Loop invariant for selectionSort

```
void selectionSort( vector<string> & A ){
  for( int i = 0; i < A.size(); i++ ) {
    int min = findMin(A,i);
    string temp = A[i];
    A[i] = A[min];
    A[min] = temp;
  }
}
```

Use n to denote `A.size()`. Assume strings in A are distinct (so I can say "increasing" rather than "non-decreasing").

Insufficient loop invariant: The contents of $A[0..i-1]$ are sorted at the start of iteration i. (Why is this not enough?)

Theorem. *For all $i = 0 \ldots n$, at the start of iteration i, the contents of $A[0..i-1]$ are the i smallest strings in A in sorted (increasing) order and the set of strings in $A[0..n-1]$ is the same as the input.*

Proof. (by induction on i)

Base case: $i = 0$ (i.e. before any loop body executes). There are no strings in $A[0..i-1]$ so the base case is true.

Inductive Hypothesis: We assume that the invariant is true at the start of iteration i.

Inductive Step: `findMin` returns the index, min, of the smallest string in $A[i..n-1]$. We swap $A[i]$ with $A[\text{min}]$. Now $A[i] \geq A[0..i-1]$ by the inductive hypothesis and $A[i] \leq A[i+1..n-1]$ by the correctness of `findMin`, so at the end of iteration i (and the start of iteration $i+1$), $A[0..i]$ are the $i+1$ smallest strings in A in increasing order. Since we swap, the set of strings in $A[0..n-1]$ doesn't change.

Termination: The for-loop terminates at the end of iteration $i = n-1$ (i.e. the start of iteration n). At this point, $A[0..n-1]$ are all the strings in A in increasing order. In other words, $A[0..n-1]$ is the sorted version of the input A. □