void optsearchtree( int n, const float p[], float & minavg, index R[][]) {

    index i, j, k, diagonal;
    float A[1..n+1][0..n];

    for (i = 1; i <= n; i++) {
        A[i][i-1] = 0;
        A[i][i] = p[i];
        R[i][i] = i;
        R[i][i-1] = 0;
    } // end for

    A[n+1][n] = 0;
    R[n+1][n] = 0;
    for (diagonal = 1; diagonal <= n-1; diagonal++) {
        for (i = 1; i <= n-diagonal; i++) {
            j = i + diagonal;
            A[i][j] = \min_{i \leq k \leq j} (A[i][k-1] + A[k+1][j]) + \sum_{m=0}^{j} p[m];
            R[i][j] = a value of k above that gave the minimum;
        } // end for
    } // end for

    minavg = A[1][n];
} // end optsearchtree

1. Use the above algorithm to determine the optimal binary search tree for the following keys and probabilities of access.

<table>
<thead>
<tr>
<th>Keys:</th>
<th>Key1</th>
<th>Key2</th>
<th>Key3</th>
<th>Key4</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>p[2] = 0.3</td>
<td>p[3] = 0.4</td>
<td>p[4] = 0.1</td>
<td>p[1] = 0.2</td>
</tr>
<tr>
<td>D</td>
<td>p[4] = 0.1</td>
<td>p[1] = 0.2</td>
<td>p[2] = 0.3</td>
<td>p[3] = 0.4</td>
</tr>
</tbody>
</table>

2. Traveling Salesperson Problem (TSP) -- Find an optimal (i.e., minimum length) tour when at least one tour exists. A tour (or Hamiltonian circuit) is a path from a vertex back to itself that passes through each of the other vertices exactly once. (Since a tour visits every vertice, it does not matter where you start, so we will general start at v1.) What are the length of the following tours?

a) [v1, v2, v3, v4, v1]

b) [v1, v3, v2, v4, v1]

c) List another tour starting at v1 and its length.
d) For a graph with "n" vertices (\(v_1, v_2, ..., v_n\)), one possible approach to solving TSP would be to brute force generate all possible tours to find the minimum length tour. "Complete" the following decision tree to determine the number of possible tours.

3. To apply dynamic programming, we need to be able to compute an optimal solution in a bottom-up (smaller to larger subproblems).
   a) What is part of an optimal tour?

   b) It helps me to think of the "final" (top-most) calculation giving the optimal tour.