CS412 Solution sheet 5

Nondeterministic wp and machine consistency

1. Calculate and simplify (assuming a type-correct context):

   (a) $[xx :\in \{ii \mid ii \in \mathbb{N} \wedge ii < 5\}]xx < 2$
   $= \forall\, ss \bullet (ss : \{ii \mid ii \in \mathbb{N} \wedge ii < 5\} \Rightarrow (xx < 2)[ss/xx])$
   $= \forall\, ss \bullet (ss : \{ii \mid ii \in \mathbb{N} \wedge ii < 5\} \Rightarrow ss < 2)$
   $= false$

   (b) $\left[\begin{array}{l} ANY\ xx \\ WHERE\ xx : MID \wedge xx : S \\ THEN\ members := members \cup \{xx \mapsto 62\} \\ END \end{array}\right] members \in MID \nrightarrow \mathbb{N}$
   $= \forall\, xx \bullet ((xx : MID \wedge xx : S) \Rightarrow$
   $[members := members \cup \{xx \mapsto 62\}]members \in MID \nrightarrow \mathbb{N})$
   $= \forall\, xx \bullet ((xx : MID \wedge xx : S) \Rightarrow members \cup \{xx \mapsto 62\} \in MID \nrightarrow \mathbb{N})$
   Note that if we knew that $members \in MID \nrightarrow \mathbb{N}$ initially (which seems likely) then this would be equivalent to $\mathrm{dom}(members) \cap S = \{\}$.

   (c) $\left[\begin{array}{l} ANY\ xx \\ WHERE\ xx : \mathbb{N} \wedge xx * xx < 10 \\ THEN\ yy := yy + xx \\ END \end{array}\right] yy < 8$
   $= \forall\, xx \bullet ((xx : \mathbb{N} \wedge xx * xx < 10) \Rightarrow [yy := yy + xx]yy < 8)$
   $= \forall\, xx \bullet (xx : \{0, 1, 2, 3\} \Rightarrow yy + xx < 8)$
   Which is true exactly when $yy < 5$.

   (d) $[CHOICE\ mm := xx\ OR\ mm := yy\ END]mm = xx$
   $= [mm := xx]mm = xx \wedge [mm := yy]mm = xx$
   $= xx = xx \wedge yy = xx$
   $= yy = xx$

2. Show that $wp(P, Q) \vee wp(P, R)$ is *NOT* always equal to $wp(P, Q \vee R)$.
   Hint: think of a simple nondeterministic operation such as a coin toss.

   Eg: suppose $COIN = \{heads, tails\}$ and P is $xx :\in COIN$.

   $wp(P, xx = heads \vee xx = tails)$
   $= wp(xx :\in COIN, true)$
   $= \forall\, ss \bullet (ss : COIN \Rightarrow true)$
   $= true$

   but

   $wp(P, xx = heads) \vee wp(P, xx = tails)$
   $= wp(xx :\in COIN, xx = heads) \vee wp(xx :\in COIN, xx = tails)$

$= \forall\, ss \bullet (ss : COIN \Rightarrow ss = heads) \lor \forall\, ss \bullet (ss : COIN \Rightarrow ss = tails)$
$= false \lor false$
$= false$

What this tells us is that we might know that a nondeterministic operation ends up in one of a possible number of states - but we can't say which one exactly. This is unlike deterministic operations for which the two statements above *ARE* equal.

3. (a) $[\textbf{IF } xx \neq 0 \textbf{ THEN } oo := val \ div \ xx \textbf{ END}]R$
$= (xx \neq 0 \land [R]((val \ div \ xx)/oo)) \lor (x = 0 \land R)$

   (b) $[\textbf{PRE } xx \neq 0 \textbf{ THEN } oo := val \ div \ xx \textbf{ END}]R$
$= xx \neq 0 \land [R]((val \ div \ xx)/oo)$

The IF has an implicit "do nothing" branch which allows the operation to simply skip if the condition is false.
The PRE only guarantees to work correctly if the condition is true. Outside this there is no guarantee of any sort of behaviour. Anything could happen! The operation could act as it likes - doesn't even have to terminate.
If $xx = 0$ initially, the IF is guaranteed to establish R if R is true initially. The PRE would never be guaranteed to establish R.
An implementation of IF is bound to do nothing when the condition is false.
An implementation of PRE can choose whatever course of action the implementor wants.

4. (a) The condition for an operation is: $Inv \land Pre \Rightarrow [Body]Inv$.
In this case we get:
$$inside \subseteq PID \ \land \ maxin \in \mathbb{N}_1 \ \land \ card(inside) \leq maxin \ \land \ nn \in \mathbb{N}_1 \Rightarrow$$
$$inside \subseteq PID \ \land \ nn \in \mathbb{N}_1 \ \land \ card(inside) \leq nn$$
The first and second conjunct on the RHS of the implication follow immediately from the LHS. However, the third conjunct is not derivable from any of the information on the LHS: we are unable to prove that $card(inside) \leq nn$.

   (b) It's clear that the operation doesn't check that there are not already too many people in the building. We can fix the operation by adding the offending case to the precondition. That is:
$$change\_lim(nn) \ \widehat{=} \ \textbf{PRE } nn : \mathbb{N}_1 \land card(inside) \leq nn \textbf{ THEN } maxin := nn \textbf{ END}$$

   (c) **MACHINE** *entrysys*
SETS PID; $NAME$; $CATEGORY$
VARIABLES $inside, maxin, reg$
INVARIANT $inside \subseteq PID \land maxin \in \mathbb{N}_1 \land inside \subseteq \text{dom}(reg) \land card(inside) \leq maxin \land reg \in PID \nrightarrow (NAME \times CATEGORY)$
INITIALISATION $inside := \{\} \ || \ maxin := 500 \ || \ reg := \{\}$
OPERATIONS

$ww \leftarrow whosin \;\widehat{=}\; ww := inside;$

$enter(pp) \;\widehat{=}\; \mathbf{PRE}\; pp : PID \wedge pp \notin inside \;\wedge$
$pp \in \mathrm{dom}\, reg \wedge card(inside) < maxin$
$\mathbf{THEN}\; inside := inside \cup \{pp\}$
$\mathbf{END};$

$nn \leftarrow catin(cc) \;\widehat{=}\; \mathbf{PRE}\; cc : CATEGORY$
$\mathbf{THEN}\; nn := \mathrm{dom}((reg(\!| inside |\!)) \rhd \{cc\})$
$\mathbf{END}$

END

The definition of *catin* could also be achieved using a set description, eg:

$nn := \{xx \mid xx \in NAME \wedge$
$\exists\, pp \bullet (pp \in PID \wedge pp \in inside \wedge (pp \mapsto (xx \mapsto cc)) \in reg)\}$

A query operation such as this does not change the state. So, if the invariant is true to start with it will remain so.