

## Exercise sheet 05

**Deadline:** May 27, 8:00 p.m.

Please **submit 2 files**. The first file, **ex05\_your\_name.dfy**, should be a Dafny-file containing the answers to Problems 1, 2 and 3. The second file should be a pdf-file **ex05\_your\_name.pdf**, containing your solution to problem 5.

**Problem 1** (Division by repeated subtraction). The well known *Fibonacci*-Function is defined below as recursive Function `fib`. We have also supplied an iterative method `fibIter`, which is supposed to calculate the same function, using a loop instead of recursion.

```
function fib(n:nat):nat
{ if n < 2 then n else fib(n-2)+ fib(n-1)}

method fibIter(n:int) returns (a:int)
requires
ensures
a == fib(n)
{
    | a := 0;
    var b,x := 1,0;
    while x < n
    invariant
    invariant
    {
        a,b,x := b,a+b,x+1;
    }
}
```

Complete `fibIter` with appropriate pre- and postconditions and with appropriate invariants, so that it verifies in Dafny. The Dafny-solution uses parallel assignments. Many languages do not allow parallel assignments. Therefore, you should replace the parallel assignments with single assignments.

(4 points)

**Problem 2.** The *factorial* function `fact` is defined as

```
function fact(n:nat):nat
{ if n==0 then 1 else n*fact(n-1)}
```

Write a method `factIter` which calculates `fact` iteratively and let Dafny prove correctness.

(3 points)

**Problem 3.** The greatest common divisor `gcd` is defined recursively as shown.

```
function gcd(m:nat,n:nat):nat
requires m > 0 && n > 0
{ if m==n then m else if m>n then gcd(m-n,n) else gcd(m,n-m)}

method gcdI(m:int,n:int) returns (g:int)
```

Write a method `gcdI` which calculates `gcd` iteratively using a while-loop, and let Dafny prove correctness. Additional to the `invariant(s)`, Dafny may require a `decreases` term, which is an integer expression that decreases in every pass through the loop, while remaining positive.

(3 points)

**Problem 4** (Floyd follows from Hoare). (a). Consider proposition  $Q \equiv x^2 + v \leq x * v$ , term  $t \equiv x + v$ , and variables  $x, y$  and  $v$ . Calculate concretely  $(Q[t/v])[y/v]$  as well as  $Q[(t[y/v])/v]$ . Show all intermediate steps!

(b). Argue, why in general for arbitrary Proposition  $Q$  and for arbitrary terms  $t, s$  and for arbitrary variable  $v$  one must have equality:  $(Q[t/v])[s/v] \equiv Q[(t[s/v])/v]$ . You can either indicate how to prove this inductively, or you can argue with the shape of  $Q, t$ , and  $s$  when represented syntactically as trees.

(c). Using Floyd's rule  $\{P\}v := t\{\exists v_0.(P[v_0/v] \wedge v = t[v_0/v])\}$ , derive Hoare's rule  $\frac{}{\{Q[t/v]\}v := t\{Q\}}$ . Justify every step.

(6 points)