1. **(U & G-required) [20 points]** Consider the following recursive algorithm.

   ```
   ALGORITHM Min1(A[0..n-1])
   //Input: An array A[0..n-1] of integer numbers
   if n = 1    return A[0]
   else temp ← Min1 (A[0..n-2])
       if temp ≤ A[n-1] return temp
       else return A[n-1]
   ```

   a) [5 points] What does this algorithm compute?
   b) [15 points] Set up a recurrence relation for the algorithm’s basic operation count and solve it.

2. **(U & G-required) [30 points]** Solve the following recurrences using the method of your choice.

   a) \( T(n) = 2T\left(\frac{n}{2}\right) + n^3 \)
   
   b) \( T(n) = T\left(\sqrt{n}\right) + 1 \)  **(Hint: use the same substitution we made in the example in class)**
   
   c) \( T(n) = 3T\left(\frac{n}{2}\right) + n \log n \)

3. **(U & G-required) [50 points] (U-required)**

   a) [25 points] Draw the recursion tree for \( T(n) = T(n/4) + T(n/2) + n^2 \) and provide a tight asymptotic bound on its solution.

   b) [25 points] Use the iteration method to solve the following recurrence:

   \[ T(n) = 4T(n/2) + n \]
4. (G-Required) [20 points] Consider the following recursive algorithm:

**ALGORITHM Q (n)**

// Input: A positive integer n
if \( n = 1 \)
  return 1
else
  return \( Q(n-1) + 2n - 1 \)

a) [10 points] Set up a recurrence relation for this function’s values and solve it to determine what this algorithm computes.

b) [10 points] Set up a recurrence relation for the number of multiplications made by this algorithm and solve it.

Extra credit

5. [20 points] Consider the following algorithm.

**ALGORITHM Mystery(n)**

//Input: A nonnegative integer n
\( S \leftarrow 0 \)
\( i \leftarrow 1 \) to \( n \)
  \( S \leftarrow S + i \times i \)
return \( S \)

a) [5 points] What does this algorithm compute?

b) [15 points] Compute the running time of this algorithm.