Computer Science 302 Fall 2008 Practice Final Examination

Name:_____

This practice examination is longer than the real final examination will be. If you can work all the problems here, you will probably be prepared for the final examination.

This practice examination is 500 points.

1. True or False. [5 points each]

   (a) _____ The time to heapsort an array of n items is $\Theta(n \log n)$.

   (b) _____ Open hashing uses open addressing.

   (c) _____ In the decision tree model of computation, the time complexity of any algorithm to sort n items is $\Omega(n \log n)$.

   (d) _____ The height of a binary tree with n nodes is $\Omega(\log n)$.

   (e) _____ A binary search tree is commonly used to represent unfulfilled obligations.

   (f) _____ Quicksort takes $O(n \log n)$ time to sort an array of n items.

   (g) _____ Given the choice between two algorithms, one of which takes $O(n)$ time and the other of which takes $O(n^2)$ time, it is *always* best to choose the one which takes $O(n)$ time.

   (h) _____ The built-in function `random` is an excellent choice for a hash function.

2. [5 points] What are the smallest and largest possible number of nodes that a 2-3 tree with height 3 could have? _____ and _____.

3. [10 points] What implementation of the ADT search structure would you use if the expected number of items in the structure is 1?

4. [10 points each] For each of the following code fragments, express the asymptotic time complexity by choosing the best of the following answers: $O(n)$, $O(n^2)$, $O(n \log n)$, $O(\log n)$, $O(\log \log n)$, $\Theta(n)$, $\Theta(n^2)$, $\Theta(n \log n)$, $\Theta(\log n)$, $\Theta(\log \log n)$

(a)
```
for (int i = 0; i < n; i++)
  cout << "Hi there.";
```

(b)
```
for (int i = 0; i < n; i = 2*i+1)
   cout << "Hi there.";
```

(c)
```
for (int i = 0; i < n; i++)
  for (int j = i; j > 0; j = j/2);
  cout << "Hi there.";
```

(d)
```
for (int i = 0; i < n; i++)
  for (int j = i; j > i/2; j = j/2);
    cout << "Hi there.";
```

(e)
```
for (int i = 0; i < n; i = i*i+1)
   cout << "Hi there.";
```

(f)
```
for (int i = 0; i < n; i++)
 {
  int j = unknown(i);
// unknown is a function whose value could be anything: we have no clue!
// It takes one step to evaluate unknown(i).
  if (i < j)
    i = j;
  cout << "Hi there.";
 }
```

5. [30 points]

(a) Describe the meaning of the word *collision* as used in discussions of hashing.

(b) How are collisions handled in closed hashing?

(c) How are collisions handled in open hashing?

6. [10 points] What implementation of the ADT search structure would you use if n items are to be inserted at once at the beginning of the program, there will be no further inserts, and find will be executed n^2 times during the running of the program? (There is more than one correct answer to this problem, as well as several inferior answers.)

7. [20 points] Explain how you would implement a sparse array using a search structure. Do **not** give any details whatsoever about the search structure itself, since that's not the point of this question.

8. [20 points] Walk through the steps of the stack algorithm used to evaluate the following postfix (reverse Polish) expression, showing the stack at each step. (Hint: there will be 9 illustrations of the stack.)

```
5 6 + 3 * 2 3 * -
```

9. [30 points] Describe each of the following types of search. (Be sure to say what the structure is that is being searched in each case.)

   (a) Linear search.

   (b) Binary search.

10. [20 points] Draw "before" and "after" figures illustrating *left rotation*.

11. [10 points] Suppose an $5 \times 6 \times 8$ array A is stored in column-major order, with base address 1000, and one word is required for each entry. What is the address of A[3,2,6]? Assume that array indices start at 0, as in C++.

12. [20 points] Write pseudocode for the array implementation of the ADT "stack of integer." Your code should include procedures that implement **pop**, **push**, and **empty**.

13. [20 points] Explain how you would insert and delete from a queue, given that you are using singly linked nodes in a circular linked list implementation. Draw pictures.

14. [25 points] Explain how insertion works in a 2,3-tree. Hint: the phrase "node splitting" or the equivalent must be in your explanation.

15. [30 points] Given the following:

```
class BST{ // Binary Search Tree
  public:
   BST(int); // initializes item field to parameter, links to 0
   void static inorderWrite(BST*); // all items to standard ostream inorder
   void static insert(int, BST*&); // inserts parameter, if not there
   bool find(int, BST*); // parameter is in the binary search tree
  private:
   int item; // the value stored in the node
   BST * left; // pointer to the left subtree
   BST * right; // pointer to the right subtree
 };
```

Complete the following code by writing **exactly three lines:**

```
    void BST::inorderWrite(BST * t){
       if (t != 0){
// Your three lines go here.




        }
     }
```

16. [30 points] If you use *Heapsort* to sort an array, the first step is to *heapify* the array. Given the following array, show the steps of that heapification, assuming that your final array will be sorted in alphabetical order from left to right. The number of rows in the table below may or may not be the correct number of steps; you might not use all the rows, or you might have to add more rows.

A	L	G	O	R	I	T	H	M

17. [30 points] Suppose you are writing a dynamic programming algorithm to find the minimum weight path between a given source vertex S and a given target vertex T in a weighted directed acyclic graph G.

    (a) Describe the subproblems.

    (b) In what order would you work the subproblems?

18. [30 points] Use the Floyd-Warshall algorithm to solve the all-pairs shortest path problem for the weighted directed graph shown in Figure 1. Show your work.

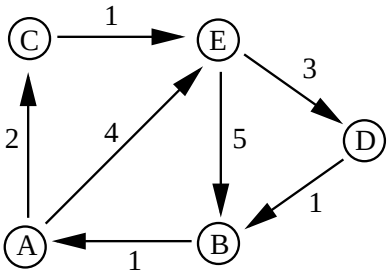

Figure 1: A weighted directed graph for problems 18 and 19.

19. [30 points] Use Dijkstra's algorithm to solve the single source shortest path problem for the weighted directed graph shown in Figure 1, where **A** is the source. Show your work.

20. [30 points] Use polyphase mergesort to sort the following list: A,L,G,O,R,I,T,H,M,S,R,F,U,N. Show all steps.