S-Q 1998 - Final Exam
August 12, 1998

Name:

Email:

|   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | Total |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|------|
|   | 12| 8 | 10| 15| 15| 15| 10| 10| 10| 10  | 5   | 10 | 10 | 10  | 150   |

Test Guidelines

- **Do not begin the test until you are told to do so.**

- **This test is closed-book and closed-notes.**

- **Answer the questions that you find easiest first.** There are 14 questions on the test. The total number of points on the test is 150, and the point value of each question corresponds roughly to the number of minutes you should spend on each question. It is very likely that some of the questions will take you more or less time to answer than our estimates; do not get bogged down on one part of the test.

- **Show your work.** An incorrect answer that is based on a sound line of reasoning and shows clear understanding (but fails to be correct because of a relatively minor error) may receive more credit than an unsubstantiated statement that happens to be correct.

- **Do not feel compelled to fill every square inch of space with writing.** There should be plenty of space below each question. If necessary, use the back of the previous page. If this is still not sufficient, ask for scrap paper.
1. (12 points) State whether each of the following statements is true, or can be false, and explain your reasoning.

**True  False**  The depth of a binary tree containing $n$ nodes is $\Omega(\log n)$.

**True  False**  Finding the second largest integer in an unsorted array of $n$ integers requires approximately $2n$ comparisons.

**True  False**  Dijkstra’s algorithm can fail if the network contains an edge that has a negative cost.

**True  False**  If an algorithm is $O(n^2)$, then executing it with an input size of $2n$ will take approximately 4 times longer than executing it with an input size of $n$. 

2. (8 points) Provide a **brief** proof to show why each of the following statements is true.

(a) If we use breadth-first search to find the path between two connected vertices in a graph, the first path the search finds will be a shortest path.

(b) Heapsort (using the ordinary heapsort algorithm from the reading) for an array of \( n \) integers is \( O(n) \) if all the numbers being sorted are identical.
3. (10 points)

(a) Briefly state the \texttt{selectionsort} algorithm.

(b) Show the swaps that would be performed by the \texttt{selectionsort} algorithm as it sorts the following array.

\begin{verbatim}
1, 4, 10, 2, 6, 0, 9, 8, 3, 7
\end{verbatim}
4. (15 points)

(a) Wendy has a large number of records that she would like to write to file. Each record consists of a fixed number of fields, and each field is a text string of arbitrary length. Describe two simple but different methods that Wendy could use. (Be sure to label which is which!) For each scheme, approximately how much space will the file require in addition to the space required to store the strings themselves?
(b) (Continued from previous page.)

Imagine that Wendy wishes to store the data in such a manner that it is possible to locate the $i$th record in the file very rapidly (for any $i$ less than the number of records in the file). Suggest a way that she can accomplish this. Does it require storing extra information in the file, and if so, approximately how much?

(If your answer to the first part of this question already allows rapid access to specific records, just describe how.)
5. (15 points) Show the results of performing each of the following operations to a hash table of size 10. The hash table is initially empty. The hash table uses open addressing, and uses quadratic probing to resolve collisions. The keys stored in the hash table are integers, and the hash function is simply the key \( \text{mod} \ 10 \). Show the hash table after each operation.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Hash Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSERT(1)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>INSERT(12)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>INSERT(0)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>INSERT(10)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>DELETE(1)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>DELETE(10)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>INSERT(20)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
<tr>
<td>INSERT(21)</td>
<td>0 1 2 3 4 5 6 7 8 9</td>
</tr>
</tbody>
</table>
6. (15 points) Show the steps that Dijkstra’s algorithm would take to find a minimum path from vertex A to vertex B in the network shown below.
7. (10 points) Show the results of performing each of the following operations to the skiplist shown below. (Instead of having you flip a coin many times, please use the heights provided for the INSERT operation.) Show the skiplist after each operation.

- Insert 10 at level 3.
- Insert 2 at level 2.
- Insert 9 at level 1.
- Insert 1 at level 1.
- Delete 8.
8. (10 points) Show the steps that the Knuth-Morris-Pratt algorithm would use to search for the following pattern in the following text. Indicate which characters in the pattern and text are compared, and what each skip is. You do not need to precompute the entire skip array.

Pattern: COCOA
Text: CHOCOLATECOCONUTCOCKTAIL
9. (10 points) Show the steps that the Boyer-Moore algorithm would use to search for the following pattern in the following text. Indicate which characters in the pattern and text are compared, and what each skip is. You do not need to precompute the entire skip array.

Pattern: SAND
Text: DATASTRUCTURESCANANDALGORITHMS
10. (10 points) Compute the following values, using the same method as the Rabin-Karp hash function.
   To help you get started, note that $157,942,387 \mod 7 = 1$, $100,000,000 \mod 7 = 2$, and $1,000,000,000 \mod 7 = 6$.
   To check your work, you can use the fact that $387,523,160 \mod 7 = 3$.
   
   - $579,423,875 \mod 7 =$
   - $794,238,752 \mod 7 =$
   - $942,387,523 \mod 7 =$
   - $423,875,231 \mod 7 =$
   - $238,752,316 \mod 7 =$
11. (5 points) Imagine that you have a random number generator named \texttt{rand100} that produces random, uniformly distributed integers in the range $0 \leq r < 100$.

Give a randomized algorithm that uses \texttt{rand100} to create a random number generator that efficiently creates \textit{truly} uniformly distributed random numbers in the range $0 \leq r < 33$. 
12. (10 points) Imagine that you have a **sorted** array of length $n$, where $n > 2$. The array contains only **two** different values, named $X$ and $Y$ (and $X < Y$), so the array contains $m$ elements with a value of $X$, followed by $n - m$ elements with a value of $Y$. There is always at least one $X$ element and one $Y$ element in the array. Note that the values of $X$ and $Y$ are unknown.

(a) Give an efficient algorithm for determining the number of $X$ elements in the array, given $n$. (You do not need to write your algorithm in pseudo-code; a few sentences should be sufficient.) What is the big-O for your algorithm?

(b) Imagine that the sorted array must contain at least one element with a value of $X$ and one element with a value of $Y$ (as before), but it **might** also contain elements with a third value $Z$, where $X < Y < Z$. (Once again, $X$, $Y$, and $Z$ are unknown.) Give an efficient algorithm for determining whether the array contains only two distinct values, or whether it contains three.
13. (10 points) Agatha has been given a buggy radixsort function, and asked to fix it. The function correctly sorts by each of the digits except the least significant digit, which is completely ignored.

Agatha believes that since the output of the array is sorted by every digit except the least significant, it is in effect “mostly” sorted. Having read in a book that insertionsort is $O(n)$ for arrays that are “mostly” sorted, she decides to fix her broken radixsort by performing an insertionsort of the array after the radixsort is finished.

(a) Is Agatha’s algorithm correct? Will the resulting array be sorted?

(b) Agatha believes that the resulting algorithm is $O(n)$, because radixsort is $O(n)$ and insertionsort is $O(n)$ for arrays that are “almost” sorted. Is she correct? Why or why not?
14. (10 points) In any tree, the *lowest common ancestor* of two distinct nodes $n_1$ and $n_2$ is the node that is an ancestor of both $n_1$ and $n_2$, but has no descendants that are ancestors of both $n_1$ and $n_2$. If either of $n_1$ or $n_2$ are the root of the tree, then their least common ancestor is defined to be **nil**.

Give an algorithm to find the lowest common ancestor of the node containing key $k_1$ and the node containing key $k_2$ in a **binary search tree** with integer keys. You can assume that all of the keys in the tree are unique, and that both $k_1$ and $k_2$ appear in the tree, and that $k_1 \neq k_2$. 