CS 310, Assignment 6

Answers

Verify the validity of the following correctness statements by adding all the intermediate assertions and so producing the proof tableau. State all the mathematical facts used. All variables are of type `int`.

1.
```
ASSERT(x == y*(y+1)/2)
y = y + 1;
x = x + y;
ASSERT(x == y*(y+1)/2)
```

ANSWER:

```
ASSERT(x == y*(y+1)/2)              // (3) simple math, qed
ASSERT(x == (y+1)*(y/2+2/2-1))
ASSERT(x == (y+1)*((y+2)/2-1))
ASSERT(x == (y+1)*(y+2)/2)-(y+1)
ASSERT(x+y+1 == (y+1)*(y+2)/2)
ASSERT(x+y+1 == (y+1)*(y+1+1)/2)    // (2) assignment
y = y + 1;
ASSERT(x+y == y*(y+1)/2)            // (1) assignment
x = x + y;
ASSERT(x == y*(y+1)/2)              // start here
```

2.
```
ASSERT(true)
if (x >= y) x = y - 1; else y = y + 1;
z = y + 1;
ASSERT(x < y < z)
```

ANSWER:

```
ASSERT( true )                          // (12) if, qed
if (x >= y)
    ASSERT( true && x >= y )            // (11) strengthening
    ASSERT( true )                      // (10) math
    FACT( y - 1 < y )
    ASSERT( y - 1 < y )                 // (9)  assignment axiom
```

```
        x = y - 1;
        ASSERT( x < y )                    // (3)  if
    else
        ASSERT( true && !(x >= y) )        // (8)  math
        ASSERT( !(x >= y) )                // (7)  strengthening
        ASSERT( !(x >= y + 1) )            // (6)  math
        FACT( !(x < y + 1) == x >= y + 1 )
        ASSERT( !!(x < y + 1) )            // (5)  math
        FACT( ForAll(s) !!s == s )
        ASSERT( x < y + 1 )                // (4)  assignment axiom
        y = y + 1;
        ASSERT( x < y )                    // (3)  if
    ASSERT( x < y )                        // (2)  math
    FACT( y < y + 1 )
    ASSERT( x < y < y + 1 )                // (1)  assignment axiom
    z = y + 1;
    ASSERT( x < y < z )                    //      start here
```

3.
```
ASSERT(x == x0)
int sign = 1;
if (x < 0) sign = -1;
x = x * sign;
ASSERT(|x| == |x0| && x >= 0)
```

The notation `|x|` denotes as usual the absolute value of `x`.

ANSWER:

```
ASSERT(x == x0)                                    // (15) strengthen, qed
FACT(x == x0 => |x| == |x0|
ASSERT(|x| == |x0|)                                // (14) math
ASSERT(|x| == |x0| && 1 == 1)                      // (13) assignment
int sign = 1;
ASSERT(|x| == |x0| && sign == 1)                   // (12) if
if (x < 0)
    ASSERT(|x| == |x0| && sign == 1 && x < 0)      // (11) strengthen
    ASSERT(|x| == |x0| && x < 0)                   // (6) strengthen
    FACT(|x| == |x0| iff |-x| == |x0|)
    ASSERT(|x*-1|==|x0| && x < 0)                  // (5) strengthen
    ASSERT(|x*-1|==|x0| && x <= 0)                 // (4) math
    ASSERT(|x*-1|==|x0| && x*-1 >= 0)              // (3) assignment
    sign = -1;
    ASSERT(|x*sign| == |x0| && x*sign >= 0)        // (2) if
else
```

```
    ASSERT(|x| == |x0| && sign == 1 && x >= 0)        // (10) math
    ASSERT(|x*sign| == |x0| && sign == 1 && x >= 0)   // (9) strengthen
    ASSERT(|x*sign| == |x0| && sign >= 0 && x >= 0)   // (8) math
    ASSERT(|x*sign| == |x0| && x*sign >= 0 && x >= 0) // (7) strengthen
    ASSERT(|x*sign| == |x0| && x*sign >= 0)           // (2) if
ASSERT(|x*sign| == |x0| && x*sign >= 0)               // (1) assignment
x = x * sign;
ASSERT(|x| == |x0| && x >= 0)
```
