1. (5) Solve this recurrence using a recursion tree. Provide costs for nodes at each depth and the total cost.
   \[ T(n) = aT(n/b) + f(n) \]
2. (5) Provide a formal definition of little-omega

3. (5) Precisely provide the max-heap property/properties

4. a. (3) Quantify the running time of a lucky vs unlucky partitioning in quicksort.

   b. (2) Provide an example recurrence of each
5. (5) When would we use quicksort over radix sort given radix sort is linear and quicksort is worse than linear?

6. a. (3) What makes a good hash function?

   b. (2) Why is accomplishing this so hard?

7. (5) Provide an algorithm that finds order statistics in expected linear time (assume we are not willing to preprocess the elements to improve worse case performance)
8. (5) In the analysis of open hashing, how do we calculate the expected number of probes? How does this affect search time in the successful case and the unsuccessful case?

9. a. (3) Precisely provide the AVL tree property/properties.

   b. (2) How unbalanced can an AVL tree be?

10. a. (3) What is the difference between top down and bottom up dynamic programming?

    b. (2) Why could one be better than the other?
11. (5) What is the recursive formula for finding the longest common subsequence?

12. (5) Explain what operations are involved with obtaining the running time of $\Theta((2n+m) \log n)$ for Dijkstra’s algorithm.

13. (5) Why is a binomial heap considered a mergeable heap when compared to a binary heap?

14. (5) Define optimal substructure
15. (10) Given an amount and a list of coin denominations, provide concise greedy algorithm to make change by calculating the correct number of each denomination
16. (10) Using the graph below, use Bellman Ford to solve for the single source shortest path from vertex B to the other vertices. Provide the results at the end of each pass and at the end of the algorithm including the weight at each vertices. Use the order and weights of edges in the table.
17. (10) Solve the integer knapsack problem using the items, weight and profits below by using dynamic programming. Show the dynamic programming results in table. Weight of knapsack is 8 units.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profit</td>
<td>12</td>
<td>15</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>Weight</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Include?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Answer: