Lab 10: Axiomatic Semantics

CSI 3120

Sample Solutions

Recall the Rules of Hoare Logic

$$\frac{}{\{Q_{x \to E}\}\ \texttt{x := E}\ \{Q\}}\ \text{Assignment}$$

$$\frac{\{P\}S\{Q\} \qquad P' \Rightarrow P \qquad Q \Rightarrow Q'}{\{P'\}S\{Q'\}}\ \text{Consequence}$$

$$\frac{\{P_1\}S_1\{P_2\} \qquad \{P_2\}S_2\{P_3\}}{\{P_1\}S_1; S_2\{P_3\}}\ \text{Sequence}$$

$$\frac{\{B \textit{ and } P\}S_1\{Q\} \qquad \{(\textit{not } B) \textit{ and } P\}S_2\{Q\}}{\{P\}\ \texttt{if}\ B\ S_1\ \texttt{else}\ S_2\ \{Q\}}\ \text{If Rule}$$

$$\frac{\{I \land B\}S\{I\}}{\{I\}\ \texttt{while}\ B\ \texttt{do}\ S\ \texttt{end}\{I \textit{ and } (\textit{not } B)\}}\ \text{While Rule}$$

Problem 1

(a) Find the weakest precondition of the following "if" statement, using the given postcondition.


```
if x > y then z := x else z := y
```

$\{x > 0 \textit{ and } y > 0 \textit{ and } z \geq x \textit{ and } z \geq y\}$

(b) Bonus (save for last): Prove the following Hoare triple. (In other words, prove that this program is correct.)

$\{x > 0 \textit{ and } y > 0\}$

```
if x > y then z := x else z := y
```

$\{x > 0 \textit{ and } y > 0 \textit{ and } z \geq x \textit{ and } z \geq y\}$

Solution to Problem 1(a)

Following the steps in the course notes, page 12, we have:

$\{x > 0 \text{ and } y > 0 \text{ and } x \geq x \text{ and } x \geq y\}$ `z := x` $\{x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y\}$

$\{x > 0 \text{ and } y > 0 \text{ and } y \geq x \text{ and } y \geq y\}$ `z := y` $\{x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y\}$

Thus
$$P_1 \equiv (x > 0 \text{ and } y > 0 \text{ and } x \geq x \text{ and } x \geq y)$$
$$P_2 \equiv (x > 0 \text{ and } y > 0 \text{ and } y \geq x \text{ and } y \geq y)$$

or equivalently
$$P_1 \equiv (x > 0 \text{ and } y > 0 \text{ and } x \geq y)$$
$$P_2 \equiv (x > 0 \text{ and } y > 0 \text{ and } y \geq x)$$

Again, following page 12, we need to find an assertion P such that $((x > y) \text{ and } P) \Rightarrow P_1$ and $((\text{not } (x > y)) \text{ and } P) \Rightarrow P_2$. In this case, there is a simple one:

$$P \equiv (x > 0 \text{ and } y > 0)$$

Using these formulas, we can now build the complete proof.

Solution to Problem 1(b)

1. $\{x > 0 \text{ and } y > 0 \text{ and } x \geq y\}$ `z := x` $\{x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y\}$
   by Assignment
2. $(x > y \text{ and } (x > 0 \text{ and } y > 0)) \Rightarrow (x > 0 \text{ and } y > 0 \text{ and } x \geq y)$
   by logic and arithmetic
3. $(x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y) \Rightarrow (x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y)$
   by logic
4. $\{(x > y \text{ and } (x > 0 \text{ and } y > 0))\}$ `z := x` $\{x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y\}$
   by Consequence from 1,2,3
5. $\{x > 0 \text{ and } y > 0 \text{ and } y \geq x\}$ `z := y` $\{x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y\}$
   by Assignment
6. $(\textit{not } (x > y)) \text{ and } (x > 0 \text{ and } y > 0)) \Rightarrow (x > 0 \text{ and } y > 0 \text{ and } y \geq x)$
   by logic and arithmetic
7. $\{(\textit{not } (x > y)) \text{ and } (x > 0 \text{ and } y > 0)\}$ `z := y` $\{x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y\}$
   by Consequence from 5,6,3
8. $\{x > 0 \text{ and } y > 0\}$
   `if x > y then z := x else z := y`
   $\{x > 0 \text{ and } y > 0 \text{ and } z \geq x \text{ and } z \geq y\}$
   by If Rule from 4,7

Problem 2

Below is another program to compute factorial (a minor modification of the one proved correct in class).

$\{n \geq 0\}$


```
fact := n;
count := n-1;
while count <> 1 do
        fact := fact * count;
        count := count-1;
end
```


$\{fact = n!\}$

(a) Find a loop invariant for this version that will lead to a proof.

(b) Does the precondition guarantee termination? If not replace the precondition with another one that guarantees termination.

Solution to Problem 2

(a) Here is a program trace with `n`=6 to help us figure out an invariant.

iteration	fact	count
0	6	5
1	30	4
2	120	3
3	360	2
4	720	1

A possible invariant is $fact = \frac{n!}{count!}$.

(b) Note that this program goes into an infinite loop with inputs 1,0, and negative numbers. Note that the precondition rules out negative numbers but allows inputs 0 and 1. To prove termination, as discussed on page 22 of the course notes, we need to strengthen the precondition to $n > 1$. The loop terminates because the value of the expression $count - 1$ eventually reaches 0, and at this point the condition in the `while` statement becomes false.

Solution to Problem 2 (continued)

To illustrate the solution to 2(a), the proof on the next few pages is a proof of "partial correctness" (see page 21 of the course notes). We use the following abbreviations:

$S_1 \equiv$ `fact := n;`

$S_2 \equiv$ `count := n-1;`

$S_4 \equiv$ `fact := fact * count;`

$S_5 \equiv$ `count := count-1;`

$S_3 \equiv$ `while count <> 1 do` S_4 S_5 `end`

Problem 2 Solution (continued)

1. $\{n = \frac{n!}{(n-1)!}\}$ `fact := n` $\{fact = \frac{n!}{(n-1)!}\}$
   by Assignment
3. $(n \geq 0) \Rightarrow (n = \frac{n!}{(n-1)!})$
   by logic and arithmetic
3. $(fact = \frac{n!}{(n-1)!}) \Rightarrow (fact = \frac{n!}{(n-1)!})$
   by logic
4. $\{n \geq 0\}$ `fact := n` $\{fact = \frac{n!}{(n-1)!}\}$
   by Consequence Rule from 1,2,3
5. $\{fact = \frac{n!}{(n-1)!}\}$ `count := n-1` $\{fact = \frac{n!}{count!}\}$
   by Assignment
6. $\{n \geq 0\}S_1; S_2\{fact = \frac{n!}{count!}\}$
   by Sequence Rule from 4,5
7. $\{(fact * count) = \frac{n!}{(count-1)!}\}$ `fact := fact*count` $\{fact = \frac{n!}{(count-1)!}\}$
   by Assignment Rule
8. $(fact = \frac{n!}{count!})$ *and* $(count \neq 1) \Rightarrow (fact * count = \frac{n!}{(count-1)!})$
   by logic and arithmetic
9. $(fact = \frac{n!}{(count-1)!}) \Rightarrow (fact = \frac{n!}{(count-1)!})$
   by logic
10. $\{fact = \frac{n!}{count!}$ *and* $(count \neq 1)\}$ `fact := fact*count` $\{fact = \frac{n!}{(count-1)!}\}$
   by Consequence Rule from 7,8,9

Problem 2 Solution (continued)

11. $\{fact = \frac{n!}{(count-1)!}\}$ `count := count-1` $\{fact = \frac{n!}{count!}\}$

    by Assignment

12. $\{fact = \frac{n!}{count!}$ *and* $(count \neq 1)\} S_4; S_5 \{(fact = \frac{n!}{count!})\}$

    by Sequence Rule from 10,11

13. $\{fact = \frac{n!}{count!}\}$ `while count <> 1 do` $S_4; S_5$ `end` $\{(fact = \frac{n!}{count!})$ *and* $(not(count \neq 1))\}$

    by While Rule from 12

14. $(fact = \frac{n!}{count!}) \Rightarrow (fact = \frac{n!}{count!})$

    by logic

15. $(fact = \frac{n!}{count!})$ *and* $(not(count \neq 1)) \Rightarrow fact = n!$

    by logic and arithmetic

16. $\{fact = \frac{n!}{count!}\}$ `while count <> 1 do` $S_4; S_5$ `end` $\{fact = n!\}$

    by Consequence Rule from 13,14,15

17. $\{n \geq 0\} S_1; S_2; S_3 \{fact = n!\}$

    by Sequence Rule from 6,16

Problem 3

Consider the tail recursive OCaml program below.

```
let mult_tr (a:int) (b:int) =
  let rec mult' (a:int) (b:int) (result:int) =
    if a = 0 then 0
    else if a = 1 then b + result
    else mult' (a-1) b (result+b)
  in
  mult' a b 0
```


Translate this program to an equivalent one that uses a while loop instead of recursion. (See page 62 of the course notes for Chapter 7 of the Mitchell textbook.) Use the programming language that was defined in the course notes on Axiomatic Semantics (assignment statements, if statements, while loops, and sequences of statements separated by a semi-colon). Let P be the name of your program. The following Hoare triple should be true about your program: $\{a \geq 0\}P\{result = a * b\}$. You don't have to prove it. Just make sure that your program is correct, and terminates whenever the precondition is satisfied.

Solution to Problem 3


```
result := 0;
if a = 0 then result := 0
         else (result := b;
               while not (a = 1) do
                 a := a - 1;
                 result := result + b
               end)
```