RULES:

1. Don’t start the exam until the instructor says to.

2. This is a closed-book, closed-notes, no-calculator exam. Don’t refer to any materials other than the exam itself.

3. Write your name, and your login name, on each page of the exam when you get to it. Think carefully about each question. Before you turn in your exam, make sure you have all the pages. The last page is blank and can be used as scratch paper.

4. Do not look at anyone else’s exam. Do not talk to anyone but an exam proctor during the exam. If you’re wearing a billed cap, please turn it around or take it off. And turn off cell phones, pagers, PDA’s, MP3 players, etc.

5. If you have a question, raise your hand and an exam proctor will come to you.

6. You have until the end of the class period to finish the exam. When you are done, give your exam to a proctor. The proctor will check your picture ID and sign the ID check below.

7. Your exam grade will be emailed to your account this week. Exams will be handed back in lecture next week.

<table>
<thead>
<tr>
<th>#</th>
<th>Type</th>
<th>max pts</th>
<th>actual pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>General TF</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>General MC</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Skip Lists</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Treaps</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Pointers</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>Huffman</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TOTAL</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

grader:______

PROCTOR ID CHECK:______________
1. [23 pts.] (1 pt each correct; -1 pt each incorrect; 0 pt if left blank) True or False:

   a. ______ A treap is a binary tree.
   
   b. ______ A Huffman code tree is a binary trie.
   
   c. ______ In Huffman code tree, there are never more internal nodes than leaves.
   
   d. ______ The height of a randomized search tree is always less than 2N, where N is the number of leaves in the tree.
   
   e. ______ In a red-black tree with N nodes, any path from the root to another node contains no more than 2 \( \log_2(N+1) + 2 \) nodes.
   
   f. ______ In a red-black tree, every path from the root to a null reference contains the same number of red nodes.
   
   g. ______ Considered as a tree, a heap always satisfies the AVL balance property.
   
   h. ______ In an AVL tree, the levels of any two leaves can differ by at most 1.
   
   i. ______ Every subtree of a treap is itself a treap.
   
   j. ______ Every subtree of a heap is itself a heap.
   
   k. ______ Every subtree of an AVL tree is itself an AVL tree.
   
   l. ______ Red-black trees and binary heaps have the same big-O worst-case time cost for insert operations.
   
   m. ______ Skip lists and randomized search trees have the same big-O worst-case time costs for find operations.
   
   n. ______ In a skip list, the values of the priorities associated with keys that have been inserted determine which key is in the first node of the list.
   
   o. ______ An AVL double rotation consists of two AVL single rotations.
   
   p. ______ Any AVL rotation in a BST always produces a BST.
   
   q. ______ The C++ class hierarchy is a tree whose root is the \texttt{Object} class.
   
   r. ______ AVL trees are considered to be more difficult to implement than randomized search trees.
   
   s. ______ In C++, an iterator for the \texttt{std::set} container will iterate over the elements in the set in sorted order.
   
   t. ______ Because of its \texttt{writeBit()} method, the C++ \texttt{ofstream} class is a good choice when you want to write individual bits to a file.
   
   u. ______ In C++, the \texttt{end()} method of the \texttt{std::vector} class returns an iterator pointing at the last element in the vector.
   
   v. ______ In C++, storage for objects created dynamically using \texttt{new} is automatically reclaimed by the garbage collector when the objects can no longer be accessed in your program.
   
   w. ______ In C++, “*” is a pointer dereference operator.
2. [21 pts] (3 pts each) Multiple choice. Write the capital letter of the best answer in the blank provided.

a. _____ Using the usual probabilistic assumptions, the average case number of comparisons for successful search in a binary search tree vs. a randomized search tree (RST) with the same number of nodes is
   A. much worse than RST  B. slightly worse than RST  C. the same as RST
   D. much better than RST  E. slightly better than RST

b. _____ Suppose T is a completely filled binary search tree with 7 nodes. The worst-case number of comparisons for a successful find operation in T is
   A. 1  B. 2  C. 3  D. 6  E. 7

c. _____ Suppose T is a completely filled binary search tree with 7 nodes. The best-case number of comparisons for a successful find operation in T is
   A. 1  B. 2  C. 3  D. 4  E. 7

d. _____ Suppose T is a binary search tree with 7 nodes, only one of which is a leaf. Suppose all keys are equally likely. The integer closest to the average-case number of comparisons for a successful find operation in T is
   A. 1  B. 2  C. 3  D. 4  E. 7

e. _____ Suppose T is a completely filled binary search tree with 7 nodes. Suppose all keys are equally likely. The integer closest to the average-case number of comparisons for a successful find operation in T is
   A. 1  B. 2  C. 3  D. 6  E. 7

f. _____ The maximum number of distinct symbols codeable with a Huffman code tree with 7 nodes is
   A. 3  B. 4  C. 5  D. 7  E. undetermined

g. _____ Suppose symbols P, Q, R appear in an input file (other symbols may appear as well). Which of the following can be a Huffman code for the sequence RQP:
   A. 111110  B. 001110  C. 101010  D. Any of these  E. None of these
3. [18 pts] The following integer keys were inserted in this sequence into an initially empty skip list:
   8, 3, 1, 6, 4, 7

   In doing these insertions, the following sequence of node levels (i.e., number of forward pointers) was generated by the skip list’s random level generator:
   3, 1, 2, 4, 1, 1

   Answer the following multiple-choice questions about the resulting skip list by writing the letter of the answer in the space provided. Assume 0-based indexing of pointers within nodes. The possible answers are these:

   - A: the node containing key 1
   - B: the node containing key 3
   - C: the node containing key 4
   - D: the node containing key 6
   - E: the node containing key 7
   - F: the node containing key 8
   - G: the skip list header node
   - H: null
   - I: error! There is no pointer with that description.

   a. The skip list header has an index 2 pointer pointing at: __________
   b. The node containing key 1 has an index 1 pointer pointing at: __________
   c. The node containing key 3 has an index 3 pointer pointing at: __________
   d. The node containing key 4 has an index 0 pointer pointing at it from: __________
   e. The node containing key 8 has an index 1 pointer pointing at it from: __________
   f. The node containing key 6 has an index 3 pointer pointing at: __________
4. [16 pts total] In the following treaps, keys are characters, with alphabetic ordering; priorities are integers.

a. [8 pts] Consider the Treap #1a as the treap used internally by a randomized search tree (RST). A client of this RST now inserts key A into the RST, and the RST implementation’s random number generator generates a priority of 30 for this key. Show the resulting Treap #2a after this insert operation is complete:

![Treap #1a](image1)

![Treap #2a](image2)

b. [8 pts] Usually one assumes all keys in a treap to be equally likely to be searched for, but if some keys are less likely than others, moving them farther from the root can improve average case search times. The operation used to implement this is the “priority lowering” operation, in which a node’s priority is decreased and rotations are performed as needed to restore the treap properties. Suppose key H was identified in Treap #1b, and the priority of the node containing H was decreased from 50 to 20. Show the resulting Treap #2b after the priority lowering operation is complete:

![Treap #1b](image3)

![Treap #2b](image4)
5. [12 pts total] Consider these C++ declarations. After some computation, the current state of memory is as shown. Null pointers are shown as 0; non-null pointers are shown with an arrow indicating what object they point to.

```cpp
class Node {
    public:
        Node* l;
        Node* r;
    }

Node* x;
Node* y;
```

**current state of memory:**

![Memory Diagram]

a. [6 pts] Starting with the current state of memory shown above, consider the C++ code shown below. In the space to the right, draw the state of memory after this code executes.

```cpp
y = x->l;
x->l = y->r;
y->r = x;
```

![Memory Diagram after code execution]

b. [6 pts] Again starting with the current state of memory shown above at the top of the page, consider the C++ code shown below. In the space to the right, draw the state of memory after this code executes.

```cpp
y = x->l->r;
x->r->l = y;
x->l->r = 0;
```

![Memory Diagram after code execution]
6. [10 pts. total] Suppose an information source emits symbols E, G, L, T, O with these probabilities:

E: 0.15  
G: 0.15  
L: 0.10  
T: 0.10  
O: 0.50

a. [6pts] In the space above, draw a Huffman code tree for this information source. Label edges ‘1’ or ‘0’. Label each leaf node with its symbol.

b. [4 pts] According to your code tree, write down the sequence of bits that codes this message: GETGOOGL

(The number of bits required may be less than the 3 bytes shown.)
(blank paper)