1. Let $f, g, h$ be any three positive integer functions.

   Prove that if $f(n)$ is $O(h(n))$ and $g(n)$ is $O(h(n))$ then $f(n) \times g(n)$ is $O((h(n))^2)$.

   $5 \text{ pts}$
2. Consider the recurrence relation:

\[ T(n) = \begin{cases} 
0 & \text{if } n = 1 \text{ or } n = 2 \\
2 \cdot T(\lfloor n/2 \rfloor) + 1 & \text{otherwise}
\end{cases} \]

(i) Compute \( T(n) \) for \( n = 1, 2, \ldots, 10. \) 

5 pts
(ii) Use any method to solve the recurrence relation in part (i), that is, find a function $f(n)$ such that $T(n)$ is $\Theta(f(n))$. You must explain how you arrived at your solution.

10 pts
3. Give pseudocode for a function \texttt{RANGE-SUM} that given an array \( A[1..n] \) of integers (not necessarily positive) and two indices \( 0 < i \leq j \leq n \) returns the sum of all integers in the sub-array \( A[i..j] \). Analyse the running time of your function.

\begin{verbatim}
8 pts

\textbf{function} \texttt{RANGE-SUM} (\texttt{int} \( A[] \), \texttt{int} \( n \), \texttt{int} \( i \), \texttt{int} \( j \))
\triangleright \text{returns the sum of all integers in subarray} \( A[i..j] \)
\end{verbatim}
4. Give pseudocode for a function $\text{MAX-RANGE-SUM}$ that given an array $A[1..n]$ of integers (not necessarily positive) finds the maximum range-sum amongst all possible ranges. For example, given the array $A[1..9]$

\[-2\ 2\ 2\ 3\ -4\ 2\ 3\ -2\ -1\]

the function should return 8 because the sum of numbers in the subarray $A[2..7] = 8$ and no other range has a greater sum. Analyse the worst-case runtime of your algorithm.

12 pts
5. Compute the number of multiplications performed by the following recursive algorithm $FP$ as a function of $n$.

algorithm $FP(n)$
    if $((n = 1) \text{ OR } (n = 2))$
        return(2)
    else
        return($FP([n/2]) \times FP([n/2])$)

$10$ pts