Name: ________________________

SS#: ________________________

Problem	Max. Marks	Obtained
1	15	
2	20	
3	15	
4	10	
5	10	
6	15	
7	15	
Total	100	

Remarks

- You can use any of the algorithms covered in the class without describing it.
- In the last two questions you have to justify (at least informally) why your algorithms work.

Problem 1: [15 marks] Let $f(n)$ and $g(n)$ be two monotonically increasing functions. Prove or disprove (by giving a counterexample) the following statements (you cannot simply quote the textbook here):

(i) $f(n) = O(g(n))$ implies $g(n) = O(f(n))$.

(ii) $f(n) = O(g(n))$ implies $g(n) = \Omega(f(n))$.

Problem 2: [20 marks] Write the solution to the following recurrences using the big-Oh notation. You **do not** have to derive/prove your solutions.

(i)

$$T(n) = \begin{cases} 8T(n/2) + n^2 & n \geq 2 \\ 1 & n < 2. \end{cases}$$

(ii)

$$T(n) = \begin{cases} T(\sqrt{n}) + 1 & n > 2 \\ 1 & n \leq 2. \end{cases}$$

Problem 3: [15 marks] Prove **one** of your solutions to the previous problem, using the **induction** method.

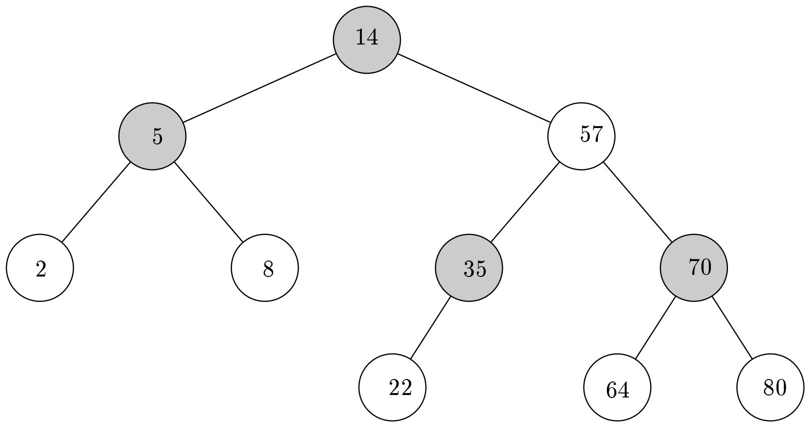
Problem 4: [10 marks] What are the minimum and maximum heights of a

(i) binary search tree,

(ii) red-black tree,

each storing n keys.

Problem 5: [10 marks] Consider the following red-black tree (shaded nodes are *black* and unshaded nodes are *red*). How does the tree look like after inserting 60 into it, using the insertion algorithm described in the book? Redraw the tree.



Problem 6: [15 marks] Let S be a set of n integers. Describe an $O(n^2 \log n)$-time algorithm to determine whether there exist three integers $a, b, c \in S$ (a, b, c need not be distinct) such that $a + b + c = 0$.
(**Hint:** *Notice that sorting n^2 items takes $O(n^2 \log n)$ time.*)

Problem 7: [15 marks] Let S be a list of n (not necessarily distinct) integers. Describe an $O(n)$-time algorithm to test whether any item occurs more than $\lceil n/2 \rceil$ times in S. (**Hint:** *Use the selection algorithm.*)