Principles of Software HW 1

Joel Grimaldi

February 2021

1

1. option 1
2. option 3
3. option 1
4. option 1
5. option 2
6. option 2

2

1. valid

2. invalid
-in the case that x+1 = N in the precondition, the postcondition would be false as x would be equal to N+1.
-to solve this problem, make the postcondition: x <= N+1

3. invalid
-in the case that i=0 and j=1, after changes i=1 and j=0, which do not add to 0.
-to solve this problem, make the postcondition: i+j != 0

4. invalid
-in the case that x=y, m will equal y but x will not be less than y.
-to solve this problem, change that side of
the postcondition to: (m = y AND x <= y)

3

1. possibly invalid
-we got no information on A or E with the only proven code being
{F} code {B}

2. valid
-we know that {F} code {B} is true
-B=>D, therefore {F} code {D} is true
-D=>F, therefore {D} code {B} is true
-C=>D, therefore {C} code {B} is true
-B=>D, therefore {C} code {D} is true

4

1.
{ x > 0 }
    x = 10;
{ x = 10 }
    y = 20 - x;
        { x = y = 10 }
    z = y + 4;
        { x = y = 10 && z = 14 }
    y = 0;
        { x = 10 && y = 0 && z = 14 }

2.
{ $|x| > 11$ }
    x = -x;
        { $|x| > 11$ }
    x = x * x;
        { $x \geq 144$ }
    x = x + 1;
        { $x \geq 145$ }

3.
{ $|x| < 5$ }
    if (x > 0) {
    { $0 < x < 5$ }
        y = x + 2;
            { $0 < x < 5$ && $2 < y < 7$ }
    } else {
    { $-5 < x \leq 0$ }
        y = x - 1;
            { $-5 < x \leq 0$ && $-6 < y \leq -1$ }
    }
{ $-5 < x < 5$ && $-6 < y < 7$ }

5

1.
```
{ wp(x = -5, y > -2x) = (y > 10) }
    x = -5;
        { wp(z = 2 * x + y, z > 0) = (2x + y > 0) = (y > -2x) }
    z = 2 * x + y;
        { z > 0 }
```

2.
```
{ wp( x > 1, x < -3 ) = { x < -3 } or { x > 1 } }
    if (x > 0) {
    { wp( x = x + 6, x > 7 ) = ( x + 6 > 7 ) = ( x > 1 ) }
        x = x + 6;
    else {
    { wp( x = 4 - x, x > 7 ) = ( 4 - x > 7 ) = ( -x > 3 ) = ( x < -3 ) }
        x = 4 - x;
    }
{ x > 7 }
```

3.
```
{ x > -1 }
    if (x > 4) {
    { wp( x = x - 3, x > 0 ) = ( x - 3 > 0 ) = (x > 3) }
        x = x -3;
    } else {
    { x > -1 }
        if (x < - 4) {
        { wp( x = x + 3, x > 0 ) = ( x + 3 > 0 ) = ( x > -3 ) }
            x = x + 3;
        } else {
        { wp( x = x + 1, x > 0 ) = ( x + 1 > 0 ) = ( x > -1 ) }
            x = x + 1;
        }
    }
{ x > 0 }
```

4.
```
{ wp( x = y + 2, x > 2y - 1 ) = ( 2y - 1 < y + 2 ) = ( 2y < y + 3 )
= ( y < 3 ) }
    x = y + 2;
        { wp (z = x + 1, z > 2y ) = ( x + 1 > 2y ) = ( x > 2y - 1 ) }
    z = x + 1;
{ z > 2y }
```

5.

```
{ x ≠ 0 && x ≠ -1 }
    if (x >= 0)
    { wp( z = x, z ≠ 0 ) = ( x ≠ 0 ) }
        z = x;
    else
    { wp( z = x + 1, z ≠ 0 ) = ( x + 1 ≠ 0 ) = ( x ≠ -1 ) }
        z = x + 1;
{ z ≠ 0 }
```

6

1.
{ $x_{pre} < y_{pre}$ }
{ wp(w = y, $y_{post} = x_{pre}$) = Never satisfied }
w = y;
{ wp(z = x + y - w, $y_{post} = x_{pre}$) = Z is irrelevant}
z = x + y - w;
{ wp(x = w, $x_{post} = y_{pre}$) = (w = y_{pre}) }
x = w;
{ $x_{post} = y_{pre}$ && $y_{post} = x_{pre}$ }

Sufficient or Insufficient: Insufficient

This is because y is never reassigned and if $x_{pre} < y_{pre}$ there is no way for $y_{post} = x_{pre}$ ever. A precondition that would allow the postcondition to work would be { $x_{pre} = y_{pre}$ }

2.
{ (x = y) or (x $\neq$ y && y > 0) }
{ wp(x == y, y $\geq$ 0) = (x $\geq$ 0 && y $\geq$ 0) }
{ wp(x $\neq$ y, x $\leq$ 1) = (x $\leq$ 1) }
if (x == y)
{ wp(x = 0, x $\leq$ y) = (y $\geq$ 0) }
x = 0;
else
{ wp(x = x * y, x $\leq$ y) = (x * y $\leq$ y) = (x $\leq$ 1) }
x = x * y;
{ x $\leq$ y }

Sufficient or Insufficient: Insufficient

This is because in the case where x = y but x and y are negative. x will be set to 0 and will end up > y. A way that the precondition can be altered to always make the postcondition correct is:
{ (x = y && x > 0) or (x $\leq$ 0 && y > 0) }