CSCI 2600 Principles of Software

Homework 2: Reasoning About Loops

Due: June 26 @ 11:59pm

Introduction

In this assignment you will prove correctness of loops using the techniques we discussed in class.

Submission Instructions

- Follow the directions in the version control handout for cloning your hw02 git repo.
- Submit your Dafny code as `problem1.dfy`, `problem2.dfy`, and `problem3.dfy` files in the `answers/` directory of your repository.
- Submit your answers in a single .PDF file named `hw2_answers.pdf` in the `answers/` directory of your repository.
  You MUST type up your answers. Handwritten solutions will not be accepted or graded, even if they are scanned into a PDF file.
  We recommend using LaTeX. If you have never used LaTeX, take a look at this tutorial.
- Be sure to commit and push the file to Submitty. Follow the directions in the version control handout for adding and committing files.
- Important: You must press the **Grade My Repository** button for your answers to be graded. If you do not, they will not be graded and you will receive a zero for this homework.

Problems

Problem 1 (15 pts): Exponentiation by squaring

Below is the pseudocode for exponentiation by squaring.

```
Precondition: n >= 0
int power(int m, int n) {
  int x = m;
  int y = n;
  int result = 1;
  while (y != 0) {
    if (y is even) {
      x = x*x;
      y = y/2;
    }
    else {
      result = result*x;
      y = y-1;
    }
  }
  return result;
}
Postcondition: result = m^n
```

a) Find a suitable loop invariant. (3 pts)

b) Show that the invariant holds before the loop (base case). (1 pt)

c) Show by induction that if the invariant holds after k-th iteration, and execution takes a k+1-st iteration, the invariant still holds (inductive step). (6 pts)

d) Show that the loop exit condition and the loop invariant imply the postcondition result = m^n. (1 pt)

e) Find a suitable decrementing function. Show that the function decreases at each iteration and that when it reaches the minimum the loop is exited. (2 pts)

f) Implement exponentiation by squaring in Dafny. (2 pts, autograded)

- **Do NOT** include method `Main` in your Dafny code that you submit on Submitty. If you use `Main` method for testing, make sure you comment it out before submitting on Submitty.
- Method that implements exponentiation by squaring must be named `power` and have the following header:
  `method power(m: int, n: int) returns (result: int)`
- You might need to define additional functions or predicates to be used in conditions.
- To help Dafny prove that squaring the base and halving the exponent doesn't change the value, define the following lemma:

```
lemma square_and_halve(m:int, n:int)
  requires n >= 0
  ensures exp(m, 2 * n) == exp(m * m, n);
{
  if (n != 0)
  {
    square_and_halve(m, n - 1);
  }
}
```

  Then use it in your code:

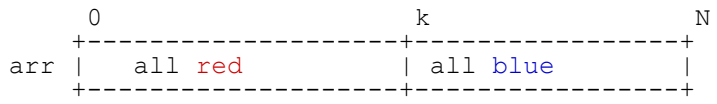
```
if y % 2 == 0 {
{
  square_and_halve(x, y / 2);
  .
  .
  .
}
```

- Make sure to include the precondition and the postcondition, as well as your invariant and the decrementing function.
- Verify your code with Dafny before submitting.
- Submit your Dafny code as a file named problem1.dfy in the answers/ folder.

Problem 2 (14 pts) The Simplified Dutch National Flag Problem

Given an array arr[0..N-1] where each of the elements can be classified as red or blue, write pseudocode to rearrange the elements of arr so that all occurrences of blue come after all occurrences of red and the variable k indicates the boundary between the regions. That is, all arr[0..k-1] elements will be red and elements arr[k..N-1] will be blue. You might need to define method swap(arr, i, j) which swaps the ith and jth elements of arr. (6 pts)

The following picture illustrates the condition of the array at exit.



Write an expression for the postcondition. (2 pts)

Write a suitable loop invariant for all loops in your pseudocode. (4 pts)

Implement your pseudocode in Dafny. (2 pts, autograded)

- **Do NOT** include method `Main` in your Dafny code that you submit on Submitty. If you use `Main` method for testing, make sure you comment it out before submitting on Submitty.
- Represent each element of `arr` as either character 'r' (red) or character 'b' (blue).
- Method that solves the Simplified Dutch National Flag Problem must be named `dutch` and have the following header:
  `method dutch(arr: array?<char>) returns (k: int)`
  This method modifies `arr` and returns the value of `k`.
- Methods that modify `arr` might require a `modifies arr` annotation.
- You may find slicing and concatenating arrays useful. See Dafny Tutorial for more details and examples.
- In cases when your method changes the array, your conditions can refer to both new (current) and the old (previous) values by using the `old` keyword. For example, `arr[..] == old(arr[..])` means that `arr` has not changed.
- Make sure to include the precondition and the postcondition, as well as your invariant and the decrementing function.
- Verify your code with Dafny before submitting.
- Submit your Dafny code as a file named problem2.dfy in the answers/ folder.

Problem 3 (20 pts): Additive Factorial

Below we give, in Dafny syntax, the factorial function and a method with loops, which should be computing the factorial of a number.

Fill in the annotations at the designated places. You can use function `Factorial` in annotations. Fill in the two loop invariants and the assertion.

```
function Factorial(n: int): int
  requires n >= 0
{
  if n == 0 then 1 else n * Factorial(n-1)
}

method LoopyFactorial(n: int) returns (u: int)
  requires n >= 0
  ensures u == Factorial(n)
  {
    u := 1;
    var r := 0;
    while (r < n)
      invariant // FILL IN YOUR INVARIANT HERE
    {
      var v := u;
      var s := 1;
      while (s<=r)
        invariant // FILL IN YOUR INVARIANT HERE
      {
        u:=u+v;
        s:=s+1;
      }
      r:=r+1;
      assert // FILL IN YOUR ASSERTION HERE
    }
  }
```

Use computational induction to prove partial correctness. There are two loops. Prove the inner loop first. Assume the outer loop invariant to prove the inner loop invariant and then use the result of inner loop invariant and loop exit condition to prove the outer.

Verify your Dafny code.

- **Do NOT** include method `Main` in your Dafny code that you submit on Submitty. If you use `Main` method for

testing, make sure you comment it out before submitting on Submitty.
- Method that implements additive factorial must be named `LoopyFactorial` and have the following header: `method LoopyFactorial(n: int) returns (u: int)`
- Verify your code with Dafny before submitting.
- Submit your Dafny code as a file named problem3.dfy in the answers/ folder.

8 points – Invariants and assertion
2 points – Proof for the base case of inner loop
3 points – Proof for the inner loop induction
2 points – Proof for the outer loop base case
3 points – Proof for the outer loop induction
2 points (autograded) – Dafny code verification

Collaboration (0.5 pts)

Please answer the following questions in a file named `collaboration.pdf` in your `hw2/answers/` directory.

The standard academic integrity policy applies to this homework.

State whether or not you collaborated with other students. If you did collaborate with other students, state their names and a brief description of how you collaborated.

Reflection (0.5 pts)

Please answer the following questions in a file named `reflection.pdf` in your `hw2/answers/` directory. Answer briefly, but in enough detail to help you improve your own practice via introspection and to enable me to improve Principles of Software in the future.

1. In retrospect, what could you have done better to reduce the time you spent solving this homework?
2. What could I (the instructor) have done better to improve your learning experience in this homework?
3. What do you know now that you did not know before beginning the homework?

Submission

Push your repo containing the following files to Submitty:

- `hw2/answers/problem1.dfy`
- `hw2/answers/problem2.dfy`
- `hw2/answers/problem3.dfy`
- `hw2/answers/hw2_answers.pdf`
- `hw2/answers/collaboration.pdf`
- `hw2/answers/reflection.pdf`

Hints

- When trying to come up with a loop invariant for prewritten code, it often helps to trace through the execution of the code on paper. Choose a few different starting values of variables defined outside the block of code (such as method arguments), and write down the values of all the variables used in the loop for each iteration.
- Your Dafny code will be autograded by Submitty. If you are not getting full credit from the autograder, make sure to click "Show Details" in the autograding section to check the autograder output.

Errata

Check the Submitty Forum for possible updates or corrections.