>'d'</th>
<th>'e'</th>
<th>'f'</th>
<th>'g'</th>
<th>'h'</th>
</tr>
</thead>
<tbody>
<tr>
<td>index</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

mid = (low + high) / 2

• 'b' < 'd', so need to keep searching in the upper half.
• Set low = mid + 1;

mid = (low + high) / 2

• 'd' equals 'd', return mid (2)

Reasoning about Coding Binary Search

• Going to loop while (low <= high)
  • Looping while there is anything left to search

• For correctness, want to maintain the following loop invariant:
  • If the target is in the array/list, it is in the range [low, high]

• At each step, either find the target and return, or...cut [low, high] in half without losing the target
  • Needs sortedness
Iterative Code for DIY Binary Search?

```java
public static <T> int binarySearch(List<T> list, T target, Comparator<T> comp) {
    int low = 0;
    int high = list.size() - 1;
    while (low <= high) {
        int mid = (low + high) / 2;
        T midVal = list.get(mid);
        if (comp.compare(midVal, target) < 0) {
            low = mid + 1;
        } else if (comp.compare(midVal, target) > 0) {
            high = mid - 1;
        } else {
            return mid; // target found
        }
    }
    return -1; // target not found
}
```

What will index be after this call to binary search? * (?)

```java
String[] ar = {'win', 'bird', 'cat', 'dog', 'elephant', 'ferret', 'giraffe', 'hedgehog'};
int index = Arrays.binarySearch(ar, 'ape');
```

- 4
- 0
- 1
- 2

After running this code, index will be... * (?)

```java
String[] ar = {'cat', 'dog', 'dog', 'bird', 'hedgehog', 'elephant'};
int index = Arrays.binarySearch(ar, 'ape', Comparator.comparing(String::length));
```

- -1
- 0
- Can't tell because there are multiple possible correct values
- Can't tell because the elements are not in the correct sorted order
- Can't tell because there are duplicates in the array
Finding the first or last?

- Algorithm we have shown does **not** guarantee to find the first or last such index if there are multiple.

- You will develop versions of binary search in Project 4: Autocomplete that find such indices.
Stacks, Queues, PriorityQueue: API Perspective

Stack Abstract Data Structure: LIFO List

Applications?
Stack in the real world?

- Remember the call stack?
- History on your web browser / back button?
- Depth-first search in a graph (more coming soon!)
java.util.Stack class

• both push and pop are O(1)
  • Adds and removes from end of ArrayList
  • Could also use LinkedList

```
public static void main() {
  String x = "wonderful", y = "compaci"
  Stack<String> st = new Stack();
  for(String s : str) { // str = x+y
    st.push(s);
  }
  while (!st.isEmpty()) {
    System.out.println(st.pop());
  }
}
```

*Actually uses the Vector class (see docs), but for 201 imagine ArrayList

Queue Abstract Data Structure: FIFO List

FIFO = First In First Out

Enqueue: Add element to queue

Dequeue: Remove first in element

Applications?
Queue in the real world?

• Operating system keeps track of which program should get processor time next.
• Waitlist for class registration on DukeHub?
• Many "shortest way to get from X to Y" problems, e.g., breadth-first search in a graph (more coming soon!)
java.util.Queue interface

• Both add and remove are O(1)
  • Add at end of LinkedList
  • Remove from front of LinkedList

java.util.Deque interface

• "Double-ended queue", pronounced "deck"
  • Implemented by LinkedList, which is doubly-linked
  • Add/remove to front/end (head/tail) in O(1) time

java.util.Deque interface

- Comparison of Queue and Deque methods:
  - Summary of Deque methods:
    - First Element (Head) and Last Element (Tail)
    - Insert: offerFirst(), offerLast(), addFirst(), addLast()
    - Remove: removeFirst(), removeLast(), pollFirst(), pollLast()
    - Get: getFirst(), getLast(), peekFirst(), peekLast()

- Comparison of Stack and Deque methods:
  - Schedule: pushFirst(), pushLast(), popFirst(), popLast()
Priority Queue in the Abstract

Queue sorted by priority instead of insertion order.

java.util.PriorityQueue Class

• Kept in sorted order, smallest out first
• Objects must be Comparable OR provide Comparator to priority queue

Complexity of Java’s Priority Queue

<table>
<thead>
<tr>
<th>Method</th>
<th>Behavior</th>
<th>Runtime Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>add(element)</td>
<td>Add an element to the priority queue</td>
<td>O(logN) comparisons</td>
</tr>
<tr>
<td>remove()</td>
<td>Remove and return the minimal element</td>
<td>O(logN) comparisons</td>
</tr>
<tr>
<td>peek()</td>
<td>Return (do not remove) the minimal element</td>
<td>O(1)</td>
</tr>
<tr>
<td>size()</td>
<td>Return number of elements</td>
<td>O(1)</td>
</tr>
</tbody>
</table>
What will be printed by the stackTrace method? Write your answer with no quotes and hyphens between words (as they would appear if printed as below). For example, you might write (though it would not be correct): the-fox-jumps. *(Q)

```java
public static void stackTrace() {
    Stack<String> myStack = new Stack<>();
    String[] words = new String[] {"the", "fox", "jumps"};
    for (String s : words) {
        myStack.push(s);
    }
    System.out.print("" + myStack.peek());
    System.out.print("" + myStack.pop());
    System.out.print("" + myStack.peek());
    System.out.print("" + myStack.pop());
}
```

jumps-jumps-over

What will be printed by the queueTrace method? Write your answer with no quotes and hyphens between words (as they would appear if printed as below). For example, you might write (though it would not be correct): the-fox-jumps. *(Q)

```java
public static void queueTrace() {
    Queue<String> myQueue = new LinkedList<>();
    String[] words = new String[] {"the", "fox", "jumps"};
    for (String s : words) {
        myQueue.offer(s);
    }
    System.out.print("" + myQueue.peek());
    System.out.print("" + myQueue.poll());
    System.out.print("" + myQueue.peek());
    System.out.print("" + myQueue.poll());
}
```

the-the-fox

What will be printed by the pqTrace method? Write your answer with no quotes and hyphens between words (as they would appear if printed as below). For example, you might write (though it would not be correct): the-fox-jumps. *(Q)

```java
public static void pqTrace() {
    PriorityQueue<String> myPQ = new PriorityQueue<>();
    String[] words = new String[] {"the", "fox", "jumps"};
    for (String s : words) {
        myPQ.offer(s);
    }
    System.out.print("" + myPQ.peek());
    System.out.print("" + myPQ.poll());
    System.out.print("" + myPQ.peek());
    System.out.print("" + myPQ.poll());
}
```

fox-fox-jumps
The k largest elements of values

What is the asymptotic runtime complexity of the getK method as a function of $N = \text{values.length}$ and $k$?

```java
public static int[] getK(int[] values, int k) {
    PriorityQueuedIntegers pq = new PriorityQueuedIntegers();
    for (int value : values) {
        if (pq.size() < k) {
            pq.add(value);
        } else if (pq.peek() < value) {
            pq.remove();
            pq.add(value);
        }
    }
    int[] result = new int[k];
    for (int i = 0; i < k; i++) {
        result[i] = pq.remove();
    }
    return result;
}
```

$N$ iters, $O(\log k)$ time/iter. ⇒ $O(N \log k)$

How else might you find $k$-smallest without PQ? Sort then return first $k$ items ⇒ $O(N \log N)$ time. PQ helps!

Binary Trees
Comparing TreeSet/Map with HashSet/Map

**TreeSet/Map**
- $O(\log(N))$ add, contains, put, get are **not amortized**.
- Stored in sorted order
  - Natural ordering by default, can provide Comparator
  - Can get range of values in sorted order efficiently

**HashSet/Map**
- $O(1)$ add, contains, put, get, are amortized.
- Unordered data structures
- Cannot get range efficiently, stored unordered

TreeNode to store Strings

```
public class TreeNode {
    TreeNode left;
    TreeNode right;
    String info;
    TreeNode(String s, TreeNode llLink, TreeNode rLink){
        info = s;
        left = llLink;
        right = rLink;
    }
}
```

APT TreeNode to store ints

APT TreeNode will only hold integer. Would need to create another class to hold Strings? Another for...

```
public class TreeNode {
    int info;
    TreeNode left;
    TreeNode right;
    TreeNode(int y){
        info = y;
    }
    TreeNode(int x, TreeNode llNode, TreeNode rLink){
        info = x;
        left = llNode;
        right = rLink;
    }
}
```