Supplemental Document for Reading #4
Another Useful Theorem about `while` Loops

The following theorem was stated, as "Loop Theorem #2", in the notes for Reading #4.

Second Loop Theorem: Consider a `while` loop

```
while (t) do { S }
```

with a loop test `t` and a loop body `S`. Suppose that the following properties are satisfied.

(a) If the problem's precondition is satisfied when an execution of the algorithm including this loop begins, then every execution of the loop body (that is part of this execution of the algorithm) ends.

(b) A bound function for this `while` loop exists.

Then every execution of this `while` loop, included in an execution of the algorithm beginning with the problem's precondition satisfied, ends.

Furthermore, the value of the bound function immediately before the beginning of an execution of this loop is an ***upper bound*** for the number of times that the loop body is executed before this execution of the loop ends.

The following, slightly more general, claim will be proved first.

Claim: Consider a `while` loop

```
while (t) do { S }
```

while a loop test `t` and a loop body `S`. Suppose that the following properties are satisfied.

(a) If the problem's precondition is satisfied when an execution of the algorithm including this loop begins, then every execution of the loop body (that is part of this execution of the algorithm) ends.

(b) A bound function for this `while` loop exists.

Then, for every integer $k \geq 0$, if the bound function for the `while` loop has value k when the loop test `t` is checked (during an execution of the algorithm that began with the corresponding problem's precondition being satisfied), then this execution of the loop will end after *at most* k more executions of the body of the loop.

Proof: This will be proved will be proved by induction on k. The *strong* form of mathematical induction will be used, and the case $k = 0$ will be considered in the basis.

Basis: Suppose that properties (a) and (b) in the statement of the claim are satisfied. It is necessary to show that if the bound function has value 0 when the loop test `t` is checked (during an execution of the algorithm that began with the corresponding problem's precondition being satisfied), then this execution of the loop will end without *any* more executions of the body of the loop.

This follows by the definition of a ***bound function*** for a `while` loop, which implies that the loop test `t` will fail when the bound function's value is less than or equal to zero, so that the loop will terminate immediately, as required.

Inductive Step: Once again, suppose that properties 1 and 2 in the statement of the claim are satisfied. Let h be an integer such that $h \geq 0$. It is necessary and sufficient to use (and assume) the following

> Inductive Hypothesis: For every integer j such that $0 \leq j \leq h$, if the bound function for the `while` loop has value j when the loop test `t` is checked (during an execution of the algorithm that began with the corresponding problem's precondition being satisfied), then this execution of the loop will end after *at most* j more executions of the body of the loop.

in order to prove the following

> Inductive Claim: If the bound function for the `while` loop has value $k+1$ when the loop test `t` is checked (during an execution of the algorithm that began with the corresponding problem's precondition being satisfied), then this execution of the loop will end after *at most* $k+1$ more executions of the body of the loop.

With that noted, suppose that the value of the bound function is $k+1$ when the loop test `t` is checked (for there is nothing to be proved, otherwise). Either the test `t` passes or fails; these cases are considered separately, below.

- *Case:* The loop test `t` passes. Now, since the problem's precondition was satisfied at the beginning of the execution of the algorithm, it follows by Property (a) that the subsequent execution of the loop body will eventually end.

  It is possible that the bottom of the loop body is not reached before that and the loop test `t` never gets checked again; for example, this would happen if the loop included a `return` statement that was executed during this execution of the loop body. In any such case, this must be the *final* execution of the loop body, so that that there was only one more execution of the loop body after the test `t` was checked. Now, since $k \geq 0$, $1 \leq k + 1$, and this establishes the Inductive Claim in this case.

  Suppose, one the other hand, that the bottom of the loop body *is* reached during this execution of the loop body, so that the loop test `t` is checked again. Let j be the value of the bound function when this happens; then it follows by the definition of a bound function for a `while` loop that $j \leq k$, since the value of the bound function has been decreased by *at least* one since the last time the loop test `t` was checked.

  If $j < 0$ then it follows, once again, by the definition of a bound function for a `while` loop that the loop test `t` fails — establishing the Inductive Claim, as in the previous subcase, since $1 \leq k + 1$.

  Otherwise $0 \leq j \leq k$, and it follows by the Inductive Hypothesis that there are at most j more executions of the loop body, after *this* execution of the loop test. Consequently, there are at most $j + 1 \leq k + 1$ executions after the execution of the loop test `t` when the bound function had value $k + 1$, as needed to establish the Inductive Claim in this case too.

- *Case:* The loop test `t` fails. In this case the execution of the loop ends immediately after this, implying the Inductive Claim since $0 \leq k + 1$.

Since the Inductive Claim has been established in every possible case this completes the Inductive Step, and the proof of the claim. □

Proof of Second Loop Theorem: It remains only to notice that this is a straightforward consequence of the above claim. □