CSC236 tutorial exercises, Week #8
sample solutions

For each of the algorithms in questions 1-3, prove termination. If proving termination requires a loop invariant, you may state it without proof. (Though you should be confident that your invariant actually holds, and comfortable with how it *could* be proved, if necessary.)

1.

```
1 def ssum(A):
2     """Pre: A is a list of non-negative ints
3     Post: return the sum of A
4     WARNING: A may be irrerversibly altered!
5     """
6     i = 0
7     s = 0
8     while i < len(A):
9         if A[i] == 0:
10            i += 1
11        else:
12            s += 1
13            A[i] -= 1
14    return s
```

Solution:
At the end of a given iteration j, let

- Σ_j be the sum of elements in A_j
- m_j be $\Sigma_j + (\text{len}(A_j) - i_j)$ (this will be my loop measure)

I claim the following as loop invariants for any iteration j:

- $\Sigma_j \in \mathbb{N}$.
- $i_j \in \mathbb{N}$
- $i_j \leq \text{len}(A)$

It follows that $m_j \in \mathbb{N}$.

Furthermore, the sequence $m_0, m_1, \ldots$ is decreasing. To see why, observe by the code that based on the condition checked on line 9, exactly one of the following happens in each iteration:

- i is increased

- Σ is decreased

In either case, m decreases.

2.

```python
def binsearch(A, x):
    """Pre: A is a sorted list of numbers. x is a number.
    Post: return i such that A[i] = x, or -1 if x is
          not an element of A.
    """
    lo = 0
    hi = len(A)-1
    while lo <= hi:
        m = lo + (hi - lo) // 2
        mid = A[m]
        if mid == x:
            return m
        elif mid < x:
            lo = m + 1
        else:
            hi = m - 1
    return -1
```

Solution:
In this answer I will assume the invariant $hi_j, lo_j \in \mathbb{N}$. I will also assume the invariant $lo_j \leq hi_j + 1$

Lemma 1. *For every iteration $j > 0$, $lo_{j-1} \leq m_j \leq hi_{j-1}$*

Proof. By the loop condition, $lo_{j-1} \leq hi_{j-1}$.

It follows that $(hi_{j-1} - lo_{j-1}) \mathbin{//} 2$ is non-negative, which means that $lo_{j-1} \leq m_j$.

To see why the other inequality holds, observe that $hi_{j-1} = lo_{j-1} + (hi_{j-1} - lo_{j-1})$. It follows that $lo_{j-1} + (hi_{j-1} - lo_{j-1}) \mathbin{//} 2 \leq hi_{j-1}$, since dividing the second term by 2 makes it no larger. □

(**Note:** It would also have been acceptable to simply treat the above lemma as another assumed loop invariant, rather than proving it.)

For a given iteration j, define loop measure $q_j = (hi_j - lo_j) + 1$. It follows from the previous lemma that q_j decreases with each iteration, since (if the loop doesn't exit early), either hi is decreased, or lo is increased. Either action decreases q_j.

That $q_j \in \mathbb{N}$ follows from the invariants I assumed initially.

3.

```python
def perambulate(A):
    """Pre: A is a non-empty list of non-negative ints
    """
    seen = []
    curr = A[0]
    i = 0
    while curr not in seen:
        i = (i + curr) % len(A)
        seen.append(curr)
```

```
10          curr = A[i]
11      return curr
```

Solution:
For each iteration j, define $m_j = \text{len}(A) - \text{len}(seen_j)$. It is clear that this decreases with each iteration, since we always increase the length of $seen$ on line 9.

For sake of contradiction, assume there is some iteration j such that $m_j < 0$. I will use the invariant that $seen_j$ contains only elements from A. In this scenario, it follows from the pigeonhole principle that $seen_j$ contains at least one duplicate element x. Consider the iteration $k+1$ in which x was last appended to $seen$. Then $curr_k = x$, and $seen_k$ is a list containing at least one instance of x. But in order for a $k+1$th iteration to execute, the loop condition required that $curr_k = x$ was not an element of $seen_k$, a contradiction. Therefore, by contradiction, $m_j \geq 0$ for all j.

Thus $m_0, m_1, \ldots$ is a decreasing sequence of natural numbers, and the loop exits.

Note: Once again, we could have replaced some of the argument above with another assumed loop invariant, such as that $seen_j$ contains no duplicate elements. A proof of this invariant would use the same line of reasoning as the above.

4. Identify the logical flaw in the following "proof" of termination of the function `mean`.

```python
def mean(a, b):
    """Pre: a and b are ints, a < b
    Post: return the arithmetic mean of a and b
    """
    while a != b:
        a += 1
        b -= 1
    return a
```

Proof of termination Define loop measure $m_j = b_j - a_j$.

$m_j \in \mathbb{N}$, since $a < b$, by the precondition.

It remains to show that m decreases with each iteration. For an arbitrary iteration $j > 0$, $m_j = m_{j-1} - 2$ (since we increase a by 1 and decrease b by 1).

Thus $\langle m_0, m_1, \ldots \rangle$ is a decreasing sequence of natural numbers, and therefore finite, so `mean` terminates. □

Solution:
The second line of the proof is not valid. The precondition only applies to a_0 and b_0, the initial values of a and b. It ensures that $m_0 \in \mathbb{N}$, but when we modify a and b inside the loop, we might reach a situation where $a_j > b_j$. In fact, this happens for any inputs with an odd difference. A call such as `mean(2, 3)` will loop infinitely.