

This is a closed-book test. You may not use calculators, books, or notes during the exam. (No problem involves long messy computations if you approach it right.)

Please write all of your answers on the answer sheet, and write your name and ID number, as well as the test version, on both sides of the answer sheet. This is version A. You may keep the exam.

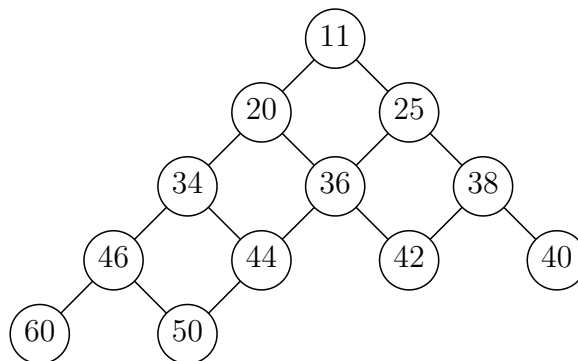
Use your time wisely; it might be a good idea to read over the entire midterm before starting. Feel free to ask if anything is unclear.

The maximum possible score is 50 on Part I and 35 on Part II, for a total of 85.

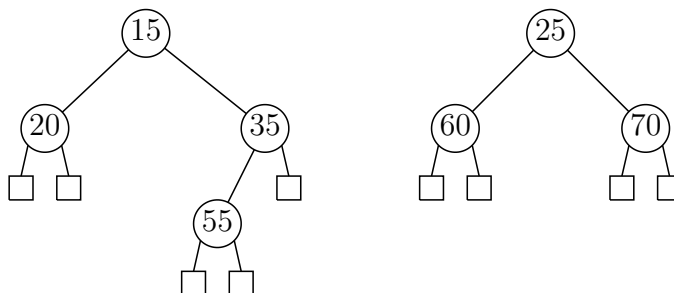
### Part I. Written answers

Each question in this part is worth 10 points. The notation “explanation optional” on some problems means that a completely correct answer without explanation will get full credit, though an explanation could conceivably help you get partial credit.

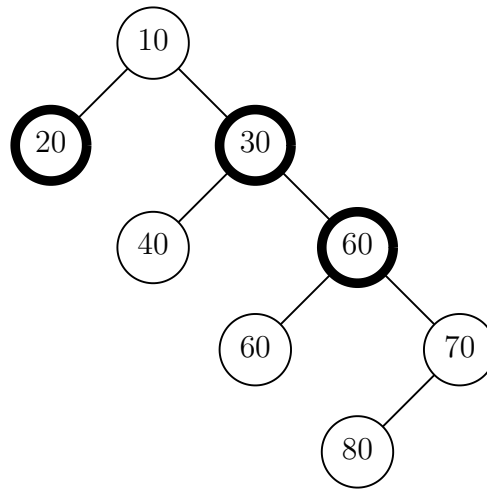
- [Explanation optional] Show the result of inserting a 15 into the beap below.



- [Explanation optional] Suppose that we are going to meld the two leftist heaps shown below.
  - Show the intermediate result when we have just merged the rightmost paths.
  - Show the final result after we have also walked the rightmost path from the leaf back to the root, switching left and right subtrees as needed.



3. [Explanation optional] Show the result of doing a DecreaseKey operation on the Fibonacci heap below, changing the key 70 to 48. I've indicated marked nodes by giving them a thicker circle. Please be sure to indicate which nodes are marked in your answer.



4. Suppose we defined a variation of a heap which allowed the following four operations: MakeHeap( $c$ ), InsertConst, ExtractMin, and DecreaseKey. As in class, ExtractMin removes and returns the minimum key, and DecreaseKey decreases the value in a node (assuming we are given a pointer to that node). MakeHeap( $c$ ) creates an empty heap and sets its *default key* to  $c$ ; this value can never be changed for this heap. InsertConst inserts a new node whose key has the default value set when the heap was created, and returns a pointer to the node.

Show that it is not possible to design an algorithm which works by doing comparisons and performs each of the above operations in amortized  $O(1)$  time.

5. Suppose we define a hypothetical data structure consisting of a single integer  $i$  in the range 1 to  $n$ , with  $n$  being even, on which just two operations are possible: Increment and Decrement. Initially  $i = n/2$ . Each Increment costs one time unit and increases  $i$  by 1; however, if the value of  $i$  reaches  $n$ , it is set back to  $n/2$  at an additional cost of  $n/2$  time units. Similarly, each Decrement costs one time unit and decreases  $i$  by 1; if the value of  $i$  reaches 0, it is set back to  $n/2$  at an additional cost of  $n/2$  time units.

Use a potential function argument to show that the amortized cost of each operation is at most 2. Explain clearly, including a clear specification of your potential function. You may use without proof that fact that if a potential function is initially zero, and never goes negative, then we can obtain a valid amortized cost for each operation by adding its actual cost to the change it causes in the potential function.

## Part II. Multiple choice.

For each question, choose the *best* answer and write its letter clearly as a large plain capital on the answer form. Each question is worth 5 points, and there is no penalty for guessing.

It's a good idea to read each question carefully and look at all the choices. (Sometimes they may be in a different order than you would expect.)

1. Which of the statements below is true? (Use the formal definition of  $O$ -notation.)

- A.  $n^2 + 3n + 4 \in O(n^3)$
- B.  $n^2 + 3n + 4 \in \Omega(n^3)$
- C.  $n^9 \in O\left(\left(\frac{8}{7}\right)^n\right)$
- D. A and C
- E. All of the above
- F. None of the above.

2. Let  $b_k$  denote the Catalan numbers, i.e.,

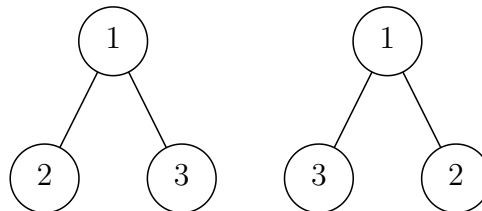
$$b_k = \frac{1}{k+1} \binom{2k}{k}.$$

Which of the following grows most rapidly for large  $n$ ?

- A.  $b_n$
- B.  $n^2$
- C.  $(\log n)^n$
- D.  $n^{\log n}$

3. The two trees shown below are equivalent if we are considering them as

- A. free trees
- B. rooted trees
- C. ordered trees
- D. binary trees
- E. A or B



4. Let  $X$  be a random variable which assumes values in  $\{1, 2, 3, 4\}$ . Which statement below is true?

- A.  $\mathbf{E}[\log X] \leq \log(\mathbf{E}[X])$ .
- B.  $\log(\mathbf{E}[X]) \leq \mathbf{E}[\log X]$ .
- C. We cannot conclude that either of A or B holds.

5. In class we discussed several implementations of heaps. Which of the choices below is most appropriate (from a theoretical time complexity standpoint) if we only care about the total cost of the entire sequence of operations to be done, and we know that the number of DecreaseKey operations will be much greater than the number of any other type of operation?

- A.  $d$ -heaps
- B. Binomial heaps
- C. Fibonacci heaps

6. What was (theoretically) an *advantage* of a beap over an AVL tree as an implementation of a dictionary?
- A. The beap had better worst-case access times.
  - B. The beap was an implicit data structure.
  - C. The beap had better worst-case deletion times.
  - D. The beap had better worst-case insertion times.
7. Suppose that we define a modified tree in the same way as we defined Red-Black trees, except that now we allow a red node to be a parent of another red node, provided that on any path from a root to a leaf we never see three red nodes in a row. Which choice below gives a correct recurrence for the minimum number  $n_k$  of nodes in such a tree, if the tree has black height  $k$ ?
- A.  $n_k = 2n_{k-1} + 1$
  - B.  $n_k = 2n_{k-1} + 3$
  - C.  $n_k = n_{k-1} + n_{k-2} + 1$
  - D.  $n_k = n_{k-2} + n_{k-3} + 1$
  - E.  $n_k = n_{k-2} + n_{k-3} - 1$

Please be sure you wrote your name, ID, and test version (A) on both sides of the answer form.