MODULE *tracker*

Packages

EXTENDS *Integers*, *TLC*

Constants

CONSTANTS

N,
CID, set of all possible container identifiers
PID set of all possible phase identifiers

ASSUME

$N > 0$

$MATERIAL \triangleq \{\text{“glass”}, \text{“metal”}, \text{“plastic”}, \text{“liquid”}\}$ the only materials each container is expected to have
$ERROR \triangleq \{\text{“ok”}, \text{“E1”}, \text{“E2”}, \text{“E3”}, \text{“E4”}, \text{“E5”}, \text{“E6”}\}$ one error for each action

Note:
*E*1: maps to errors from *new_tracker* action (*e*1, *e*2, *e*3, *e*4)
*E*2: maps to errors from *new_phase* action (*e*1, *e*5, *e*6, *e*7, *e*8)
*E*3: maps to errors from *new_container* action (*e*5, *e*9, *e*10, *e*11, *e*12, *e*13, *e*14, *e*18)
*E*4: maps to errors from *remove_container* action (*e*15)
*E*5: maps to errors from *remove_phase* action (*e*1, *e*9)
*E*6: maps to errors from *move_contaier* action (*e*9, *e*11, *e*12, *e*13, *e*15, *e*16, *e*17)

set of all phase records
$PHASE \triangleq [capacity : Int, curr\_cont : Int, curr\_rad : Int, exp\_mats : \text{SUBSET } MATERIAL]$
$CONTAINER \triangleq [radioactivity : Int, curr\_phase : PID, mat : MATERIAL]$ set of all container records
$VALUE \triangleq 0 \,..\, N$ used for defining numbers for *radioactivity*. Abstracting away from decimals

Variables

VARIABLES

cid, set of container ids in the tracker
containers, set of functions mapping from *CID* to *CONTAINER*
pid, set of phase ids in the tracker
phases, set of functions mapping from *PID* to *PHASE*
mpr, the maximum radiation allowed in a phase
mcr, the maximum radiation allowed in a container
e error message holder

Helper Functions used for remvoving element from domain and range from function
Subtraction on domain of f for set S: This was obtained from Course Forum User *amin*9 $**$
$DomSub(S, f) \triangleq [x \in \text{DOMAIN } f \setminus S \mapsto f[x]]$
Subtraction on range of f for set S: This was obtained from Course Forum User *amin*9 $**$
$RanSub(f, S) \triangleq [x \in \{y \in \text{DOMAIN } f : f[y] \notin S\} \mapsto f[x]]$

**** INVARIANTS *****

type invariant

$$\begin{aligned} \mathit{TypeOK} \triangleq\ & \land e \in \mathit{ERROR} \\ & \land \mathit{pid} \subseteq \mathit{PID} \\ & \land \mathit{cid} \subseteq \mathit{CID} \\ & \land \mathit{phases} \in [\mathit{pid} \to \mathit{PHASE}] \\ & \land \mathit{containers} \in [\mathit{cid} \to \mathit{CONTAINER}] \\ & \land \mathit{mpr} \in \mathit{Int} \\ & \land \mathit{mcr} \in \mathit{Int} \end{aligned}$$

checking *radioactivity* is within limits of tracker

$$\begin{aligned} \mathit{SafeRadioactivity} \triangleq\ & \land \forall\, \mathit{id} \in \mathit{pid} :\ \land 0 \leq \mathit{phases}[\mathit{id}].\mathit{curr\_rad} \\ & \qquad\qquad\qquad\qquad\ \land \mathit{phases}[\mathit{id}].\mathit{curr\_rad} \leq \mathit{mpr} \\ & \land \forall\, \mathit{id} \in \mathit{cid} :\ \land 0 \leq \mathit{containers}[\mathit{id}].\mathit{radioactivity} \\ & \qquad\qquad\qquad\qquad\ \land \mathit{containers}[\mathit{id}].\mathit{radioactivity} \leq \mathit{mcr} \end{aligned}$$

checking that all phases are within capacity in terms of the number containers in them

$$\begin{aligned} \mathit{PhasesNotOverCapacity} \triangleq\ & \forall\, \mathit{id} \in \mathit{pid} :\ \land 0 \leq \mathit{phases}[\mathit{id}].\mathit{curr\_cont} \\ & \qquad\qquad\qquad\qquad\qquad\ \land \mathit{phases}[\mathit{id}].\mathit{curr\_cont} \leq \mathit{phases}[\mathit{id}].\mathit{capacity} \end{aligned}$$

All containers are in a phase that exists in the tracker

$$\mathit{AllContainersInPhase} \triangleq \forall\, \mathit{id} \in \mathit{cid} : \mathit{containers}[\mathit{id}].\mathit{curr\_phase} \in \mathit{pid}$$

**** ACTIONS *****

$$\begin{aligned} & \mathit{new\_tracker}(p,\ c) \triangleq \\ & \quad \land p \geq 0 \quad \text{max phase radiation is non-negative} \\ & \quad \land c \geq 0 \quad \text{max container radiation is non-negative} \\ & \quad \land p \geq c \quad \text{max container radiation cannot be greater than max phase radiation} \\ & \quad \land \mathit{cid} = \{\} \quad \text{there should be no containers in tracker} \\ & \quad \text{\_\_\_\_}\mathit{guard}\ \text{ends}\ \mathit{here}\text{\_\_\_\_} \\ & \quad \land \mathit{cid}' = \mathit{cid} \\ & \quad \land \mathit{containers}' = \mathit{containers} \\ & \quad \land \mathit{pid}' = \mathit{pid} \\ & \quad \land \mathit{phases}' = \mathit{phases} \\ & \quad \land \mathit{mpr}' = p \quad \text{set the values for maximum phase and container radiation} \\ & \quad \land \mathit{mcr}' = c \\ & \quad \land e' = \text{“ok”} \\ & \quad \land \mathit{Print}(\langle \text{“new\_tracker”},\ p,\ c \rangle,\ \text{TRUE}) \end{aligned}$$

$new\_phase(p,\ cap,\ mats) \triangleq$
$\land\ p \notin pid$ phase *id* must not already exist
$\land\ cap > 0$ capacity must be positive
$\land\ mats \subseteq MATERIAL$ check that expected materials is subset of all possible *MATERIALs*
$\land\ cid = \{\}$ there should be no containers in tracker
____guard ends *here____*
$\land\ cid' = cid$
$\land\ containers' = containers$
$\land\ pid' = pid \cup \{p\}$ add new *id* to set of phase ids and extend phase function with a new record
$\land\ phases' = phases\ @@\ p :> [capacity \mapsto cap,\ curr\_cont \mapsto 0,\ curr\_rad \mapsto 0,\ exp\_mats \mapsto mats]$
$\land\ mpr' = mpr$
$\land\ mcr' = mcr$
$\land\ e' =$ "ok"
$\land\ Print(\langle$"new_phase", $p,\ cap\rangle,$ TRUE$)$

$new\_container(c,\ rad,\ p,\ m) \triangleq$
$\land\ c \notin cid$ container *id* must not already exist
$\land\ p \in pid$ make sure the phase exists
$\land\ rad \geq 0 \land rad \leq mcr$ radiation must be within limits of tracker
$\land\ phases[p].curr\_rad + rad \leq mpr$ make sure the phase radiation won't exceed
$\land\ phases[p].curr\_cont + 1 \leq phases[p].capacity$ and capacity won't exceed
$\land\ m \in MATERIAL \land m \in phases[p].exp\_mats$ check phase accepts this material
____guard ends *here____*
$\land\ cid' = cid \cup \{c\}$ add new *id* to set of ids and extend container function with new record
$\land\ containers' = containers\ @@\ c :> [radioactivity \mapsto rad,\ curr\_phase \mapsto p,\ mat \mapsto m]$
$\land\ pid' = pid$ update the phase by adding new container to it. That is update the radiation and count
$\land\ phases' = [phases$ EXCEPT $![p].curr\_rad = @ + rad,\ ![p].curr\_cont = @ + 1]$
$\land\ mpr' = mpr$
$\land\ mcr' = mcr$
$\land\ e' =$ "ok"
$\land\ Print(\langle$"new_container", $c,\ rad,\ p\rangle,$ TRUE$)$

$remove\_container(c) \triangleq$
$\land\ c \in cid$ this container must exist
____guard ends *here____*
$\land\ pid' = pid$ update phase by removing container from it
$\land\ phases' = [phases$ EXCEPT
$![containers[c].curr\_phase].curr\_rad = @ - containers[c].radioactivity,$
$![containers[c].curr\_phase].curr\_cont = @ - 1]$
$\land\ cid' = cid \setminus \{c\}$ remove *id* from set of container ids and also remove the *id* from the
$\land\ containers\ = RanSub(containers,\ \{c\})$ domain and the record from the range
$\land\ containers' = DomSub(\{c\},\ containers)$ of containers function
$\land\ mpr' = mpr$
$\land\ mcr' = mcr$
$\land\ e' =$ "ok" $\land\ Print(\langle$"remove_container", $c\rangle,$ TRUE$)$

$remove\_phase(p) \triangleq$
$\land\ p \in pid$ this phase must exist
$\land\ cid = \{\}$ cannot remove phase when there are containers currently in tracker
____guard ends *here____*
$\land\ cid' = cid$
$\land\ containers' = containers$
$\land\ pid' = pid \setminus \{p\}$ remove *id* from set of phase ids
$\land\ phases\ = RanSub(phases, \{p\})$ remove *id* from domain and record from range
$\land\ phases' = DomSub(\{p\}, phases)$ of phases function
$\land\ mpr' = mpr$
$\land\ mcr' = mcr$
$\land\ e' =$ "ok"
$\land\ Print(\langle$"remove_phase", $p\rangle$, TRUE$)$

$move\_container(c, p1, p2) \triangleq$
$\land\ c \in cid$ check container exists
$\land\ p1 \in pid \land p2 \in pid$ check phases exist
$\land\ p1 \neq p2$ source and destination phases must be different
$\land\ containers[c].curr\_phase = p1$ container should be in source phase
$\land\ phases[p2].curr\_rad + containers[c].radioactivity \leq mpr$ safe radiation
$\land\ phases[p2].curr\_cont + 1 \leq phases[p2].capacity$ and capacity on destination should not exceed
$\land\ containers[c].mat \in phases[p2].exp\_mats$ check destination phase accepts container material
____guard ends *here____*
$\land\ cid' = cid$ update record of container with the new phase it is in
$\land\ containers' = [containers$ EXCEPT $![c].curr\_phase = p2]$
$\land\ pid' = pid$ update phase records by removing container from $p1$ and adding it to $p2$
$\land\ phases' = [phases$ EXCEPT
$\quad ![p1].curr\_rad = @ - containers[c].radioactivity,$
$\quad ![p1].curr\_cont = @ - 1,$
$\quad ![p2].curr\_rad = @ + containers[c].radioactivity,$
$\quad ![p2].curr\_cont = @ + 1]$
$\land\ mpr' = mpr$
$\land\ mcr' = mcr$
$\land\ e' =$ "ok"
$\land\ Print(\langle$"move_container", $c, p1, p2\rangle$, TRUE$)$

******** Defining *INIT* and NEXT ********

$$
\begin{aligned}
\mathit{Init} \triangleq\ & \land\ e = \text{“ok”} \quad \text{initialize error } \mathit{ok} \\
& \land\ \mathit{mpr} = 0 \quad \text{max phase radiation is 0} \\
& \land\ \mathit{mcr} = 0 \quad \text{max container radiation is 0} \\
& \land\ \mathit{cid} = \{\} \quad \text{empty set of container ids} \\
& \land\ \mathit{containers} = \langle\rangle \quad \text{make empty set of function from } \mathit{cid} \to \mathit{CONTAINER} \\
& \land\ \mathit{pid} = \{\} \quad \text{make emtpy set} \\
& \land\ \mathit{phases} = \langle\rangle \quad \text{make empty set of function from } \mathit{pid} \to \mathit{PHASE}
\end{aligned}
$$

$$
\begin{aligned}
\mathit{Next} \triangleq\ & \lor\ (\exists\, p,\ c \in \mathit{VALUE} : \mathit{new\_tracker}(p,\ c)) \\
& \lor\ (\exists\, p \in \mathit{PID},\ \mathit{cap} \in \mathit{VALUE},\ \mathit{mats} \in \text{SUBSET } \mathit{MATERIAL} : \mathit{new\_phase}(p,\ \mathit{cap},\ \mathit{mats})) \\
& \lor\ (\exists\, c \in \mathit{CID},\ \mathit{rad} \in \mathit{VALUE},\ p \in \mathit{PID},\ \mathit{mat} \in \mathit{MATERIAL} : \mathit{new\_container}(c,\ \mathit{rad},\ p,\ \mathit{mat})) \\
& \lor\ (\exists\, c \in \mathit{CID} : \mathit{remove\_container}(c)) \\
& \lor\ (\exists\, p \in \mathit{PID} : \mathit{remove\_phase}(p)) \\
& \lor\ (\exists\, c \in \mathit{CID},\ p1,\ p2 \in \mathit{PID} : \mathit{move\_container}(c,\ p1,\ p2))
\end{aligned}
$$