Bhavesh Rajesh Talreja (A20516822)
CS 536 HW3

1. Hoare triples

Task 1.1

Given: Let s = while i < x do x := x ∗ i; i := i + 1 od
For each of the following, is the triple satisfied or unsatisfied in the given state? Explain why by unfolding the definition as we saw in the class. Note that some of these are partial correctness triples and others are total correctness.

a) {i = 1, x = 6} ⊨ [i < x] s [i = x]

Ans. The definition of the total correctness says,
When σ is the start state, we define σ ⊨ [P]S[Q] as
if σ ⊨ P, then
⟨S, σ⟩ →* ⟨skip, σ'⟩ , and
σ' ⊨ Q

Given: P = [i < x], Q = [i = x], σ = {i = 1, x = 6}

Unfolding the definition,
{i = 1, x = 6} ⊨ [i < x] Yes
⟨S, {i = 1, x = 6}⟩ →* ⟨skip, σ'⟩ , and No because the loop in statement S is an infinite loop.
σ' ⊭ [i = x]

Since the value of x grows very fast as compared to i, and hence i ≠ x as specified in the post condition.

The state σ does not satisfy the Hoare triple.

b) {i = −1, x = 5} ⊨ {i < x} s {i ≥ 0 ∧ x ≤ 0}

Ans. The definition of the partial correctness says,
When σ is the start state, we define σ ⊨ {P}S{Q} as
if σ ⊨ P, and
⟨S, σ⟩ →* ⟨skip, σ'⟩ , then
σ' ⊨ Q

Given: P = {i < x}, Q = {i ≥ 0 ∧ x ≤ 0}, σ = {i = −1, x = 5}

Unfolding the definition,
{i = −1, x = 5} ⊨ { i < x } Yes
⟨S, {i = -1, x = 5}⟩ →* ⟨skip, {i = 0, x = -5}⟩ , then Yes

$\{i = 0, x = -5\} \vDash \{i \geq 0 \wedge x \leq 0\}$ Yes

The state σ satisfy the Hoare triple.

c) $\{i = 1, x = 0\} \vDash \{i < x\}\ s\ \{i = x\}$

Ans. The definition of the partial correctness says,
When σ is the start state, we define $\sigma \vDash \{P\}S\{Q\}$ as
if $\sigma \vDash P$, and
$\langle S, \sigma\rangle \rightarrow^* \langle skip, \sigma'\rangle$, then
$\sigma' \vDash Q$

Given: $P = \{i < x\}$, $Q = \{i = x\}$, $\sigma = \{i = 1, x = 0\}$

Unfolding the definition,
$\{i = 1, x = 0\} \nvDash \{i < x\}$

Since the precondition P is not satisfied by σ, we don't have to check about S & Q as the partial correctness doesn't say anything if $\sigma \nvDash P$.

The state σ satisfy the Hoare triple.

d) $\{i = 1, x = 2, k = 2\} \vDash \{x = k\}\ s\ \{x = k!\}$

Ans. The definition of the partial correctness says,
When σ is the start state, we define $\sigma \vDash \{P\}S\{Q\}$ as
if $\sigma \vDash P$, and
$\langle S, \sigma\rangle \rightarrow^* \langle skip, \sigma'\rangle$, then
$\sigma' \vDash Q$

Given: $P = \{x = k\}$, $Q = \{x = k!\}$, $\sigma = \{i = 1, x = 2, k = 2\}$

Unfolding the definition,
$\{i = 1, x = 2, k = 2\} \vDash \{x = k\}$ Yes
$\langle S, \{i = 1, x = 2, k = 2\}\rangle \rightarrow^* \langle skip, \{i = 2, x = 2, k = 2\}\rangle$, then Yes
$\{i = 2, x = 2, k = 2\} \vDash \{x = k!\}$ Yes

The state σ satisfy the Hoare triple.

e) $\{i = 1, x = 6\} \vDash \{T\}\ s\ \{\exists k.x = k!\}$

Ans. The definition of the partial correctness says,
When σ is the start state, we define $\sigma \vDash \{P\}S\{Q\}$ as

if σ ⊨ P, and
⟨S, σ⟩ →* ⟨skip, σ'⟩ , then
σ' ⊨ Q

Given: P = {T}, Q = {∃k.x = k!}, σ = {i = 1, x = 6}

Unfolding the definition,
{i = 1, x = 6} ⊨ {T} Yes
⟨S, {i = 1, x = 6}⟩ →* ⟨skip, σ'⟩ , then No because the loop in statement S is an infinite loop.
σ' ⊭ Q

The state σ satisfy the Hoare triple.

Task 1.2

Consider program s in the previous task. For each of the following triples, say if it's valid (satisfied in all states or not). If not, provide a counterexample and then fix either the precondition, or the postcondition or the statement to make the triple valid. Don't make your change trivial (that is, don't make the precondition a contradiction, the postcondition a tautology or the statement something that always errors or diverges).

a) {T } s {x > 0}

Ans. The definition of the partial correctness says,
When σ is the start state, we define σ ⊨ {P}S{Q} as
if σ ⊨ P, and
⟨S, σ⟩ →* ⟨skip, σ'⟩ , then
σ' ⊨ Q

Given: P = {T}, Q = { x > 0}

We want to check if this triple is valid or not,
Unfolding the definition,
σ ⊨ {T} All states, Yes
⟨S, σ⟩ →* ⟨skip, σ'⟩ , then All states, No

The triple is not valid due to states where (i=positive and x=negative), (i=negative and x=negative), and (i=negative and x=positive).

To make the triple valid, we will change our precondition, i.e., we will make it stronger.

⊨**{x > 0 ∧ i > 0} s {x > 0}** **Valid Hoare Triple**

b) {x = k} s {x = k}

Ans. The definition of the partial correctness says,
When σ is the start state, we define σ ⊨ {P}S{Q} as
if σ ⊨ P, and
⟨S, σ⟩ $\rightarrow^*$ ⟨skip, σ′⟩ , then
σ′ ⊨ Q

Given: P = {x=k}, Q = { x = k}

We want to check if this triple is valid or not,
Unfolding the definition, it does not hold in all the states.

Hence, the triple is not valid due to states where i=negative value.

To make the triple valid, we will change our precondition, i.e., we will make it stronger.

⊨**{x = k ∧ i > 0} s {x = k}** **Valid Hoare Triple**

c) [i = 1 ∧ x = k ∧ x > 0] s [i = k ∧ x = k!]

Ans. The definition of the total correctness says,
When σ is the start state, we define σ ⊨ [P]S[Q] as
if σ ⊨ P, then
⟨S, σ⟩ $\rightarrow^*$ ⟨skip, σ′⟩ , and
σ′ ⊨ Q

Given: P = [i = 1 ∧ x = k ∧ x > 0], Q = [i = k ∧ x = k!]

Unfolding the definition, it does not hold in all the states.

Hence, the triple is not valid due to states as the value of x changes, and it is used in the conditional statement of while loop.

To make the triple valid, we will change our statement,

⊨**[i = 1 ∧ x = k ∧ x > 0] s [i = k ∧ x = k!]** **Valid Hoare Triple**

where, s = while i<k do x := x * i; i := i + $\overline{1}$ od

This performs the factorial of number x.

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW3

Task 1.3

Fill in an appropriate precondition such that the following triple is valid. Don't provide a trivial precondition (that is, don't make the precondition a contradiction).

$$[\_\_]\ n := -m;\ while\ n \neq 0\ do\ r := r * -3;\ n := n - 1\ od\ [r = 3^{-m}]$$

Ans. I tried multiple values of m, r and finally understood that because we are taking n = -m, our m must be negative value. So, m < 0.
Also, whenever m was even and negative (e.g.: -2, -4, -6, etc.), value of r must be 1 so that the triple holds for total correctness. Along with this, whenever m was odd and negative (e.g.: -1, -3, -5, etc.), value of r must be -1 so that the triple holds for total correctness.

Precondition for the above triple:
$m<0 \wedge ((m\%2=0 \wedge r=1) \vee (m\%2\neq0 \wedge r=-1))$

Given triple becomes,

$$\vDash [\mathrm{m} < 0 \wedge ((\mathrm{m}\%2 = 0 \wedge \mathrm{r} = 1) \vee (\mathrm{m}\%2 \neq 0 \wedge \mathrm{r} = -1))]\ n := -m;\ while\ n \neq 0\ do\ r := r * -3;\ n := n - 1\ od\ [r = 3^{-m}]$$

Task 1.4

Write a precondition P1 such that the triple ⊨ [P1] x := sqrt(x)/y [T] is valid. Don't make your precondition trivial (a contradiction). Explain by unfolding the definition of the triple as we saw in the class.

Ans. The definition of the total correctness says,
When σ is the start state, we define σ ⊨ [P]S[Q] as
if σ ⊨ P, then
⟨S, σ⟩ $\rightarrow^*$ ⟨skip, σ'⟩ , and
σ' ⊨ Q

Given: P = [P1], Q = [T], S = x := sqrt(x)/y

In order for this triple to be valid, the program S should terminate without any errors. The two possible situations that I can think of where this triple will fail due to the statement is division by zero error and sqrt of a negative number.
So, if we have a check in the precondition saying that x has to be a perfect square and the value of y cannot be zero, the triple should hold as per my understanding.

Hence, P1 should be [$x = k^2 \wedge y \neq 0$]

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW3

The triple will look like

$\models$[x = k^2 $\wedge$ y $\neq$ 0] s [T] **Valid Hoare Triple**

Task 1.5

a) Write a program S2 and a precondition P2 such that $\not\models$[P2]S2[T]. Explain by unfolding the definition of the triple as we saw in the class.

Ans. The definition of the total correctness says,
When σ is the start state, we define σ ⊨ [P]S[Q] as
if σ ⊨ P, then
⟨S, σ⟩ $\rightarrow^*$ ⟨skip, σ'⟩ , and
σ' ⊨ Q

Given: P = [P2], Q = [T], S = S2

In order for this triple to be invalid, the program S should either terminate or throw any error(s).
We can think of some statement which throws an error.

Say, P2 can be [x > 0 ∧ y > -50] and S2 can be x := (x/y)
The triple will look like

$\not\models$[x > 0 ∧ y > -50] x := (x/y) [T] **Hoare Triple**

b) Have you used a while clause in S2? If yes, can you provide another S2 this time without using the while clause? If no, can you provide another S2 this time with using the while clause?

Ans. No, I did not use a while clause in S2.

Here's another example with while clause.

The definition of the total correctness says,
When σ is the start state, we define σ ⊨ [P]S[Q] as
if σ ⊨ P, then
⟨S, σ⟩ $\rightarrow^*$ ⟨skip, σ'⟩ , and
σ' ⊨ Q

Given: P = [P2], Q = [T], S = S2

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW3

In order for this triple to be invalid, the program S should either terminate or throw any error(s).
We can think of a while statement which never terminates.

Say, P2 can be [x > 0] and S2 can be while x>0 do x := x + $\bar{1}$ od
The triple will look like

$\not\models$[x > 0] while x>0 do x := x + $\bar{1}$ od [T] **Hoare Triple**

Task 1.6

a) Write a triple for the following program that is valid if and only if the program terminates when the initial value of x is greater than three.
while x > 1 do if even(x) then x := 5x + 1 else x := x/2 fi od

Ans. As mentioned by the professor over Discord and confirmed by the TA, we just have to write a triple and do not think whether the triple will ever be valid or not.

Given wording: initial value of x is greater than three.

This suggests that precondition should be {x > 3}.

S = while x > 1 do if even(x) then x := 5x + 1 else x := x/2 fi od

Only left is the post condition.

Now, as the wording says that the program will terminate if we start with an initial value of x as greater than three. Our statement has a while loop with the conditional statement x>1. This statement will terminate when x $\leq$ 1.

This can be a potential postcondition which we can say about x.

Hence, as per my understanding, a triple for the given program that is valid if and only if the program terminates when the initial value of x is greater than three can be:

{x>3} while x > 1 do if even(x) then x := 5x + 1 else x := x/2 fi od {x $\leq$ 1}

b) Write another triple for the above program which is valid if and only if the program does not terminate when the initial value of x is less than or equal to three.

Ans. As mentioned by the professor over Discord and confirmed by the TA, we just have to write a triple and do not think whether the triple will ever be valid or not.

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW3

Given wording: initial value of x is less than or equal to three.

This suggests that precondition should be $\{x \leq 3\}$.

S = while x > 1 do if even(x) then x := 5x + 1 else x := x/2 fi od

Only left is the post condition.

Now, as the wording says that the program does not terminate if we start with an initial value of x as less than or equal to three. Our statement has a while loop with the conditional statement x>1. This statement will never terminate if value of x always stays greater than 1 which suggests the condition x > 1.

This can be a potential postcondition which we can say about x.

Hence, as per my understanding, a triple for the above program which is valid if and only if the program does not terminate when the initial value of x is less than or equal to three can be:

$\{x \leq 3\}$ while x > 1 do if even(x) then x := 5x + 1 else x := x/2 fi od $\{x > 1\}$

2. Substitution

Task 2.1

Compute the given substitutions. Just substitute the expression for the value; you don't need to simplify anything further. Recall that you may need to perform α-conversions to avoid variable capture.

a) $[y + 2/y]\exists z.\forall x.(x + y \geq z + y)$

Ans. The variable y is not bounded by the quantifiers. Hence, we can substitute the value of y = y+2

$\exists z.\forall x.(x + y + 2 \geq z + y + 2)$

b) $[y + 2/x]\exists z.\forall x.(x + y \geq z + y)$

Ans. The variable x is bounded by the quantifiers. Hence, we cannot directly substitute the value of x = y+2. We will use the concept of ghost variable.

$[y + 2/x_0]\exists z.\forall x_1.(x_1 + y \geq z + y)$
This becomes,

$\exists z.\forall x_1.(x_1 + y \geq z + y)$

c) $[x + 2/y]\exists z.\forall x.(x + y \geq z + y)$

Ans. The variable y is not bounded by the quantifiers. However, there is x in the substitution expression of y as well as there is a variable x which is bounded by the quantifier. Hence, we cannot directly substitute the value of y = x+2. We will use the concept of ghost variable.

$[x_0 + 2/y]\exists z.\forall x_1.(x_1 + y \geq z + y)$

This becomes,

$\exists z.\forall x_1.(x_1 + x_0 + 2 \geq z + x_0 + 2)$

d) $[z/x](x \geq z \rightarrow (\exists z.\forall x.x + y \geq z + y) \wedge y > z)$

Ans. The variable x is bounded by the quantifiers. Also, there is z in the substitution expression of x as well as there is a variable z which is not bounded by the quantifier. Hence, we cannot directly substitute the value of x = z. We will use the concept of ghost variable.

$[z/x](x \geq z \rightarrow (\exists z.\forall x.x + y \geq z + y) \wedge y > z)$

Can be rewritten as:
$[z_0/x_0](x_0 \geq z_1 \rightarrow (\exists z_2.\forall x_1.x_1 + y \geq z_2 + y) \wedge y > z_1)$

This becomes,
$(z_0 \geq z_1 \rightarrow (\exists z_2.\forall x_1.x_1 + y \geq z_2 + y) \wedge y > z_1)$

e) $[z/x](x \geq z \rightarrow (\exists x.x + y \geq z + y) \wedge y > z)$

Ans. The variable x is bounded by the quantifiers. Also, there is z in the substitution expression of x. Hence, we cannot directly substitute the value of x = z. We will use the concept of ghost variable.

$[z/x](x \geq z \rightarrow (\exists x.x + y \geq z + y) \wedge y > z)$

Can be rewritten as:
$[z_0/x_0](x_0 \geq z_1 \rightarrow (\exists x_1.x_1 + y \geq z_1 + y) \wedge y > z_1)$

This becomes,

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW3

$(z_0 \geq z_1 \rightarrow (\exists x_1.x_1 + y \geq z_1 + y) \wedge y > z_1)$

3. Proofs and Proof Outlines

Task 3.1

a) Write a program in IMP that given an array a of size two, returns its maximum element in variable m.

Ans. Given: array a of size two. After the program execution, variable m must hold the maximum value of the two elements in array a.

IMP language program for the given task,

```
if a[0] ≥ a[1] then
   m := a[0]
else
   m := a[1]
fi
```

b) Write a partial correctness triple to state that your program does what is described in part (a).

Ans. From the above IMP program,

We can think of {P}S{Q} as

$P = \{m = 0\}$
$S \triangleq$ if $a[0] \geq a[1]$ then $m := a[0]$ else $m := a[1]$ fi
$Q = \{m = \max(a[0],a[1])\}$

The definition of the partial correctness says,
When σ is the start state, we define $\sigma \models \{P\}S\{Q\}$ as
if $\sigma \models P$, and
$\langle S, \sigma \rangle \rightarrow^* \langle skip, \sigma' \rangle$, then
$\sigma' \models Q$

Given:
$P = \{m = 0\}$, $Q = \{m = \max(a[0],a[1])\}$, $S =$ if $a[0] \geq a[1]$ then $m := a[0]$ else $m := a[1]$ fi

c) Prove your partial correctness triple in Hoare logic using either proof trees or Hilbert-style proofs.

Bhavesh Rajesh Talreja (A20516822)
CS 536 HW3

Ans. Proving the total correctness triple by Hilbert style proof:

1. {a[0] = max(a[0], a[1])} m := a[0] {m = max(a[0], a[1])} Assign
2. {a[1] = max(a[0], a[1])} m := a[1] {m = max(a[0], a[1])} Assign
3. {m = 0 ∧ (a[0] ≥ a[1])} m := a[0] {m = max(a[0], a[1])} Consequence (1)
4. {m = 0 ∧ (a[0] < a[1])} m := a[1] {m = max(a[0], a[1])} Consequence (2)
5. {m = 0} if a[0] ≥ a[1] then m := a[0] else m := a[1] fi {m = max(a[0], a[1])} if(3,4)

d) Write a proof outline for your partial correctness triple. You can use either rule for if that we discussed in class.

Ans. Proving the same triple using a proof outline:

{m=0}	
if a[0] ≥ a[1] then	{m = 0 ∧ (a[0] ≥ a[1])}
m := a[0]	{m = a[0]}
else	{m = 0 ∧ (a[0] < a[1])}
m := a[1]	{m = a[1]}
fi	{m = a[0] ∨ m=a[1]}

Task 3.2

Convert the following proof outline to a Hilbert-style proof. The remainder operator x%y returns the remainder when x is divided by y.

{x ≥ 0}	
if(x%3 = 0) then	{x ≥ 0 ∧ x%3 = 0}
s := x	{s ≥ 0 ∧ s%3 = 0} ⇒ {s%3 = 0}
else	{x ≥ 0 ∧ x%3 ≠ 0}
if(x%3 = 1) then	{x ≥ 0 ∧ x%3 ≠ 0 ∧ x%3 = 1} ⇒ {x − 1 ≥ 0 ∧ x − 1%3 = 0}
s := x – 1	{s ≥ 0 ∧ s%3 = 0} ⇒ {s%3 = 0}
else	{x ≥ 0 ∧ x%3 ≠ 0 ∧ x%3 ≠ 1} ⇒ {x − 2 ≥ 0 ∧ x − 2%3 = 0}
s := x – 2	{s ≥ 0 ∧ s%3 = 0} ⇒ {s%3 = 0}
fi	{s%3 = 0}
fi	{s%3 = 0}

Ans. By referring the notes and examples solved in class, I tried to convert the proof outline to Hilbert style proof.

1. {x%3 = 0} s := x {s%3 = 0} Assign
2. {x ≥ 0 ∧ x%3 = 0} s := x {s%3 = 0} Consequence (1)
3. {x - 1%3 = 0} s := x – 1 {s%3 = 0} Assign
4. {x - 1 ≥ 0 ∧ x - 1%3 = 0} s := x – 1 {s%3 = 0} Consequence (3)

5. $\{x - 2\%3 = 0\}$ s := x – 2 $\{s\%3 = 0\}$ Assign
6. $\{x - 2 \geq 0 \wedge x - 2\%3 = 0\}$ s := x – 2 $\{s\%3 = 0\}$ Consequence (5)

   Now that we have separate parts, we want to bring all these statements together using the if rule. However, we face an issue where Q for both statement 4 and statement 6 is same, but precondition should be of the form $(P \wedge e)$ and $(P \wedge \neg e)$ to combine. But we cannot see that here. So, we will apply consequence rule on statement 4 and 6.

7. $\{(x \geq 0 \wedge x\%3 \neq 0) \wedge x\%3 = 1\}$ s := x – 1 $\{s\%3 = 0\}$ Consequence (4)
8. $\{(x \geq 0 \wedge x\%3 \neq 0) \wedge x\%3 \neq 1\}$ s := x – 2 $\{s\%3 = 0\}$ Consequence (6)
9. $\{x \geq 0 \wedge x\%3 \neq 0\}$ if x%3 = 1 then s := x-1 else s := x-2 fi $\{s\%3 = 0\}$ if (7,8)
10. $\{x \geq 0\}$ if x%3 = 0 then s := x else if x%3 = 1 then s := x-1 else s := x-2 fi fi $\{s\%3 = 0\}$ if (2,9)

4. One more wrap-up question

   Task 4.1

   How long (approximately) did you spend on this homework, in total hours of actual working time?

   Ans. Totally I spent 16 hours on this assignment.