1. Consider this frequency table:

<table>
<thead>
<tr>
<th>Letter</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>22</td>
</tr>
<tr>
<td>C</td>
<td>8</td>
</tr>
<tr>
<td>E</td>
<td>28</td>
</tr>
<tr>
<td>N</td>
<td>7</td>
</tr>
<tr>
<td>K</td>
<td>6</td>
</tr>
<tr>
<td>M</td>
<td>12</td>
</tr>
<tr>
<td>U</td>
<td>9</td>
</tr>
<tr>
<td>F</td>
<td>14</td>
</tr>
</tbody>
</table>

(a) (6 pts) Construct an optimal Huffman tree for encoding a document from the table.

(b) (2 pts) Using your tree, how would “FUN” be encoded?

2. (2 pts each) Name the complexity of each of these algorithms/operations.

(a) left-rotate in a red-black tree
(b) red-black tree delete (including fixup)
(c) quicksort (worst case)
(d) heapify
(e) radix sort
(f) mergesort
(g) BFS
(h) Dijkstra’s single-source shortest path algorithm

3. (6 pts) Trace by hand radix sort for this array: \( A = <123, 323, 22, 112, 321, 133, 131, 13, 222> \).

4. (6 pts) Trace counting sort for this array: \( A = <5, 3, 2, 5, 2, 1, 5> \).

5. (2 pts each) Briefly define each of the following terms:

(a) stable sort
(b) degree of a vertex
(c) complete graph

6. (4 pts) List two stable sorts.

7. (8 pts) Suppose a red-black tree is implemented using the following class:

```java
class RBNode {
    int data;
    RBNode left, right, parent;
    char color;
};
```
Write a method called `bheight` which given the root of a red-black tree will return the black height of the tree. Assume that the color field stores a ‘B’ for a black node and an ‘R’ for a red node. Assume the existence of a global variable named `NIL` that is black in color and is used in place of `null` to mark the bottom of the tree. Do not include the `NIL` node in the calculation of black height.

8. Consider the red-black tree given in figure 1.

(a) (3 pts) Show the order in which nodes would be visited in an inorder traversal.

(b) (5 pts) Show the steps required to insert 140. Be sure to redraw the tree after applying each case and be sure to specify which case you apply at each step.

(c) (5 pts) Show the steps required to remove 150. Be sure to specify which cases you apply at each step. Use the original tree as the starting point.

Figure 1: A Red-Black Tree

9. Answer the following questions based on the graph below. When tracing an algorithm, always select the node that comes first in the alphabet if the order doesn’t matter to the algorithm.

(a) (4 pts) Show how the graph could be represented as a weighted adjacency matrix. Use $\infty$ to represent the absence of an edge.

(b) (6 pts) Trace Dijkstra’s shortest path algorithm on the graph. Redraw the graph after every step and indicate the edges selected for inclusion in the shortest path by drawing them bold faced. Start with vertex A.
(c) (4 pts) List order in which vertices in the graph would be visited if the graph were traversed using a DFS starting with vertex A.

10. Suppose the following array represents a heap: < 54, 32, 50, 30, 28, 20, 25, 26, 21, 19 >

(a) (3 pts) Draw the tree representation of this heap.

(b) (4 pts) Illustrate how a heap delete operation would work, redrawing the tree at each step.

(c) (4 pts) Illustrate how a heap insert of the element 40 would work, redrawing at each step. NOTE: Start with the original heap (not with the heap resulting from problem 10b).

11. (Up to 5 BONUS points) A riddle: You are riding in a train along with the tooth fairy, the Easter bunny, and a \( \Theta(n) \) comparison sorting algorithm. There is a $100 bill on the floor. The train goes through a tunnel and darkness engulfs the cab. When the train emerges from the tunnel there is a loud gasp because the $100 bill is gone! Who took the money? Explain.