Some Practice Problems

1. With one or two well-chosen phrases, express the essential idea(s) of each item below:

   (a) java.util.Collection
   (b) java.util.ArrayList
   (c) java.util.LinkedList
   (d) implements Iterable
   (e) stack
   (f) queue
   (g) deque (aka dequeue)
   (h) circular list
   (i) heap property
   (j) Θ(n log n)
   (k) interface
   (l) concrete class
   (m) Randomized Partition
   (n) Quicksort
   (o) worst-case linear-time algorithm for Select
   (p) Iterator vs. iterator
   (q) Comparative sort
2. For this heap:

<table>
<thead>
<tr>
<th>index</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>200</td>
<td>100</td>
<td>75</td>
<td>50</td>
<td>90</td>
<td>60</td>
<td>65</td>
<td>28</td>
<td>21</td>
<td>70</td>
</tr>
</tbody>
</table>

(a) Draw the corresponding binary tree.

(b) What does the heap look like after the 200 is removed? **Draw the array**, not the tree:

(c) Starting from there, what does the heap look like after 78 is inserted? Again, **draw the array**, not the tree:

(d) Is this a min (i.e., minimum) heap? Explain:

\[
\begin{array}{ccccccccccc}
\text{index} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
\text{data} & 1 & 2 & 3 & 4 & 8 & 28 & 12 & 40 & 30 & 50 & 51 & 13 & 27 \\
\end{array}
\]

(e) Here’s a max heap:

\[
\begin{array}{ccccccccccc}
\text{index} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
\text{data} & 93 & 50 & 40 & 20 & 10 & 30 & 19 & 5 & 18 \\
\end{array}
\]

What would it look like after we remove one item from it?

(f) Start with that same heap (in b, immediately above).
What would it look like after inserting a 52?
3. Characterize each of these recurrences using big-theta notation:

(a) \( T(n) = \text{if } n = 1 \text{ then } 1 \text{ else } 2 \ T(n/2) + 3 \)

(b) \( T(n) = \text{if } n = 1 \text{ then } 1 \text{ else } 5 \ T(n/5) + n^2 \)

(c) \( T(n) = \text{if } n = 1 \text{ then } 1 \text{ else } 9 \ T(n/3) + n^2 + 12n + 5 \)

(d) \( T(n) = \text{if } n = 1 \text{ then } 1 \text{ else } 2 \ T(n - 1) \)

(e) \( T(n) = \text{if } n = 1 \text{ then } 1 \text{ else } 16 \ T(n/10) + 100 \) [estimate the value of the ugly exponent]

(f) \( T(n) = \text{if } n = 7523 \text{ then } 1 \text{ else } 7523 \ T(n/7523) + 7523n + 7523 \)

(g) \( T(n) = 1 + n^3 \)
4. More drills:

(a) Compute $\log_3 (1/10)$

(b) Compute $\log_{10} 2500$

(c) Compute $\log_5 2500$

(d) A certain queue, $q$, contains the letters A B C D E. After we execute $q.insert('F')$, it looks like A B C D E F. What will it look like after we execute $q.insert(q.remove())$?

(e) A certain stack, $s$, contains the letters A B C D E. After we execute $s.push('F')$, it looks like A B C D E F. What will it look like after we execute $s.push(s.pop())$?

(f) Convert 42513 base six to base nine:

(g) Consider the largest possible number you can make in base eight with one hundred digits. Approximately how many digits will it have (1) in base two and (2) in base ten?

(h) Among these numbers, which is the median? You have two minutes:

<table>
<thead>
<tr>
<th>256</th>
<th>618</th>
<th>19</th>
<th>241</th>
<th>81</th>
<th>567</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>497</td>
<td>311</td>
<td>224</td>
<td>3</td>
<td>12</td>
<td>990</td>
</tr>
<tr>
<td>34</td>
<td>723</td>
<td>162</td>
<td>47</td>
<td>551</td>
<td>83</td>
<td>257</td>
</tr>
<tr>
<td>67</td>
<td>402</td>
<td>2</td>
<td>264</td>
<td>45</td>
<td>807</td>
<td>56</td>
</tr>
<tr>
<td>401</td>
<td>49</td>
<td>73</td>
<td>237</td>
<td>841</td>
<td>201</td>
<td>27</td>
</tr>
<tr>
<td>33</td>
<td>68</td>
<td>719</td>
<td>5</td>
<td>441</td>
<td>654</td>
<td>123</td>
</tr>
<tr>
<td>392</td>
<td>32</td>
<td>478</td>
<td>461</td>
<td>59</td>
<td>452</td>
<td>78</td>
</tr>
</tbody>
</table>

(i) As succinctly as possible, sketch the remarkable $\Theta(n)$ worst-case selection algorithm that we presented in class and explain how it accomplishes its linear time bound:

(j) The first line of java.util.ArrayList begins basically like this:

```java
public class ArrayList extends AbstractList
    implements List...
```

What’s going on here? What’s become of Collection? And, Iterable?
5. The iterator for

    public class FunnyList extends java.util.ArrayList

    will deliver the items with indices 1, 0, 3, 2, 5, 4, 7, 6, etc.

    For this class, implement

    public java.util.Iterator iterator() :


6. Is System.out.println() a classwide method or an instance method? Explain:

7. Is it legal to take input from String[] args and from standard input in the same program? Explain: