Homework Problem Set #1

CS 598MP3: Due Sep 27

Problem 1.

Consider the Dafny program provided on the website: *Product1, Product2, Half, and DividesIt*. Prove them correct using Dafny by writing down appropriate loop invariants. You are allowed to only add (side-effect free) annotations to the code— you cannot change the executing code in any way (not even in trivial ways that may seem correct to you). Submit your program (with proof that Dafny verified it) using a hardcopy printout, and also email your (verified) programs to madhu@illinois.edu in a tarball.

A few notes:

- Product1 and Product2 are inefficient ways to compute the product. Aim is to get you started thinking about inductive invariants and writing them formally in logic and in Dafny. Note that we are using the type nat (natural numbers), a subtype of integers, rather than integers here. Dafny will hence check that every assignment to a natural number necessarily assigns a non-negative number. Also, in Dafny, int and nat types are pure mathematical integers and natural numbers, and don't overflow, etc.

- Half is a program that uses integer division and asks you to write invariants using them. Note the use of "/" (integer division) and "%" (modulo) operators.

- In DividesIt, Dafny needs a lemma to prove the program correct. Lemmas are written as procedures. We have already provided the lemma (make sure you understand it and you agree it's valid). We don't prove the lemma but just assume it. Also, this program has a decreases clause to prove the loop terminates— ignore understanding this for now. But keep it there so that Dafny does not complain. In the other programs above, Dafny actually uses a simple heuristic to reason that the loops terminate, and hence doesn't require a decreases clause.

Problem 2

Consider the following program.

```
int pm(int x, int y)
@requires x >= 0 & y >= 0;
@ensures res = x * y
{
  a := x;
  b := y;
  res := 0;
  while (a > 0) {
    if ((a mod 2) = 1) then res := res + b;
    a := a div 2;
    @assert a >= 0;
    b := b * 2;
  }
  return res;
}
```

The above code implements the algorithm for Peasant's multiplication (aka Egyptian multiplication, an ancient Egyptian algorithm that shows how you can do multiplication by just multiplying by 2 and doing addition[1]). In the above, div denotes integer division— i.e., $a \; div \; b = \lfloor \frac{a}{b} \rfloor$.

- Write down an inductive loop invariant for the while loop that will prove the program correct.
- Write down the 6 basic blocks for the above program, using your loop invariant annotation. (Note that there is an assertion and a postcondition in the program above.)
- For each basic block, write down the verification conditions (you can use any method for writing verification conditions).
- Show that each verification condition is valid using an SMT solver (either Z3 or CVC4), to conclude the program is correct.

[1] Why do we teach kids "multiplication tables" up to 9?