

This is a closed-book test. You may not use calculators, books, or notes during the exam.

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.

I suggest you look over the entire midterm before starting.

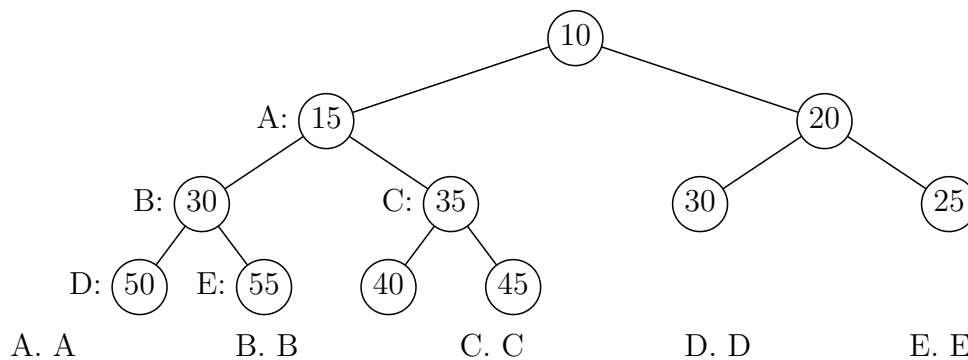
The maximum possible score is 44 on Part I and 55 on Part II. You get one extra point for free, for a total of 100.

### Part I. 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 4 points, and there is no penalty for guessing.

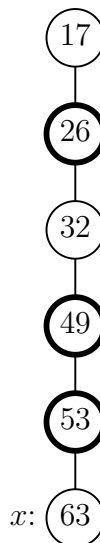
It's a good idea to read each question carefully and look at all the choices. (Sometimes they may be in a different order than you would expect.)

- Below is a 2-heap (i.e., a  $d$ -heap with  $d = 2$ ) of height 3. If we delete the 15 where will the key 45 appear in the resulting heap?



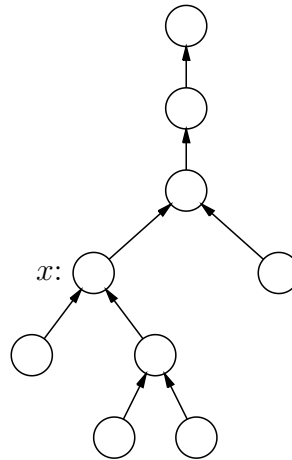
- Suppose we have a 3-heap (i.e., a  $d$ -heap with  $d = 3$ ) containing 22 items. What is the height of this tree?  
 A. 1                  B. 2                  C. 3                  D. 4                  E. 5
- Suppose that the root of some binomial tree (note that I did not say binomial heap) has 7 children; it follows that the tree has 128 nodes. What is the last digit of the number of leaves in the tree? (Each choice gives two digits; pick the choice that includes the correct answer.)  
 A. 0 or 5                  B. 1 or 6                  C. 2 or 7                  D. 3 or 8                  E. 4 or 9

4. Suppose we have a binomial heap containing 107 items. How many roots does the heap have? (I.e., how many separate binomial trees are in the heap?)
- A. 6                      B. 5                      C. 4                      D. 3                      E. 2
5. When we do a DeleteMin from a Fibonacci heap, we remove the node containing the minimum value, add the children of the deleted node to the list of roots, and then perform a consolidate operation which involves linking roots of equal degree until all roots remaining have distinct degrees. Suppose that just before the consolidate operation there are 6 roots in the list, with degrees 1,1,2,3,4,6. After the consolidate, how many roots will there be?
- A. 6                      B. 5                      C. 4                      D. 3                      E. 2
6. Suppose that the path from the root to some node  $x$  in a Fibonacci heap is as shown below; marked nodes are indicated by bold circles. Then we perform a DecreaseKey to lower the value of the key in  $x$  to 15. When the operation has completed, by how much will the number of roots in the top-level tree list have increased?



- A. 1                      B. 2                      C. 3                      D. 4                      E. 5
7. Suppose that we make an empty Fibonacci heap and then perform  $n$  Insert,  $n$  DeleteMin, and  $n^3$  DecreaseKey operations (not necessarily in that order). What is the total amount of time that would be used in the worst case?
- A.  $\Theta(n)$               B.  $\Theta(n \log n)$       C.  $\Theta(n^2)$               D.  $\Theta(n^3)$               E.  $\Theta(n^3 \log n)$

8. Suppose we are working with a Union-Find data structure, as described in class and the text. The tree representing one of the sets is shown below. If we perform a Find on  $x$ , how many children will the root of this tree have afterwards? (Assume we are using the path compression rule.)

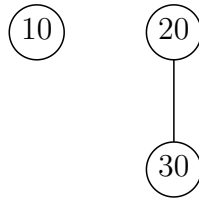


- A. 2                      B. 3                      C. 4                      D. 5                      E. 6
9. Which expression below gives an exact formula for  $\lg^* 2^{(2^n)}$ , assuming that  $n$  is a positive integer?
- A.  $\lg^*(n^2)$             B.  $1 + \lg^* n$             C.  $2 + \lg^* n$             D.  $2 \lg^* n$             E.  $(\lg^* n)^2$
10. Suppose that we are working with the lowest common ancestors problem discussed in class, and that  $x$  and  $y$  are two nodes in the tree  $T$ . The inorder numbers of  $\phi(x)$  and  $\phi(y)$  are (in binary) 1010101 and 1001001. What is the inorder number of the node in  $B$  that is the lowest common ancestor of  $\phi(x)$  and  $\phi(y)$ ? (See the Appendix if you want a reminder of some of the concepts and terms we used. For this and the next problem I haven't checked that these values can actually occur; just give the results the algorithm would compute given these values.)
- A. 1100000            B. 1100101            C. 1101001            D. 1010000            E. 1001000
11. Continuing the previous problem, assume that, in binary,  $\text{ASCENDANT}(x) = 1001101$  and  $\text{ASCENDANT}(y) = 1111101$ . What is the inorder number of the node in  $B$  that is the image, under  $\phi$ , of the lowest common ancestor of  $x$  and  $y$ ?
- A. 1010100            B. 1011000            C. 1010000            D. 1100000            E. 1000000

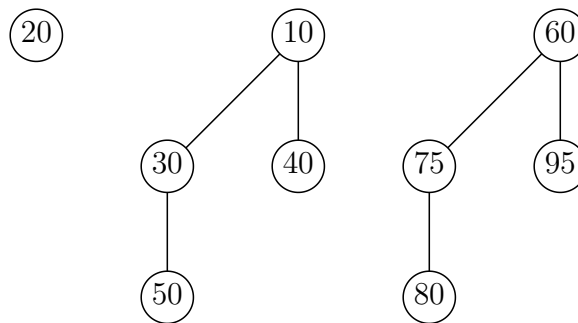
## Part II. Written answers

Point values for each question in this part are given in square brackets after the question number. The notation “explanation optional” on some problems means that a completely correct final answer will get full credit, though an explanation might help you get partial credit.

12. [6—Explanation optional] Below is a binomial heap consisting of two binomial trees. Show the binomial heap that results if we insert the value 15.

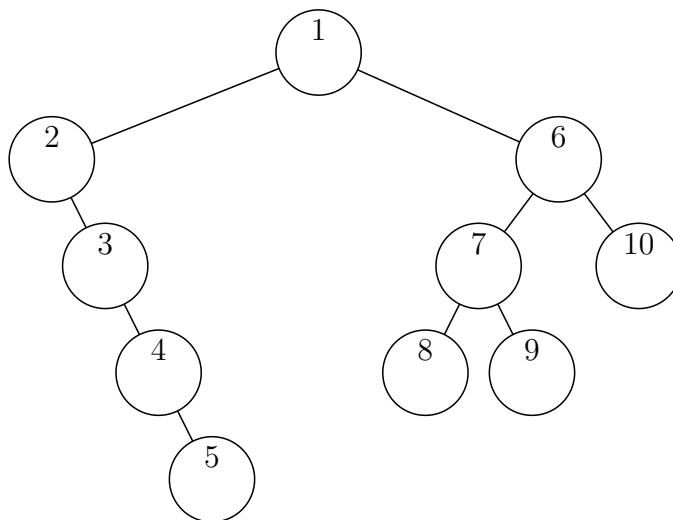


13. [10—Explanation optional] Show the result of deleting the minimum from the Fibonacci heap shown below. (Since Fibonacci heaps are not ordered, I will disregard the order of the trees and subtrees when grading this.)



14. [12—Explanation optional] Below is a tree with its nodes numbered in preorder. Suppose we wish to perform the LCA algorithm on this tree.

- a) [8] Give the value of INLABEL for each node in the tree. (Write the INLABEL numbers below the preorder numbers on the tree on the answer sheet.)
- b) [4] Give the value of ASCENDANT for just the nodes whose preorder numbers are 5 and 10.



15. [12] Suppose that we are given a string  $A = a_1a_2 \cdots a_n$  and an integer  $m < n$ . Assume we have code to build a suffix tree in linear time. Give an algorithm that runs in time  $O(n)$  and finds a substring  $x$  of length  $m$  in  $A$  that occurs most frequently. (In other words,  $x$  is a substring of  $A$  of length  $m$  and no other string of length  $m$  occurs more often as a substring of  $A$ .) If you define any node fields that are not standard in a suffix tree, explain how they would be computed.

(Note: Someone pointed out to me that the solution given in class for the problem of finding the longest repeated substring assumed two substrings were distinct if they started at different positions, even if they overlapped. Your solution should also assume this.)

16. [15] Suppose that we modify the standard Fibonacci heap algorithm by saying that we do not perform a Consolidate after each DeleteMin; instead, we only perform a Consolidate after every other DeleteMin. That is, if we perform a number of DeleteMin operations  $d_1, d_2, d_3, d_4, \dots$ , perhaps mixed with other operations, we perform a Consolidate only after  $d_2, d_4, \dots$ . Prove that the amortized times for Insert, DeleteMin, and DecreaseKey (shown on slide 5-40) still hold, by modifying the potential function argument given on the slides.

Make your potential function be one which, like the one in the original argument, takes on only integer values. The value of the potential function may depend on the state of the data structure and also on whether an even or odd number of DeleteMin operations have occurred so far. You don't have to repeat the whole argument—just clearly describe the new potential function and the changes that are needed in the argument. In particular, for each of Insert, DeleteMin, and DecreaseKey, discuss the actual time used by the operation, the changes that occur in your new potential function, and the resulting amortized time bound.

To help you remember the analysis of Fibonacci heaps, copies of some of the slides about Fibonacci heaps are included. You may refer to these in your answer.

## Appendix

Here is a reminder of some facts and definitions about the data structure we gave for finding lowest common ancestors.

We said that the inorder number of a node  $x$  in  $B$ , when expressed in binary, gave the labels on the edges from the root to  $x$ , followed by a 1, followed by enough 0s to pad to a length of  $(1 + \text{the height of } B)$ .

Each vertex  $x$  in  $T$  had the following fields:

PREORDER( $x$ ): The preorder number of  $x$ .

LEVEL( $x$ ): The distance from  $x$  to the root.

INLABEL( $x$ ): The most even PREORDER value of any descendant of  $x$  (including  $x$  itself). Let  $\phi(x)$  be the node in  $B$  whose inorder number is INLABEL( $x$ ).

ASCENDANT( $x$ ): A bit string  $\dots h_3h_2h_1h_0$ , where  $h_i$  is 1 iff for some ancestor  $y$  of  $x$  in  $T$ ,  $\phi(y)$  has height  $i$  in  $B$ .

**Notes written at the exam**

Problem 16. Where it says “discuss discuss”, that should of course just be “discuss”. This problem is fairly hard—I don’t expect that most people will get it right.

Problem 2. (If you’re not sure how I’m defining the height of a  $d$ -heap, see problem 1 for an example.)

Problem 15. Assume the suffix tree is compressed.