Answer all questions. Give informal (at least) proofs for all answers. Grading will be on completeness and logical correctness, and if applicable, efficiency, as well as correctness. Out of 80 points.

Order questions- 10 points each For each, answer True or False, and give a short explanation for your answer.

1. \( n^2 \in O(2^n) \).
2. \( n^2 \in \Theta(2^n) \).
3. If \( f \) and \( g \) are functions from positive integers to positive integers, and \( f(n) \in O(g(n)) \), then \( f(n) + g(n) \in O(g(n)) \).
4. If \( f \) and \( g \) are functions from positive integers to positive integers, and \( f(n) \in O(g(n)) \), then \( 2f(n) \in O(2g(n)) \).

Analyzing loops-20pts Say that an airline has a sorted array of arrival times for flights at an airport \( A[1...n_a] \) and a sorted array of departure times for flights at the same airport, \( D[1...n_d] \). The airline wants to give one hour for a passenger to make a connection. Here is an algorithm that finds for each arriving flight \( 1 \leq i \leq n_a \), the first departing flight \( 1 \leq j \leq n_d \) where a passenger arriving on the first has enough time to make their connection and depart on the second. If there are no such flights, we define the answer as 0.

\[
\text{FirstConnection}(A[1...n_a], D[1...n_d]: \text{sorted lists of positive integers})
\]

1. Initialize \( FC[1..n_A] \) as an array of integers.
2. \( I \leftarrow 1, J \leftarrow 1 \)
3. FOR \( I=1 \) to \( n_a \) do:
   4. While \( J \leq n_d \) and \( B[J] < A[I] + 1 \) do: \( J++ \)
   5. IF \( J > n_d \) do \( FC[I] \leftarrow 0 \) else \( FC[I] \leftarrow J \).
6. Return \( FC[1..n_a] \).

Give a time analysis, up to order, for this algorithm, in terms of the total input size \( n = n_a + n_d \). Be sure to explain your answer.

Correctness Proof: 20 points Prove the following loop invariant for the algorithm above: at all times, for every \( 1 \leq i \leq n_a, 1 \leq j \leq n_d, \) if a passenger arriving on the \( i \)'th arriving flight can connect to flight \( j \), then \( j \geq J \).