Lab: Exploring Hoare Logic

Juncheng Zhao, Havin Lim

February 14th 2023

1 Problem 1: Have a Go About It

Consider the following Python code snippet:

```
# w is previously defined

x = w
w = w * 2
y = w
w = w * 2
z = w
```

Use Hoare logic to show that if w ¿ 0 then $z - y - x > 0$.

Proof: We use following Hoare logic derivation to derive a precondition suitable to achieve the desired postcondition, $z - y - x > 0$:

```
# {{ (w * 2 * 2) - (w * 2) - w > 0 }}
x = w
# {{ (w * 2 * 2) - (w * 2) - x > 0 }}
w = w * 2
# {{ (w * 2) - w - x > 0 }}
y = w
# {{ (w * 2) - y - x > 0 }}
w = w * 2
# {{ w - y - x > 0 }}
z = w
# {{ z - y - x > 0 }}
```

We can simplify our derived precondition as follows:

```
(w * 2 * 2) - (w * 2) - w > 0
-->* 4w - 2w - w >0
--> w > 0
```

Thus, our postcondition holds given our precondition.

2 Probelm 2: Swap Swap Swap

Consider the following Python code snippet:

```
x = 0
y = 1
z = x
x = y
y = z
```

Proof: here is the proof to derive that the code snippet swaps the contents of x and y.

The desired postcondition: x = 1, y = 0

```
#{{void}}
x = 0
#{{x = 0}}
y = 1
#{{x = 0, y = 1}}
z = x
#{{y = 1, z = 0}}
x = y
#{{x = 1, z = 0}}
y = z
#{{x = 1, y = 0}}
```

Therefore, using the Hoare logic we can prove the code snippet switches the content of x and y.

3 Problem 3: Reasoning About Conditionals

Consider the following code snippet:

```
# x is previously defined

if x == 0:
    y = 1
else:
    y = 5
```

Use Hoare logic to prove that y ¿ 0 after the conditional finishes executing.

Proof: We use following Hoare logic derivation to derive a precondition suitable to achieve the desired postcondition, $y > 0$, we can use the conditional rule and apply it to the two branches of the conditional

```
if x == 0:
    y = 1
else:
    y = 5
# {{ y > 0 }}

For the "if" branch where x is equal to 0
#{{ x == 0 }}
y = 1
#{{ y > 0 }}

For the "else" branch where x is not 0:
#{{ x != 0}}
y = 5
#{{ y > 0}}

Therefore, using the Conditional rule, we can conclude that
#{{x == 0 or x != 0}}
if x == 0:
y = 1
else:
y = 5
#{{y > 0}}
```

4 Problem 4: upgrading conditionals

```
a. the postcondition holds for both branches of the snippet as we could ensure 2 > 0 and x +

 b.
 For the "if" branch where x is equal to 0
 #{{ x == 0}}
 y = 2
 #{{ y >= x}}

 For the "else" branch where
 #{{ x != 0}}
 y = x + 1
 #{{y >= x}}

Therefore, using the Conditional rule, we can conclude that
#{{x == 0 or x != 0}}
if x == 0:
y = 2
else:
y = x + 1
```

```
#{{y >= x}} (incorrect)
 We can prove that the postcondition holds for the "else" branch by using the conditional ru


 c. To prove the postcondition, we need to provide additional information about the precondi

 The formal proof looks like
 if branch:
 #{{P and x == 0}}
 if x == 0:
 y = 2
 #{ x <= y}
 else:
 #{{False}}
 else branch:
 {{P and x != 0}}
 if x != 0:
 y = x + 1
 #{{x <= y}}
 else:
 #{{False}}
```