CS 536 - Science of Programming
Assignment - 3

1)

a) Given,

$S \equiv$ if $x > y \rightarrow x := x-1 \;\square\; x > y \rightarrow y := y+1 \;\square\; x+y=4 \rightarrow x := y | x \;\square\; x+y=4 \rightarrow x := x|y$ fi.

$\sigma = \{x=3, y=1\}$

we need to calculate $M(S, \sigma)$

$M(S,\sigma) = ($if $x > y \rightarrow x := x-1 \;\square\; x > y \rightarrow y := y+1 \;\square\; x+y=4 \rightarrow x := y|x \;\square\; x+y=4 \rightarrow x := x|y$ fi, $\{x=3, y=1\})$

So, for the given state $\sigma = \{x=3, y=1\}$ each and every guard passes.

Hence, we get

$M(S,\sigma) = \{\{x=2, y=1\}, \{x=3, y=2\}, \{x=1/3, y=1\}, \{x=3, y=1\}\}$

b) Given,

$W \equiv$ do $x > y \rightarrow x := x-1 \;\square\; x > y \rightarrow y := y+1 \;\square\; x+y=4 \rightarrow x := y|x \;\square\; x+y=4 \rightarrow x := x|y$
od

$\sigma = \{x = 3, y = 1\}$

we need to calculate $M(W, \sigma)$

$M(W, \sigma) = (\text{do } x > y \to x := x - 1 \,\square\, x > y \to y := y + 1 \,\square\, x + y = 4 \to x := y/x \,\square\, x + y = 4 \to x := x/y \text{ od}, \{x = 3, y = 1\})$

After first iteration, we get the states

$\{\{x = 2, y = 1\}, \{x = 3, y = 2\}, \{x = 1/3, y = 1\},$

$\{x = 3, y = 1\}\}$

After the second iteration, the states are updated to.

$\{\{x = 1, y = 1\}, \{x = 2, y = 2\}, \{x = 3, y = 3\}\}$

In this second iteration, every state except $\{x = 1/3, y = 1\}$ passes

Here, we can observe that $\{x = 3, y = 1\}$ creates infinite loop so, it diverges $(\perp_d)$

Third iteration,

Here only $\{x = 2, y = 2\}$ passes the guards

so, the updated new states are

$\{x = 1, y = 2\}$.

So,

$M(W, \sigma) = \{\{x=1, y=1\}, \{x=1, y=2\}, \{x=3, y=3\}, \{x=1/3, y=1\}, \perp_d\}$

2) Given that,

b is an array of size $n \geq 1$ and $\forall 0 \leq i < n. b[i] = 0 \vee b[i] = 1$

Let us consider K_0 and K_1 as pointers for 0 & 1

Now, we need to find (0 or 1) which is majority in b without counting their quantities.

$$
\begin{aligned}
MAJORITY \equiv\ & K_0 := 0;\ K_1 := 0; \\
& \text{while } K_0 < n \wedge K_1 < n \text{ do } J; \\
& K_0 := K_0 + 1; \\
& K_1 := K_1 + 1; \\
& \text{od};
\end{aligned}
$$

In the above program, J is:

$$J \equiv \text{do } b[K_0] = 1 \rightarrow K_0 := K_0 + 1 \ \square\ b[K_1] = 0 \rightarrow K_1 := K_1 + 1 \text{ od}$$

3)

a) True, because deterministic statements will always lead to one unique state for a given input σ

b) False, because it is also possible that $\sigma \nvDash P$.

c) False, because there can also be $\sigma \vDash P$ but $M(S, \sigma) \nvDash q$

d) False, because even if $\sigma \nvDash P$ it does not affect the validity of $\sigma \vDash \{P\} S \{q\}$. It can still be valid under certain conditions.

e) True, because if partial correctness does not hold then total correctness cannot hold either.

4)

a) valid, because if $P(K, s+1)$ is true, then $(s+1)^2 \leq K \leq (s+2)^2$. Once after incrementing s, we will have $s^2 \leq K \leq (s+1)^2$ which basically satisfies $P(K, s)$

b) Not valid, because if $P(K, s)$ is true, then $s^2 \leq K \leq (s+1)^2$. After incrementing s, we will need $(s+1)^2 \leq K \leq (s+2)^2$ which is not guaranteed.

c) Not valid, If $P(k,s) \wedge s < 0$ after both s and k being incremented we have to check if the new values satisfy the predicate. The post-condition constraint might not hold if k was initially $(s+1)^2$ because $k+1$ would be too large for the post-condition.

d) Not valid, The precondition does not affect the post condition since it refers to a fixed value s_0. So, changing s does not affect the truth of the predicate with respect to fixed initial value.

e) Valid, If precondition holds for incremented values of both variables then after executing the statement the original predicate should hold for non-incremented values.

5)

a) The possible values for $\sigma(x)$ are:

* For negative numbers, it will be $\perp_d$.
* For odd numbers, it will be $\perp_d$
* For $\sigma(x)=0$, pre-condition will be false
* $\sigma(x)$ can't be even numbers because the program will execute and it will be terminated without satisfying q.

b) The possible value for $\sigma(x)$ for this is only 0.

6)
a) This is valid under total correctness because $(P_1 \wedge P_2)$ makes sure a strong precondition which leads to at least one of the post conditions $(q_1 \vee q_2)$ being true.

b) This is not necessarily valid for total correctness because having only one of the preconditions $(P_1 \vee P_2)$ does not guarantee both post conditions $(q_1 \wedge q_2)$ will be true.

c) This is valid under total correctness as the disjunction in the pre-condition leads to a disjunction in the post-condition which is a weaker requirement and thus more easily satisfied.

7)
a) valid, if both P_1 and P_2 hold then after executing S both q_1 and q_2 must hold satisfying $q_1 \wedge q_2$.

b) valid, because executing S with precondition P_2 makes sure q_2 it also ensures $q_1 \rightarrow q_2$ because if q_1 is true then q_2 must be true.

c) Not valid, The precondition $\neg P_1 \rightarrow P_2$ does not necessarily ensure that either $\neg q_1$ or q_2 will hold after executing S.

8)

a) True, Since w is the weakest precondition $\models_{tot} \{w\} S \{q\}$,

b) True, because $wp(S,q) \Rightarrow \models_{tot} \{w\} S \{q\}$ which also implies $\models \{w \wedge q\} S \{q\}$

c) False, There might exist a state σ such that $\sigma \not\models w$ and $\sigma \models \{w\} S \{q\}$ but it is not guaranteed.

d) True, This follows from $\models_{tot} \{w\} S \{q\}$ $\sigma \not\models w$, $M(S,\sigma) = \{\top\}$

e) False, for any state $\sigma \not\models w$, $M(S,\sigma)$ must be either a pseudo-state or a state satisfying $\neg q$. Thus, we can have $\sigma \models_{tot} \{\neg w\}$ $S\{\neg q\}$ but not necessarily $\models_{tot} \{\neg w\} S \{\neg q\}$

9) Given,

$S \equiv y := y \% x$ and $q \equiv sqrt(y) > x$

a) $wlp(S,q) \equiv wlp(y := y\%x,\ sqrt(y) > x)$

$$\equiv (sqrt(y \% x) > x)$$

b) $wp(S,q) \equiv$

$$D(S) \equiv D(y := y \% x) \equiv x \neq 0$$

$$D(wlp(S,q)) \equiv D(sqrt(y \% x) > x) \equiv y \% x \geq 0 \wedge x \neq 0$$

so,

$$wp(S,q) \equiv (x \neq 0) \wedge (sqrt(y \% x) > x) \wedge (y \% x \geq 0 \wedge x \neq 0)$$

$\Leftrightarrow x \neq 0 \wedge sqrt(y \% x) > x \wedge y \% x \geq 0$

10) $S \equiv$ if $y \geq 0 \rightarrow x := y/x \;\square\; x \geq 0 \rightarrow x := x/y$ fi
and $q \equiv x < y < 2$

a) $wlp(S, q)$

$wlp(S, q) \equiv wlp($if $y \geq 0 \rightarrow x := y/x \;\square\; x \geq 0 \rightarrow x := x/y$ fi$, x < y < 2)$

$\equiv (y \geq 0 \rightarrow wlp(x := y/x, x < y < 2))$
~~$\wedge$ (y≥0 → wlp~~
$\wedge (x \geq 0 \rightarrow wlp(x := x/y; x < y < 2))$

$\equiv (y \geq 0 \rightarrow x < y/x < 2) \wedge (x \geq 0 \rightarrow x < x/y < 2)$

b) $D(S) \equiv T \wedge (y \geq 0 \rightarrow x \neq 0) \wedge (x \geq 0 \rightarrow y \neq 0) \Leftrightarrow$
$y \geq 0 \rightarrow x \neq 0$

$D(wlp(S, q)) \equiv x \neq 0 \wedge y \neq 0$.

so, Here.

$wp(S, q) \equiv (y \geq 0 \rightarrow x \neq 0) \wedge (y \geq 0 \rightarrow x < y/x < 2)$
$\wedge (x \geq 0 \rightarrow x < x/y < 2) \wedge$.
$(x \neq 0 \wedge y \neq 0)$.