S-Q 1998 - Hourly 2
July 27, 1998

Name:

Email:

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 six questions on the test. The total number of points on the test is 50, 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. (5 points) Complete the following table:

<table>
<thead>
<tr>
<th>Sort</th>
<th>Comparisons (big-O)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Expected</td>
</tr>
<tr>
<td>Bubble sort</td>
<td></td>
</tr>
<tr>
<td>Quick Sort</td>
<td></td>
</tr>
<tr>
<td>Merge Sort</td>
<td></td>
</tr>
<tr>
<td>Insertion Sort</td>
<td></td>
</tr>
<tr>
<td>Selection Sort</td>
<td></td>
</tr>
</tbody>
</table>
2. (5 points) Show the insertions that would be performed by the `insertionsort` algorithm as it sorts the following array.

10, 5, 12, 4, 8, 3, 1, 7
3. (10 points) Bertram is trying to write a bucketsort routine to sort arrays containing 1000 or more integers, where each integer is known to be greater than or equal to 0 and less than 100. Unfortunately, Bertram does not understand bucketsort very well, and has made several mistakes. Find Bertram’s errors, and explain how to fix them.

```c
#define BUCKETS 100

void bucketsort (int *array, int array_length)
{
    int i, j = 0, buckets [BUCKETS];

    for (i = 0; i < BUCKETS; i++) {
        buckets [i] = 0;
    }
    for (i = 0; i < array_length; i++) {
        buckets [array [i]] = 1;
    }
    for (i = 0; i < BUCKETS; i++) {
        if (buckets [i] != 0) {
            array [j++] = i;
        }
    }
}
```
4. (10 points) Show the steps that Kruskal’s algorithm would take to find a minimum spanning tree for the following network.
5. (10 points) Topologically sort the directed graph shown below. If there is more than one possible topological sort, you need only find one. Show your work.

![Directed Graph]

A → B → C → D → E
F → G → I
J
6. (10 points) Short answer:

(a) What is the largest number of keys that can be stored in a B-tree of order 5 and depth 3?

(b) What is the minimum number of keys that can be stored in a B-tree of order 5 and depth 3?

(c) What causes the worst-case behavior of quicksort?

(d) What causes the worst-case behavior of insertionsort?

(e) Why is FIND always $O(k)$, where $k$ is the length of the key, even in a de la Briandais trie, even though the number of nodes we need to search may depend in part on the number of keys in the trie?