1. Consider the following algorithm:

```
func(n):
    # Pre: n is a natural number
    x = 0
    i = 0
    while i < n:
        i = i + 1
        x = x + i
    return x
```

(a) State preconditions and postconditions for this algorithm.

Postconditions: \( x = \sum_{i=0}^{n} i \)

(b) Use induction to prove the loop invariants \( i \leq n \) and \( x = \sum_{j=0}^{i} j \) for the while loop.

Let \( k \geq 0 \). Assume \( H(k) \): \( i_k \leq n \) and \( x_k = \sum_{j=0}^{i_k} j \).

Want to show that \( i_{k+1} \leq n \) and \( x_{k+1} = \sum_{j=0}^{i_{k+1}} j \)

Case: There is no \( k+1 \)th iteration. Then \( i_{k+1} = i_k \) and \( x_{k+1} = x_k \), so the loop invariant holds.

Case: There is a \( k+1 \)th iteration of the loop.

Then, \( i_k \) was such that the loop test passed, i.e. \( i_k < n \). Thus, \( i_{k+1} = i_k + 1 \leq n \).

\( x_{k+1} = x_k + i_{k+1} \) (since \( i = i + 1 \) is first)
\( = \sum_{j=0}^{i_{k+1}} j + i_{k+1} \) (by \( H(k) \))
\( = \sum_{j=0}^{i_k} j \)

So, the loop invariant holds in all cases.

Base case: Let \( k = 0 \). \( i_0 = 0 \leq n \) because \( n \in \mathbb{N} \).

\( x_0 = 0 = \sum_{j=0}^{0} j = \sum_{j=0}^{i_0} j \).

So the loop invariant holds in all cases.

(c) Prove that the loop terminates.

Let \( E_k = n - i_k \).

Need to show that (1) \( E_k \in \mathbb{N}, \forall k \) and (2) \( E_{k+1} < E_k \), if there is a \( k+1 \)th iteration.

(1): \( i_k, n \in \mathbb{N} \), and \( i_k \leq n \) \( \forall k \) by part (b). Thus, \( E_k \in \mathbb{N} \), and \( E_k \geq 0, \forall k \).

(2): If there is a \( k+1 \)th iteration, then \( E_{k+1} = n - i_{k+1} = n - (i_k + 1) = n - i_k - 1 < n - i_k = E_k \).

Thus, the loop terminates.
2. Prove that the following function is correct (by showing partial correctness and termination), according to its pre- and postconditions.

```python
def f(A):
    # Pre: A is a list of integers
    # Post: Returns true if and only if there is an even number of positive numbers in A
    even = True
    i = 0
    while i < A.length:
        if A[i] > 0:
            even = not even
        i = i + 1
    return even
```

**Partial Correctness:** Consider the loop invariant $i \leq A.length$ and even is True iff there are an even number of positive numbers in $A[0..i-1]$.

**Proof of loop invariant:** Let $k \geq 0$.

Assume $H(k)$: $i_k \leq A.length$ and $even_k$ is True iff there are an even number of positive numbers in $A[0..i_k-1]$.

Show $H(k) \implies C(k)$: $i_{k+1} \leq A.length$ and $even_{k+1}$ is True iff there are an even number of positive numbers in $A[0..i_{k+1}-1]$.

**Case:** There is no $k+1$th iteration. Then $i_{k+1} = i_k$ and the loop invariant holds.

**Case:** There is a $k+1$th iteration. Then the loop condition passed, so $i_k < A.length$ and $i_{k+1} = i_k + 1 \leq A.length$.

If $A[i_{k+1}]$ is non-positive, then $even_{k+1} = even_k$, and by $H(k)$ represents the number of positive numbers in $A[0..i_k]$, which is the same as the number of positive numbers in $A[0..i_{k+1}]$.

If $A[i_{k+1}]$ is positive, then $even_k$ is negated. There is one more positive number in $A[0..i_{k+1}]$ than there was in $A[0..i_k]$. By $H(k)$, $even_k$ was True if there were an even number of positive numbers in $A[0..i_k]$, and so there are an odd number of positive numbers in $A[0..i_{k+1}]$, and $even_{k+1}$ is False. A symmetric argument can be made if $even_k$ was False.

**Base Case:** Let $k = 0$. $i_0 = 0 \leq A.length$. $even_0$ is True, because there are 0 positive numbers in an empty subarray.

Thus, in all cases, the loop invariant holds.

The loop terminates when $i \geq A.length$. By the proof of the loop invariant, $i \leq A.length$. So, the loop terminates when $i = A.length$. Thus, by the loop invariant, even represents the number of positive numbers in all of $A$.

**Termination:** Let $E_k = A.length - i$. By loop invariant, $i \leq A.length$. So $E_k \geq 0 \implies E_k \in \mathbb{N}, \forall k$.

If there is a $k+1$th iteration, then $E_{k+1} = A.length - i_{k+1} = A.length - (i_k + 1) < A.length - i_k = E_k$.

Thus, the loop terminates.