There are 5 questions for a total of 10 points.

1. Solve the following recurrence relations. You may use the Master theorem where ever possible. You may write your answer using big-oh notation.
   (a) (1 point) \( T(n) \leq 2 \cdot T(n/2) + cn \log n; \ T(1) \leq c \)
   (b) (\( \frac{1}{2} \) point) \( T(n) \leq 2 \cdot T(n/2) + cn^{3/2}; \ T(1) \leq c \)
   (c) (\( \frac{1}{2} \) point) \( T(n) = 2 \cdot T(n-1) + 1; \ T(1) = 1 \)

2. (1 point) How many times does the following program print “Hello World”? You may assume that \( n \) is a power of 2.

   ```
   F(n)
   - If (n = 1) return
   - Print(“Hello World”)
   - F(n/2)
   - F(n/2)
   - F(n/2)
   - F(n/2)
   ```

3. (2 points) Analyze the running time for the following recursive program that computes the \( n^{th} \) Fibonacci number. Write a recurrence relation for the running time \( T(n) \) and then solve this recurrence relation using substitution.

   ```
   Fib(n)
   - If (n = 1 or n = 2) return(1)
   - a ← Fib(n − 1)
   - b ← Fib(n − 2)
   - return(a + b)
   ```

4. (2 points) Given an array \( A = A[1]A[2]...A[n] \) containing \( n \) distinct positive integers sorted in ascending order, design an \( O(\log n) \)-time algorithm that determines if there is an index \( i \) such that \( A[i] = i \). Discuss correctness and running time.

5. (3 points) A degree-(\( n-1 \)) polynomial \( A(x) \) may be represented as
   \[ A(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + ... + a_{n-1} \cdot x^{n-1}. \]
   Constants \( a_0, a_1, ..., a_{n-1} \) are known as the coefficients of this polynomial. Such a degree-(\( n-1 \)) polynomial may be represented using its coefficients \( (a_0, ..., a_{n-1}) \). Given coefficients \( (a_0, ..., a_{n-1}) \) and \( (b_0, ..., b_{n-1}) \) of two polynomials \( A(x) \) and \( B(x) \), design an algorithm to output the coefficients \( (c_0, c_1, ..., c_{2n-2}) \) of the product of the polynomials \( A(x) \) and \( B(x) \). That is, your algorithm should output the coefficients of polynomial \( C(x) \), where \( C(x) = A(x) \cdot B(x) \). Your algorithm should work in time \( O(n^{\log_2 3}) \). Discuss correctness and running time.