Assignment 4: Arrays
15-414/15-424 Bug Catching: Automated Program Verification

Due: **11:59pm**, Thursday 9/27/18
Total Points: 50

1. **Filter a list (10 points)** Consider the following program α, which filters all of the 0-elements out of array a and stores them in b.

```
c := 0;
i := 0;
while(i < n) {
  if(a(i) != 0) {
    b(c) := a(i);
    c := c + 1;
  }
}
```

Construct a postcondition P such that when $[\alpha]P$ is valid, then all of the elements of b between 0 and $numnz(a, n)$ will hold values that are not 0. Then give a loop invariant that is sufficient to prove this postcondition using the loop rule. You do not need to give a proof, but explain why your postcondition captures the correctness of the code, and why your invariant is sufficient to prove it.

You can assume that $numnz$ is defined as:

$$numnz(a,n) = \begin{cases} 0 & \text{if } n \leqslant 0 \\ numnz(a, n-1) & \text{if } a(n) = 0 \\ 1 + numnz(a, n-1) & \text{if } a(n) \neq 0 \end{cases}$$

2. **Existential funk (5 points)** Recall the binary search program from lecture 6.

```
l := 0;
h := n;
while(l < h && a(m) != k) {
  if(a(m) < k)
    l := m + 1;
  else
    h := m;
  m := (l + h) / 2;
}
```

We decided on the following postcondition as suitable for describing the correctness of this code:

$$(l < h \rightarrow a(m) = k) \wedge (l \geqslant h \rightarrow \forall i.0 \leqslant i < n \rightarrow a(i) \neq k)$$

At first glance this seems like a convoluted way of capturing the very simple idea that whenever there exists an index of a between 0 and n taking the value k, then m holds that index on termination. Suppose instead that we decided to write the following postcondition:

$$\exists i.0 \leqslant i < n \rightarrow a(i) = k \rightarrow m = i$$

Explain why this postcondition fails completely and utterly as a specification of correctness.

3. **Invert an array (10 points)** Suppose that an array a is an injection: distinct indices map to distinct elements. Furthermore, we assume that a is defined on all indices i such that $0 \leqslant i < n$, and that it only maps to values in this range as well. We want to write a program that inverts a into a second array b, so that if a for example starts out as (for $n = 10$):

$$[3, 1, 0, 2]$$

Then after the code runs, b has the value:

$$[2, 1, 3, 0]$$

Your task is to write a specification for this program by giving `pre` and `post`.

4. **Implement and prove it (15 points)** Now that you have specified the behavior from problem 3, write a program to implement the functionality. Then, write a loop invariant that will allow you to prove its correctness with respect to your spec. Then, use the axioms of dynamic logic to conduct a sequent calculus proof that your implementation is correct.

   Hint: your program, especially the loop body, should be very short.

5. **And diamonds (10 points)** Give an example program α for which the following dynamic logic formula is valid, or explain why no such program exists.

$$\langle\alpha\rangle x < 0 \wedge \langle\alpha\rangle x \geqslant 0$$

   Now consider the formula:

$$[\alpha]x < 0 \wedge [\alpha]x \geqslant 0$$

   Does there exist an α that satisfies this? Why or why not?