Assignment 0

Theodore S. Norvell

6892 Due 2014 Sept 18

Q0 [25] Division

(a) [10] List all the boolean expressions that need to be shown to be universally true in order to check that this proof outline is correct. All variables hold natural numbers. The notation $d|x$ means d divides x, i.e., there exists a natural number q such that $d \times q = x$.

$\{d > 0\}$
$r := a$
$\{d|(a-r)\}$
while $r \geq d$ do
    $\{d|(a-r) \wedge r \geq d\}$
    $e := d$
    $\{d|(a-r) \wedge d|e\}$
    while $r \geq e$ do
        $\{d|(a-r) \wedge d|e \wedge r \geq e\}$
        $r := r - e$
        $\{d|(a-r) \wedge d|\,(2 \times e)\}$
        $e := 2 \times e$
    end while
end while
$\{d|(a-r) \wedge r < d\}$

(b) [10] For each of the expressions from part (a), explain why it is universally true, or why it is not. Some useful facts about the divides relation include

$$\begin{array}{rcl} & & d|d \\ & & d|0 \\ d|x \wedge d|y & \Rightarrow & d|(x-y) \\ d|x & \Rightarrow & d|(2 \times x) \end{array}$$

(c) [5] As you can see, the algorithm above calculates the remainder of a divided by d, using only adders, subtractors, and comparators. Modify the algorithm to also compute the quotient, use only adders, subtractors, and comparators. Be sure to modify the assertions, as well as the executable code, so that the modified proof outline is correct. Present the modified proof outline. (You don't need to hand in proof that the modified outline is correct, but you should satisfy yourself that it is.) The modified postcondition should be

$$d|(a-r) \wedge r < d \wedge d \times q + r = a$$

Q1 [20]

The Fibonacci function is such that

$$\begin{aligned} \text{fib}(0) &= 0 \\ \text{fib}(1) &= 1 \\ \text{fib}(i+2) &= \text{fib}(i) + \text{fib}(i+1), \text{ for all } i \in \mathbb{N} \end{aligned}$$

Develop a program to efficiently[0] compute the value of fib(k), for any $k > 0$. Present a correct proof outline and show that it is correct.

Bonus [6].

(a)[3] Consider a new kind of command: the nondeterministic assignment statement. A nondeterministic assignment statement looks like this $x :\in S$ —pronounced "x becomes an element of S"— where x is a variable and S is a set. The meaning is that an arbitrary member of S is assigned to x. Give a rule that reduces the programming question

$$\text{Is } \{P\}\ x :\in S\ \{Q\} \text{ correct?}$$

to a mathematical question about universal truth.

(b)[3] The Devil gives Faust a chance to win his soul back. The Devil produces a pile of 100 pebbles. The players take turns removing anywhere from 1 to 9 (inclusive) pebbles. The game ends when the pile is empty. The last player to move wins Faust's soul. Being a gentleman, the Devil allows Faust to move first.

We can model the game as a program:

```
f := true // Faust to move
p := 100
while p ≠ 0 do
    if f then
        r :∈ {1, .., max(9, p)} // Faust's move
        p := p − r
    else
        r :∈ {1, .., max(9, p)} // The Devil's move
        p := p − r
    end if
    f := ¬f
end while
```

Propose a condition Q such that Q will be true after $f := \text{true}\ p := 100$ is executed and such that $p = 0 \wedge Q \Rightarrow f$ is universally true. A consequence is that, if Q were a loop invariant, then f would be a postcondition, meaning that the Devil wins. Now modify the Devil's move —staying within the rules— so that Q is preserved by the loop's body.

[0] If your program is such that the number of addition operations that it does is roughly proportional to k, it is efficient enough.