Loop Invariants: Prove loops are correct

setup
while (condition) {
    stuff
}
Great output

Induction tailored to loops

Parts of Loop Invariant Proof

1. State Invariant: thing(s) that is true before & after each loop iteration

2. Base Case: Show invariant is true before loop starts.

3. Maintenance: Show if invariant is true before an iteration, it is true after an iteration

Input: Array $A$ of integers of length $n$

Output: Smallest value of $A$

1 $\text{min} = A[1]$;
2 $i = 2$;
3 while $i \leq n$ do
4     if $A[i] < \text{min}$ then
5         $\text{min} = A[i]$
6     end
7     $i++$;
8 end
9 return $\text{min}$;

Algorithm 1: $\text{Smallest}(A)$
**Loop Invariants**

ex:

\[
\text{MIN(array A of length n)}
\]

- \( \min = A[1] \)
- \( i = 2 \)
- while (i \leq n)
  - if (\( A[i] < \min \)):
    - \( \min = A[i] \)
  - \( i++ \)
- return \( \min \)

**Loop Invariant**

- \( \min \) is minimum of \( A[1:i-1] \)

Base Case: Before loop starts: \( i=2, \min = A[1] \).
- \( \min \) is minimum of \( A[1:1] \)
Maintenance

\[ \text{min} = \text{minimum} \{A[1:i-1]\} \text{ at start of loop. But} \]
\[ \text{minimum} \{A[1:i]\} = \text{minimum} \{A[1:i-1], A[i]\} = \text{minimum} \{\text{min, } A[1:j]\} \]

which is what min becomes at the end of the loop.

\text{Termination:} 
- \text{loop terminates at } i = n + 1.
- i increases, so termination will occur.

\text{Invariant:} \quad \text{min} = \text{minimum} \{A[1:n]\}

* Make sure your invariant gives you the result you want at end of loop *
Input : Array $A$ of integers of length $n$

Output: Array containing sorted elements of $A$

1 \hspace{1em} \textbf{for} $k = 1 \text{ to } n - 1$ \textbf{do}
2 \hspace{2em} \textbf{for} $j = n \text{ to } k + 1$ \textbf{do}
3 \hspace{3em} \textbf{if} $A[j] < A[j - 1]$ \textbf{then}
4 \hspace{4em} \textbf{Swap} $A[j]$ and $A[j - 1]$;
5 \hspace{3em} \textbf{end}
6 \hspace{2em} \textbf{end}
7 \hspace{1em} \textbf{end}
8 \text{ return } A;

Algorithm 2: \textbf{BubbleSort}(A)