IIT CS536: Science of Programming

Homework 5: Loop Invariants and Proof Outlines

Prof. Stefan Muller

Out: Wednesday, Oct. 25
Due: Monday, Nov. 6, 11:59pm CDT

Updated Oct. 27

This assignment contains 3 written task(s) and 2 task(s) that you can solve on paper or in Dafny, for a total of 50 points.

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 (`notin.dfy`, `mostlyPos.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, 10 points).

Let's take our not-quite-factorial program from Homework 3 and try to prove that it computes the factorial of x (as you may have noticed, it doesn't). The following is an attempt at a minimal proof outline (we've added an initial value of i to make the program slightly closer to correct).

$\{x \geq 0\}$
$i := \overline{0};$
$\{\textbf{inv}\ i \leq x \wedge x = i!\}$
while $i < x$ do
  $x := x * i$
  $i := i + \overline{1}$
od
$\{\exists k.x = k!\}$

Apply the algorithm from class to turn the minimal proof outline into a full proof outline. The bug(s) in the program will show up as one or more predicate logic proof obligations that can't be proven. List **all** of the proof obligations that you need to prove in your full proof outline and briefly (1-2 sentences for each) explain why each one is or is not provable.

Task 1.2 (Written, 12 points).

We fixed the program so it actually computes $x!$ (but now returns it as r). *Updated 10/27*: Fixed postcondition.

$\{x \geq 0\}$
$i := \overline{0};$
$r := \overline{1};$
while $i < x$ do
  $i := i + \overline{1};$
  $r := r * i$
od
$\{r = x!\}$

Write a **full proof outline** for the program, with the given pre- and post-conditions. List **all** of the proof obligations that you need to prove in your full proof outline and briefly explain (using predicate logic and arithmetic) why each one holds. Note that you will need to provide a suitable loop invariant.

More tasks on the next page!

2 Proofs with Loops

Task 2.1 (Programming, 16 points).

a) Write a program (in IMP in your written answers, or in Dafny in `notin.dfy`) that takes an array a of integers and sets r to an integer that **is not in the array**. Formally, your program should meet the pre- and post-condition below:

Precondition: T
Postcondition: $\forall i \in \mathbb{Z}.(0 \leq i < |a|) \rightarrow a[i] \neq r$.

Your program should always terminate and never raise an error (i.e., the postcondition above should hold in a total correctness sense), but you don't need to prove this.

b) For any while loops in your program (you will certainly need at least one), write a correct loop invariant (i.e., it must hold at the beginning of the loop, imply the postcondition at the end of the loop, and be preserved by the loop body).

Write your answer in your written answer or provide it as an `invariant` annotation in the `notin` method in `notin.dfy`.

Hint: You can use a loop invariant that proves something considerably stronger than the actual postcondition we want, as long as it's true at the beginning of the loop and preserved by the loop body.

Task 2.2 (Programming, 12 points).

The following program takes an array a and sets b to true if and only if more than half of the elements in the array are greater than 0.

$i := \overline{0}$;
$n := \overline{0}$;
while $(i < |a|)$ do
  if $a[i] > \overline{0}$ then $n := n + \overline{1}$ else skip fi;
  $i := i + \overline{1}$
od
$b := n > size(a)/2$

Precondition: T
Postcondition: $b \Leftrightarrow numPos(a, 0, |a|) > |a|/2$

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

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 `mostlyPos.dfy`) by adding an `invariant` to the while loop in `mostlyPos` 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 $b := n > size(a)/2$ 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 One more wrap-up question

Task 3.1 (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.