IIT CS536: Science of Programming

Homework 5: Proofs, Loop Invariants

Prof. Stefan Muller
TAs: Chaoqi Ma, Zhenghao Zhao

Out: Thursday, Mar. 21
Due: Friday, Apr. 1, 11:59pm CST

This assignment contains 4 written task(s) and 3 tasks which you may solve on paper or using Dafny, for a total of 50 points.

SOLUTIONS

Logistics

Submission Instructions

Please read and follow these instructions carefully.

- Submit your homework on Blackboard under the correct assignment by the deadline (or the extended deadline if taking late days).
- You may submit multiple times, but we will only look at your last submission. Make sure your last submission contains all necessary files.
- Email the instructor and TAs ASAP if
  - You submit before the deadline but then decide to take (more) late days.
  - You accidentally resubmit after the deadline, but did not intend to take late days.

  Otherwise, you do not need to let us know if you're using late days; we'll count them based on the date of your last submission.
- Submit your written answers in a single PDF or Word document. Typed answers are preferred (You can use any program as long as you can export a .pdf, .doc or .docx; LaTeX is especially good for typesetting logic and math, and well worth the time to learn it), but *legible* handwritten and scanned answers are acceptable as well.
- If you are using Dafny for your programming questions, submit any Dafny files you modified (`sumarray.dfy`, `find.dfy`, `posneg.dfy`). **Do not rename these files.**
- Your Blackboard submission should contain only the file with your written answers, as well as the Dafny files if you are using Dafny. Do not compress or put any files in folders.

Collaboration and Academic Honesty

Read the policy on the website and be sure you understand it.

1 Minimal and Full Proof Outlines

Task 1.1 (Written, 7 points).

Convert the following minimal proof outline to a full proof outline by adding weakest preconditions and/or strongest postconditions. Proof obligations should appear in the proof outline as weakenings ($p \Rightarrow q$), but you don't need to prove them explicitly.

```
                                  $\{n \geq 0\}$
$i := \overline{0}$
$s := \overline{0}$
$\{\textbf{inv } i \leq n \land s = i^2\}$
while $(i < n)$ {
    $s := s + (\overline{2} * i + \overline{1})$;
    $i := i + \overline{1}$
}                                 $\{s = n^2\}$
```

Hint: You can use the fact that $i^2 = (i+1)^2 - 2i - 1$ (remember FOIL from middle/high school math?)

```
                            $\{n \geq 0\}$
$i := \overline{0}$                      $\{n \geq 0 \land i = 0\}$
$s := \overline{0}$                      $\{n \geq 0 \land i = 0 \land s = 0\}$
$\{\textbf{inv } i \leq n \land s = i^2\}$
while $(i < n)$ {             $\{i \leq n \land s = i^2 \land i < n\} \Rightarrow \{i < n \land s + 2i + 1 = (i+1)^2\}$
    $s := s + (\overline{2} * i + \overline{1})$;   $\{i < n \land s = (i+1)^2\}$
    $i := i + \overline{1}$               $\{i \leq n \land s = i^2\}$
}                           $\{i \leq n \land s = i^2 \land i \geq n\} \Rightarrow \{s = n^2\}$
```

Task 1.2 (Written, 5 points).

Consider the following (incorrect) version of the above program.

```
                                  $\{n \geq 0\}$
$i := \overline{0}$
$s := \overline{0}$
$\{\textbf{inv } i \leq n \land s = i^2\}$
while $(i < n)$ {
    $s := s + (\overline{2} * i)$;
    $i := i + \overline{1}$
}                                 $\{s = n^2\}$
```

Rewrite the proof outline from above for this version of the program. The bug shows up in the proof outline in the form of a proof obligation that cannot be proven (i.e., a predicate logic implication $p \Rightarrow q$ that is invalid); write the invalid predicate logic implication.

The new outline is:

```
                              {$n \geq 0$}
$i := \overline{0}$                      {$n \geq 0 \land i = 0$}
$s := \overline{0}$                      {$n \geq 0 \land i = 0 \land s = 0$}
{**inv** $i \leq n \land s = i^2$}
while $(i < n)$ {            {$i \leq n \land s = i^2 \land i < n$} $\Rightarrow$ {$i < n \land s + 2i = (i+1)^2$}
    $s := s + (\overline{2} * i)$;        {$i < n \land s = (i+1)^2$}
    $i := i + \overline{1}$              {$i \leq n \land s = i^2$}
}                            {$i \leq n \land s = i^2 \land i \geq n$} $\Rightarrow$ {$s = n^2$}
```

This obligation is invalid: $(i \leq n \land s = i^2 \land i < n) \Rightarrow (i < n \land s + 2i = (i+1)^2)$ because $i^2 \neq (i+1)^2 - 2i$.

2 Proofs with Loops

Task 2.1 (Programming, 8 points).

Write a program that takes an array a and sets s to the sum of the elements in a, and *provide its minimal proof outline*.

Precondition: T

Postcondition: $s = sumA(a, 0, |a|)$

where $sumA(a, i, j) = \Sigma_{k=i}^{j-1} a[k]$.

You may write the program and proof outline in your written answers, or you may fill in the method `sumArray` in `sumarray.dfy`, which already includes the pre- and post-conditions.

Task 2.2 (Programming, 12 points).

The following program takes an array a and an integer x and returns the index of the first element of a greater than x (or ends with $i = |a|$ if no element is greater).

```
$i := \overline{0}$;
while $(i < |a| \wedge a[i] \leq x)$ {
    $i := i + \overline{1}$
}
```

Precondition: T

a) Write a postcondition for the program that formally describes the requirements for the program written in English above (i is the index of the first element of a greater than x, or $i = |a|$ if no element is greater).

b) Write a loop invariant for the loop that would allow you to prove this postcondition.

You may answer this question in your written answers, or solve it in Dafny (in `find.dfy`) by a) replacing the `ensures` clause of `findFirstGreater` with your postcondition and b) adding a `invariant` to the while loop in `findFirstGreater` so that verification succeeds.

Task 2.3 (Programming, 12 points).

The following program takes an array a and sets e to true if and only if a has an equal number of positive and negative elements (0 is considered neither positive nor negative).

```
$i := \overline{0}$;
$n := \overline{0}$;
$p := \overline{0}$;
while $(i < |a|)$ {
    if $a[i] > \overline{0}$ then $\{n := n + \overline{1}\}$ else {skip};
    if $a[i] < \overline{0}$ then $\{p := p + \overline{1}\}$ else {skip};
    $i := i + \overline{1}$
}
$e := n = p$
```

Precondition: T
Postcondition: $e \to \mathit{numPos}(a, 0, |a|) = \mathit{numNeg}(a, 0, |a|)$

where $\mathit{numPos}(a, i, j)$ is the number of positive elements in a between $a[i]$ (inclusive) and $a[j]$ (exclusive), and similar for numNeg. For example, $\mathit{numPos}([1, -2, 3, 4], 0, 3) = 2$, $\mathit{numPos}([1, -2, 3, 4], 0, 4) = 3$ and $\mathit{numNeg}([1, -2, 3, 4], 1, 3) = 1$.

Write a loop invariant for the loop that would allow you to prove this postcondition.

You may answer this question in your written answers, or solve it in Dafny (in `posneg.dfy`) by adding an `invariant` to the while loop in `eqPosNeg` so that verification succeeds.

Hint: Your loop invariant will need to be quite a bit stronger than the postcondition for the function. Think about the weakest precondition of $e := n = p$ and the function's postcondition. This wp needs to be the postcondition of your loop. You will still need to strengthen it a little, but this gets you closer.

3 Weakest Preconditions with Array Assignments

Task 3.1 (Written, 6 points).

Calculate the following weakest liberal preconditions. Show your work.

a) $wlp(a[x = 0 \,?\, i : j] := 1, a[i] = 1)$

b) $wlp(a[i] := 5, a[a[1]] = 5)$

c) $wlp(a[j] := a[i] + 1, a[j] > a[i])$

a)

$$\begin{aligned} & wlp(a[x = 0 \,?\, i : j] := 1, a[i] = 1) \\ = \; & [1/a[x = 0 \,?\, i : j]](a[i] = 1) \\ = \; & ((i = x = 0 \,?\, i : j) \,?\, 1 : a[i]) = 1 \end{aligned}$$

b)

$$\begin{aligned} & wlp(a[i] := 5, a[a[1]] = 5) \\ = \; & [5/a[i]](a[a[1]] = 5) \\ = \; & (e = i \,?\, 5 : a[e]) = 5 \qquad\qquad e = 1 = i \,?\, 5 : a[1] \\ = \; & (((1 = i \,?\, 5 : a[1]) = i) \,?\, 5 : a[1 = i \,?\, 5 : a[1]]) = 5 \end{aligned}$$

c)

$$\begin{aligned} & wlp(a[j] := a[i] + 1, a[j] > a[i]) \\ = \; & [a[i] + 1/a[j]](a[j] > a[i]) \\ = \; & (a[i] + 1) > (i = j \,?\, a[i] + 1 : a[i]) \\ (\Leftrightarrow \; & i = j \,?\, (a[i] + 1 > a[i] + 1) : (a[i] + 1 > a[i])) \\ (\Leftrightarrow \; & i = j \,?\, F : T) \\ (\Leftrightarrow \; & i \neq j) \end{aligned}$$

4 One more wrap-up question

Task 4.1 (Written, 0 points).

How long (approximately) did you spend on this homework, in total hours of actual working time? Your honest feedback will help us with future homeworks.