1. Analysis of algorithms.

(a)  
\[ P \] Printing the keys in a binary search tree in ascending order.

\[ U \] Finding a minimum spanning tree in a weighted graph.

\[ P \] Finding all vertices reachable from a given source vertex in a graph.

\[ P \] Checking whether a digraph has a directed cycle.

\[ P \] Building the Knuth-Morris-Pratt DFA for a given string.

\[ P \] Sorting an array of strings, accessing the data solely via calls to `charAt()`.

\[ I \] Sorting an array of strings, accessing the data solely via calls to `compareTo()`.

\[ I \] Finding the closest pair of points among a set of points in the plane, accessing the data solely via calls to `distanceTo()`.

(b)  
\[ A \] Insert into a red-black tree.  
\[ B \] Insert into a binary heap.  
\[ C \] Insert into a 2d-tree.

\[ A \] \( \log N \) worst case

\[ B \] \( \log N \) amortized

\[ C \] \( \log N \) average case on random inputs

(c)  
- The \( N^3 \) one might be much easier to correctly implement, debug, and test.
- The \( N^3 \) algorithm might be faster for the values of \( N \) of interest (e.g., because of the leading constant).
- The \( N^3 \) algorithm might use less memory.

(d)  
56 bytes.

Each `Point` object consumes 32 bytes (8 bytes for each of the three `double` instance variables; 8 bytes of object overhead).

Each `Node` object consumes 56 bytes (4 bytes for each of the 3 reference instance variables; 4 bytes for the `int` instance variable; 32 bytes for the `Point3D` object; 8 bytes of object overhead).
2. Breadth-first search.
   (a) A B C D E G F H I
   (b) d

3. Minimum spanning tree.
   (a) 1 2 3 5 6 7 8 12
   (b) $w \leq 8$
   (c) 6 1 3 2 5 7 8 12
   (d) Find the unique path between $x$ and $y$ in $T$. This takes $O(V)$ time using DFS because there are only $V - 1$ edges in $T$. We claim the edge $T$ remains an MST if and only if $w$ is greater than or equal to the weight of every edge on the path.
   - If any edge on the path has weight greater than $w$, we can decrease the weight of $T$ by swapping the largest weight edge on the path with $x$-$y$. Thus, $T$ does not remain an MST.
   - If $w$ is greater than or equal to the weight of every edge on the path, then the cycle property asserts that $x$-$y$ is not in some MST (because it is the largest weight edge on the cycle consisting of the path from $x$ to $y$ plus the edge $x$-$y$). Thus, $T$ remains an MST.

4. Shortest paths.
   (a) vertex: A C D F H E B G I
distance: 0 1 12 20 25 28 34 40 53
   (b) $A \rightarrow C$, $C \rightarrow D$, $C \rightarrow B$, $D \rightarrow F$, $F \rightarrow H$, $H \rightarrow E$, $E \rightarrow G$, $G \rightarrow I$

5. Ternary search tries.
   (a) ear fo his hitch hold holdup hotel hum humble ill
   (b)
   (c) • faster, especially for search miss
   • support character-based operations such as prefix match (autocomplete), longest prefix, and wildcard match


<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
7. Regular expressions.

   (a) 5
   b b a b a c a a

   (b) b a b a b a a b a

   I only.

10. Tandem repeats.
    (a) This problem is a generalization of substring search (is there at least one consecutive copy of b within s?) so we need an algorithm that generalizes substring search.

    Create the Knuth-Morris-Pratt DFA for \( k \) copies of b, where \( k = \lfloor N/M \rfloor \). Now, simulate DFA on input \( s \) and record the largest state that it reaches. From this, we can identify the longest repeat.

    (b) \( M + N \).

11. Reductions.
    (a) \( \{ -3M, x_1 + M, x_2 + M, \ldots, x_N + M \} \)

    If we can force any solution to this 4SUM instance to choose \( x_i = -3M \) as one of the integers, then the remaining three integers are \( x_i + M, x_j + M, \) and \( x_k + M \) and we have \( x_i + x_j + x_k = 0 \).

    We force any solution to this 4SUM instance to choose \(-3M\) by choosing \( M = 1 + \max\{|x_1|, |x_2|, \ldots, |x_N|\} \) to be large, thereby making \(-3M\) the only negative integer.

    (b) None.