ICS 161 — Algorithms — Spring 2002 — Goodrich — First Midterm

Name:

ID:

1:

2:

3:

4:

5:

total:

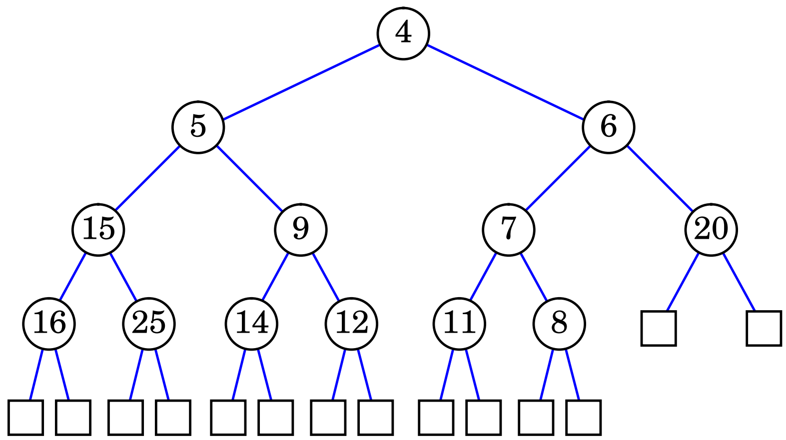
1. (50 points). Please define each of the following terms:

(a) $f(n)$ is $O(g(n)$.

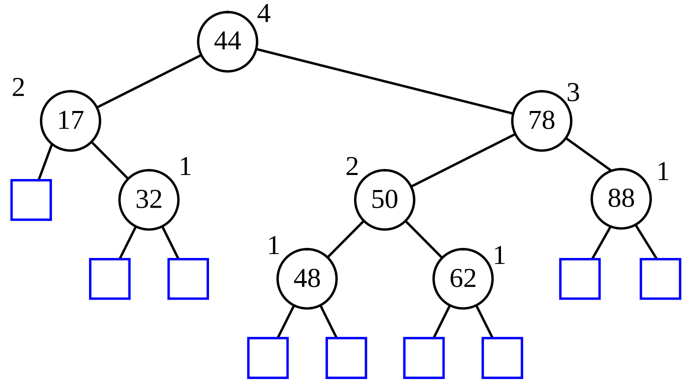
(b) depth of a node v in a tree T.

(c) stable sorting algorithm.

2. (50 points). Draw the heap that results from performing a removeMin() operation on the following heap:



3. (50 points). Draw the AVL tree resulting from inserting an item with key 46 into the following AVL tree:



4. (50 points). The following questions all deal with sorting.

(a) Suppose that we change quick-sort to always pick the first element in the input sequence as the pivot. State what is the worst-case running time of this version of quick-sort and give an example input that would require this version of quick-sort to take this much time.

(b) Is merge-sort an in-place sorting algorithm? Why or why not?

(c) Name an algorithm discussed in class that would have the fastest worst-case running time for sorting a set of n items with integer keys in the range $[0, 2^n - 1]$? You may assume that integer comparisons and array indexing both take $O(1)$ time. (Do not give any pseudo code for this—just say which known sorting algorithm you would use to do this.) What is the running time of your method?

5. (50 points). The LowExpectations dating service has an array W of n women that it wants to match up with an array M of n men. It assigned each man in M a unique number between 0 and $n-1$ (in random order). The company is then doing the matching of women to men by using following algorithm:

> **for** $i \leftarrow 0$ to $n-1$ **do**
> Let $a \leftarrow W[i]$.
> Scan down the array M to find the man b assigned number i.
> Output "a's date is b."
> **end for**

(a) What is the worst-case running time of the dating service's algorithm?

(b) Briefly explain how you could do the same kind of matching the LowExpectations service is doing, but in worst-case optimal time. (You may use any algorithm studied in class as a subroutine.) What is the running time of your algorithm?