1. For each pair of 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.

   a. \( 10n^5 \)  
   \( 1,000,000n^4 + 7n^2 \log n \)

   b. \( n^{1/\log n} \)  
   1

   c. \( 3^n \)  
   \( 2^n \)

   d. \( n^{\sqrt{n}} \)  
   \( \sqrt{n}^n \)

   e. \( \log n \)  
   \( \log \log n \)

2. Consider the following code fragment.

   ```
   function Comp(n):
       if n ≤ 1 then return(0);
       else return(n + Comp(n - 1) + Comp(n - 2))
   end
   ```

   a. Write down the recurrence equation for the running time \( T(n) \) when a call to \( \text{Comp}(n) \) is made.

   b. Write down the recurrence equation for the value \( C(n) \) computed by the call to \( \text{Comp}(n) \).
3. Consider the following code fragment.

\begin{verbatim}
function Rand(n);
    if n ≤ 2 then return(1);
    else do
        assign x to be 0 with probability \(\frac{1}{6}\), 1 with probability \(\frac{2}{6}\), and 2 with probability \(\frac{3}{6}\);
        if x = 0 then return(21 × Rand(n))
        else if x = 1 then return(3 × Rand(n − 1) + 2 × Rand(n − 2))
        else (* x = 2 *) return(7 × Rand(n − 1) + 4 × Rand(n − 1))
    end (* else do *)
end
\end{verbatim}

a. Write down the EXACT recurrence equation for the expected (average) number of times Z(n) that x is assigned the value 0 over the course of the whole computation when a call to Rand(n) is made.

b. Write down the EXACT recurrence equation for expected (average) number of times BC(n) that Rand(1) is called over the course of the whole computation when a call to Rand(n) is made. (BC(1) is defined to be 1.)
4. Use the recursion tree method to solve, as best you can, the following recurrence equations. It is OK to write $T(n) = \text{some specific sum of } n \text{ or } \log n \text{ terms or whatever if you do not recall how to add up the sequence you obtain).}

a. \[ R(1) = 2 \]
\[ R(n) = 1 + R(n/2) \quad n > 1 \text{ and } n \text{ is an integer power of 2.} \]

b. \[ S(0) = 1 \]
\[ S(n) = 2^n + 4S(n - 1) \quad n > 0 \]
c. It suffices to obtain a $\Theta$ estimate for $T(n)$.

\[
T(n) = 1 \quad n \leq 1
\]

\[
T(n) = 4 \quad n = 2
\]

\[
T(n) = n^2 + 2T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) \quad n > 1 \text{ and } n \text{ is an integer power of } 4.
\]
5. Consider the following MaxHeaps, and show the effect of the indicated operations:

a. (3 pts.) Insert(22)  
Result:

```
   24
  / \  
 /   
18   23
 /   /  
/   /  
9   11 16 2
 /  
5
```

b. (3 pts.) DeleteMax  
Result:

```
   42
  / \  
 /   
25   36
 /   /  
/   /  
14  16 7 18
 /  
4   6
```

c. (2 pts.) What is the maximum number of nodes in a $k$ level heap?

d. (2 pts.) Hence how many levels does an $n$-node heap have?
6. Let $T$ be an arbitrary tree so each vertex can have any number of children.

Suppose that there is a field $v.val$ which already holds an integer value for each vertex, possibly a negative value. You are to write an algorithm which will use an additional field, $v.path\_tot$. For each node $v$ your task is to compute the sum of the values on the path from the tree root to $v$ including the value at node $v$; this value should be stored in the field $v.path\_tot$.

The following example tree shows the values to be computed.

Input tree showing \texttt{.val} field output tree showing \texttt{.path\_tot} field

\begin{verbatim}
  3  
 / \  
5 -4  
/ | \  
6 2 3
\end{verbatim}

\begin{verbatim}
  3  
 / \  
8 -1  
/ | \  
5 1 2
\end{verbatim}

Please complete the following procedure. Remember to make an initial call. 

\begin{verbatim}
PathCalc(v, 

for each child $w$ of $v$ do

PathCalc($w$, 

end (* for *)

end

Initial call (answer here too):
\end{verbatim}
b. Now suppose that we want to also compute in field $v.lf.pth$ the maximum $x.path_tot$ value over all the leaves in $v$'s subtree. Augment your program for part a so as to compute this value too.

The following example tree shows the values to be computed.

<table>
<thead>
<tr>
<th>Input tree showing</th>
<th>output tree showing</th>
<th>output tree showing</th>
</tr>
</thead>
<tbody>
<tr>
<td>.val field</td>
<td>.path_tot field</td>
<td>.l.pth field</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>/ \</td>
<td>/ \</td>
<td>/ \</td>
</tr>
<tr>
<td>/ \</td>
<td>/ \</td>
<td>/ \</td>
</tr>
<tr>
<td>5 -4</td>
<td>8 -1</td>
<td>8 5</td>
</tr>
<tr>
<td>/</td>
<td>\</td>
<td>/</td>
</tr>
<tr>
<td>/</td>
<td>\</td>
<td>/</td>
</tr>
<tr>
<td>6 2 3</td>
<td>5 1 2</td>
<td>5 1 2</td>
</tr>
</tbody>
</table>

PathCalc($v$);

for each child $w$ of $v$ do

PathCalc($w$);

end (* for *)

end
7. Suppose you are given two $n$-item sorted arrays, $A[1 : n]$ and $B[1 : n]$, where all the items across both arrays have distinct values. Your task is to give an algorithm to find the rank $n$ item $z$ among the $2n$ items in $A$ and $B$. You may assume that $n$ is an integer power of 2.

a. (3 pts.) Suppose you look at $x = A[n/2]$ and $y = B[n/2]$. If $x < y$ what can you deduce about $z$’s location?

b. (7 pts.) Hence give a procedure to locate $z$ running in time $\Theta(\log n)$. 