1. Consider the implementation of a 4-key binary search tree. The following relative frequency counts for key searches over a 24-hour period have been collected:

<table>
<thead>
<tr>
<th>Key:</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count:</td>
<td>7</td>
<td>12</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>

An optimal implementation is desired.
(a) Derive the optimal binary search tree by inspection. What is the cost?
(b) Show that the design in (a) is optimal.

Although an optimal binary search tree can be designed by inspection an algorithm that can handle any size tree is needed.
(c) Derive the optimal binary search tree using a dynamic algorithm and show each step of the algorithm.
(d) What is the time and space complexity of the algorithm for n keys?

2. Implement the method `PrintRoot(int i, j)` which will print the optimal sub-tree defined between nodes i and j in pre-order traversal using the best matrix, and show how you would call this method to print the complete n-node tree. Assume all the nodes are enumerated.

3* The Viterbi algorithm is an important dynamic algorithm for finding the optimum state sequence in a finite-state machine system. Two important applications are in speech recognition decoding and decoding in communication system codecs based on convolutional codes. Many other applications for this algorithm have also been found. In this question the formulation of the Viterbi algorithm for speech recognition decoding will be discussed. We define:

- $b_j(o_t) =$ likelihood that observation at time $t$, $o_t$, was produced by state $j$
- $a_{ij} =$ probability of jumping from state $i$ to state $j$
- $\text{Init}_i =$ probability of being in state $i$ at time 0 (initial condition)

Assume we have N states and T observations. As an example if we know that the state sequence at each time-step is: 1 1 2 3 4 5 5, then we can calculate the likelihood of the observations as:

\[
\text{Init}_1.b_1(o_1).a_{11}.b_1(o_2).a_{12}.b_2(o_3).a_{23}.b_3(o_4).a_{34}.b_4(o_5).a_{45}.b_5(o_6).a_{55}.b_5(o_7)
\]

But the state sequence is hidden and only the observation sequence is known. Thus we define the best path as that path yielding the maximum likelihood (i.e. most likely path).
(a) A brute-force solution would be to consider all possible path permutations. What is the time complexity of this approach?
(b) The Viterbi algorithm is a dynamic algorithms which solves the problem by keeping track of the best path sequence likelihood up to time $t$ in state $i$, $d_i(t)$, and the state that maximises that likelihood at time $t$, $T_i(t)$. What is $d_i(t)$?
(c) Now what is $d_i(j)$ based on $d_i(i)$? The argument that maximises $d_i(j)$ is stored in $T_i(j)$. What is $T_i(j)$?
(d) Now what is $d_j(j)$ based on $d_i(i)$? What is $T_j(j)$?
(e) What is the likelihood of the best path state sequence?
(f) Show how to use $T_i(j)$ to derive the best path state sequence.
(g) What is the time complexity of the Viterbi algorithm?