1. [11 points] Prove Algorithm 1 for binary search is correct using strong induction on \( n \), where \( n = f - s \). You should also use proof by cases for the if-elses. **Note:** This is a complex proof, and you will probably not get all of the parts correct. Just try your best :) Recall, if \( A \) is sorted in increasing order and no integers are repeated, that means \( i < j \) if and only if \( A[i] < A[j] \).

   **Input** : (1) Array \( A \) containing integers, where there are no repeated integers and the integers are sorted from smallest to largest, (2) an element \( V \) in \( A \), and (3) two indeces \( s \) and \( f \), where \( s \leq f \) and the index of \( V \) is between \( s \) and \( f \) (inclusive)

   **Output**: Index \( j \) such that \( A[j] = V \), and \( s \leq j \leq f \).

   ```
   // Base Case
   1 if f - s = 0 then
   2     return s;
   3 end
   // Recursive step
   4 mid = \lfloor (f + s) / 2 \rfloor;
   // \lfloor \cdot \rfloor means round down to the nearest integer
   5 if A[mid] = V then
   6     return mid;
   7 else
   8     if A[mid] < V then
   9         return BinarySearch(A, V, mid + 1, f);
  10     else
  11         return BinarySearch(A, V, s, mid - 1);
  12 end
  13 end

   Algorithm 1: BinarySearch(A, V, s, f)
   ```

2. **Party-trick Proof** [11 points] Suppose you are at a party with 19 acquaintances (so there are 20 people at the party). Prove (using a proof by contradiction) that there must be at least two people at the party who talked to the same number of people over the course of the evening. (Note: we assume that if Alice talked to Bob, that also means that Bob talked to Alice.)