1. This question concerns 2–3 trees.
   a. **2 pts.** What are the guides at the root for the 2–3 tree shown below?

   ![2-3 Tree Diagram](image_url)

   b. **5 pts.** Show the effect of the operations Delete(8) on the 2–3 tree shown below.

   ![2-3 Tree Diagram](image_url)

   c. **3pts.** What are the exact minimum and exact maximum heights of a 2–3 tree storing \( n \) items (where the height of a single node tree is defined to be zero)?
2. **10 pts.** Show the steps in running each of Prim’s and Kruskal’s MST algorithms on the following graph (it suffices to show the changed tree or forest as each successive edge is added. Prim’s algorithm should be started at the vertex $s$.

![Graph Diagram]

**Prim’s algorithm:**

**Kruskal’s algorithm:**

**Other possible data structures and algorithms for step by step illustration:**
heaps, merge sort, radix sort, insertion sort, DFS or BFS on a graph, Dijkstra’s algorithm, construction of a Huffman code, hashing.
3. (5 pts.) State whether the following assertions are True (T) or False (F).
   a. $100n^4 = O(n^5)$.
   b. $2^n + 1000n^2 = \Theta(2^n)$.
   c. $3^n = \Theta(4^n)$.
   d. Circle the larger of the following two functions (or circle them both if they are equal).
      \[
      2^{\log n} \quad n + 1
      \]
   e. For the following two functions, circle the one that grows to be the asymptotically larger
      as $n$ tends to infinity. If they remain within constant factors of each other, circle both.
      \[
      (\log n)^2 \quad n
      \]

4. Consider the following code fragment.

   function What(n);
   if $n \leq 1$ then return(1);
   else return($n^2 + 2 \cdot $ What($n/2$))
end

Suppose that $n$ is an integer power of 2.

a. (5 pts.) Write down the recurrence equation for $R(n)$, the number of recursive calls that
   occur when a call to What($n$) is made, excluding the initial call.

b. (5 pts.) Write down the recurrence equation for the value $V(n)$ computed by the call
to What($n$).
5. Consider the following code fragment.

1. \textbf{function} \textit{Rand}\,(n);  
2. \hspace{1em} \textbf{if} \; n \leq 3 \; \textbf{then return} (1);  
3. \hspace{1em} \textbf{else do}  
4. \hspace{2em} \text{assign } x \text{ to be 0 with probability } \frac{3}{6}, 1 \text{ with probability } \frac{1}{6}, \text{ and } 2 \text{ with probability } \frac{2}{6};  
5. \hspace{2em} \textbf{if } x = 0  
6. \hspace{3em} \textbf{then return} (\textit{Rand}\,(n) + 2)  
7. \hspace{2em} \textbf{else if } x = 1 \; \textbf{then return} (3 \times \textit{Rand}\,(n - 2) + 2 \times \textit{Rand}\,(n - 3))  
8. \hspace{2em} \textbf{else } (* x = 2 *) \; \textbf{return} (2 \times \textit{Rand}\,(n - 1) + \textit{Rand}\,(n - 2))  
9. \hspace{1em} \textbf{end } (* \text{ else do } *)  
end

\textbf{a.} (5 pts.) Write down the recurrence equation for the expected (average) number of times \( Z(n) \) that line 5 is executed over the course of the whole computation when a call to \textit{Rand}\,(n) is made.

\textbf{b.} (5 pts.) Write down the recurrence equation for \( V(n) \), the expected (average) value computed when a call to \textit{Rand}\,(n) is made.
6. Use the recursion tree method to solve the following recurrence equations. Full credit will be given for a solution that is written as a sum, whether as a closed form or as a sequence of terms (but make sure to show what are the first and last terms and what is the form of the sequence).

a. (5 pts.)

\[
\begin{align*}
R(1) &= 2 \\
R(n) &= n + 3R(n/2) & n > 1 \text{ and } n \text{ is an integer power of 2.}
\end{align*}
\]

b. (5 pts.)

\[
\begin{align*}
S(0) &= 1 \\
S(n) &= 2^n + 2S(n - 1) & n > 0
\end{align*}
\]
7. a. (5 pts.) Let \( T \) be an arbitrary tree so each vertex can have any number of children.

You are to write an algorithm which, for each vertex \( v \), will compute the number of nodes that are descendants of \( v \) including \( v \) itself, storing the result in \( v.des \).

The following example tree shows the values to be computed.

<table>
<thead>
<tr>
<th>Input tree</th>
<th>Output tree showing ( .des ) field</th>
</tr>
</thead>
<tbody>
<tr>
<td>o</td>
<td>6</td>
</tr>
<tr>
<td>/ \</td>
<td>/ \</td>
</tr>
<tr>
<td>/ \</td>
<td>/ \</td>
</tr>
<tr>
<td>o</td>
<td>1</td>
</tr>
<tr>
<td>/</td>
<td>\</td>
</tr>
<tr>
<td>/</td>
<td>\</td>
</tr>
<tr>
<td>o o o</td>
<td>1 1 1</td>
</tr>
</tbody>
</table>

Please complete the following procedure. Remember to have a driver procedure or to make an initial call.

Des\((v, \quad \quad)\);

\[
\text{for each child } w \text{ of } v \text{ do}
\]

Des\((w, \quad \quad)\);

\[
\text{end } (* \text{ for } *)
\]

end

Driver procedure/Initial call (answer here too):
b. (5 pts.) Now suppose that the input is a dag $G = (V, E)$. The task is similar: for each node $v$, compute the number of nodes reachable from $v$, storing the result in $v.rch$.

Hint: It will be helpful to compute $v.des$, the number of descendants of node $v$ in the DFS tree formed by your procedure.

Please complete the following procedure. Remember to have a driver procedure or to make an initial call.

Rch($v$, );

for each child $w$ of $v$ do

Rch($w$, );

end (* for *)

end

Driver procedure/Initial call (answer here too):
8. (10 pts.) Let $G = (V, E)$ be a directed graph stored in an adjacency matrix $A[1 : n, 1 : n]$ in which each edge $(u, v)$ has a length $L(u, v)$, and a color, $C(u, v)$, which are also both $n \times n$ arrays. The cost of a path $P$ from vertex $w$ to vertex $x$ is equal to the total edge length of the edges on the path plus the number of color changes along the path. Thus if the edges have the color sequence RRBBBR, as there are 2 color changes one needs to add 2 to the total edge length to obtain the cost of the path. As usual, suppose that there are no negative length cycles in $G$ (under this cost measure). Show how to build a graph $H$ such that the cost of shortest paths in $H$, defined in the standard way, corresponds to the cost of shortest paths in $G$ as just defined. Be sure to specify what is the correspondence. Your construction needs to be computable in time linear in the size of $G$; remember to justify this.
9.a. (4 pts.) Suppose that you are maintaining some statistics regarding tennis players, identified by name. For each player, you need to store the number of games won. Suppose that there are \( n \) players. There are three basic operations: add a player, delete a player, and for a given player, increase its score (of games won). These all need to run in \( O(\log n) \) time. What data structure would you use? Explain how it supports these three operations in \( O(\log n) \) time.

b. (6 pts.) Suppose that you want to be able to determine in \( O(\log n) \) time how many players won at least \( x \) games, given a query \( x \). Explain how to modify your solution to part (a) so as to support this operation in addition to the three operations specified in part (a).
10. **(10 pts.)** Let $E$ be an arithmetic expression comprises $2n - 1$ items: a sequence of alternating integers and operators, beginning with an integer, where the operators are either additions or multiplications. For example: $1 + 3$, $1 \times 3 + 4 \times 3$. Your task is to give an algorithm that finds the expression tree with $n$ leaves and $n - 1$ internal nodes that causes the input expression to evaluate to the maximum value possible. Your algorithm should run in $O(n^3)$ time. Remember to analyze the running time of your algorithm.

**Other possible topics.** Divide and conquer algorithms (e.g., sorting, binary search), heaps, MST algorithms.