```
MODULE callback
EXTENDS TLC, Sequences, SequencesExt, FiniteSets, Integers

CONSTANTS Timers, DeltaRange

   --algorithm callback
variables
     list for timer
     example: $\langle[delta \mapsto 3, name \mapsto \text{“}timer1\text{”}], [delta \mapsto 2, name \mapsto \text{“}timer2\text{”}]\rangle$
    $delta\_list = SetToSeq(\{[delta \mapsto random\_num(0, DeltaRange), name \mapsto x] : x \in Timers\})$ ;

     tasks
    $running = \{\}$ ;

define
    $random\_num(min, max) \triangleq$ CHOOSE $i \in min \,..\, max$ : TRUE
    $starvation\_free \triangleq \forall x \in Timers : (x \in \{y.name : y \in ToSet(delta\_list)\} \leadsto \Diamond(x \in running))$
end define

 To emulate incrementing clock, decrement the delta of the head of the delta_list.
macro increment_clock()
begin
    if $delta\_list \neq \langle\rangle \land delta\_list[1].delta > 0$ then
        $delta\_list[1].delta := delta\_list[1].delta - 1$ ;
    end if ;
end macro ;

 callback
procedure timer_callback(name)
begin
    start_callback:
        increment_clock() ;
        $running := running \cup \{name\}$ ;

    end_callback:
        $running := running \setminus \{name\}$ ;
        return ;
end procedure ;

 reenable timer with at random delay
procedure reload_timer(name)
variables
    idx ;
    delta ;
begin
    start_reload_timer:
        increment_clock() ;
        choose insertion point
        $idx := random\_num(1,\ Len(delta\_list) + 1)$ ;
        if $idx \leq Len(delta\_list)$ then
            insert to middle
            $delta := random\_num(0,\ delta\_list[idx].delta)$ ;

            $reload\_insert1$:
                update delta and insert
                $delta\_list[idx].delta := delta\_list[idx].delta - delta$ ;

            $reload\_insert2$:
                $delta\_list := InsertAt(delta\_list,\ idx,\ [delta \mapsto delta,\ name \mapsto name])$ ;
        else
            insert to the end
            $delta := random\_num(0,\ DeltaRange)$ ;

            $reload\_insert\_end$:
                $delta\_list := Append(delta\_list,\ [delta \mapsto delta,\ name \mapsto name])$ ;
            skip ;
        end if ;

    $end\_reload\_timer$:
        return ;
end procedure ;

fair + process $executor$ = "executor"
variables
    $head$ ;
    $to\_be\_reloaded = \langle\rangle$ ;
begin
    $start\_executor$:
        while TRUE do
            $increment\_clock()$ ;

            $execute$:
                while $delta\_list \neq \langle\rangle \land delta\_list[1].delta = 0$ do
                    pop front
                    $head := Head(delta\_list)$ ;
                    $delta\_list := Tail(delta\_list)$ ;

                    call the callback function
                    call $timer\_callback(head.name)$ ;

                    reenable timer later
                    $save\_timer$:
                        $to\_be\_reloaded := Append(to\_be\_reloaded,\ head.name)$ ;
                end while ;
            $reload$:
               reenable timer
               while $to\_be\_reloaded \neq \langle\rangle$ do
                  call $reload\_timer(to\_be\_reloaded[1])$ ;

                  $reload2$:
                     $to\_be\_reloaded := Tail(to\_be\_reloaded)$ ;
               end while ;
         end while ;
end process ;
end algorithm ;
 BEGIN TRANSLATION ($chksum(pcal) =$ "$528a467b$" $\land$ $chksum(tla) =$ "$dfa7a4e5$")
 Parameter name of procedure $timer\_callback$ at line 29 col 26 changed to $name\_$
CONSTANT $defaultInitValue$
VARIABLES $delta\_list, running, pc, stack$

 define statement
$random\_num(min, max) \triangleq$ CHOOSE $i \in min \, .. \, max$ : TRUE
$starvation\_free \triangleq \forall x \in Timers : (x \in \{y.name : y \in ToSet(delta\_list)\} \leadsto \Diamond(x \in running))$

VARIABLES $name\_, name, idx, delta, head, to\_be\_reloaded$

$vars \triangleq \langle delta\_list, running, pc, stack, name\_, name, idx, delta, head,$
          $to\_be\_reloaded\rangle$

$ProcSet \triangleq \{$"executor"$\}$

$Init \triangleq$  Global variables
        $\land\ delta\_list = SetToSeq(\{[delta \mapsto random\_num(0, DeltaRange), name \mapsto x] : x \in Timers\})$
        $\land\ running = \{\}$
         Procedure $timer\_callback$
        $\land\ name\_ = [self \in ProcSet \mapsto defaultInitValue]$
         Procedure $reload\_timer$
        $\land\ name = [self \in ProcSet \mapsto defaultInitValue]$
        $\land\ idx = [self \in ProcSet \mapsto defaultInitValue]$
        $\land\ delta = [self \in ProcSet \mapsto defaultInitValue]$
         Process executor
        $\land\ head = defaultInitValue$
        $\land\ to\_be\_reloaded = \langle\rangle$
        $\land\ stack = [self \in ProcSet \mapsto \langle\rangle]$
        $\land\ pc = [self \in ProcSet \mapsto$ "start_executor"$]$

$start\_callback(self) \triangleq$ $\land\ pc[self] =$ "start_callback"
                          $\land$ IF $delta\_list \neq \langle\rangle \land delta\_list[1].delta > 0$
                                THEN $\land\ delta\_list' = [delta\_list$ EXCEPT $![1].delta = delta\_list[1].delta - 1]$
                                ELSE $\land$ TRUE
                                     $\land$ UNCHANGED $delta\_list$
                          $\land running' = (running \cup \{name\_[self]\})$
                          $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“end\_callback”}]$
                          $\land \text{UNCHANGED } \langle stack, name\_, name, idx, delta, head,$
                                              $to\_be\_reloaded\rangle$

$end\_callback(self) \triangleq$ $\land pc[self] = \text{“end\_callback”}$
                          $\land running' = running \setminus \{name\_[self]\}$
                          $\land pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
                          $\land name\_' = [name\_ \text{ EXCEPT } ![self] = Head(stack[self]).name\_]$
                          $\land stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
                          $\land \text{UNCHANGED } \langle delta\_list, name, idx, delta, head,$
                                              $to\_be\_reloaded\rangle$

$timer\_callback(self) \triangleq start\_callback(self) \lor end\_callback(self)$

$start\_reload\_timer(self) \triangleq$ $\land pc[self] = \text{“start\_reload\_timer”}$
                          $\land \text{IF } delta\_list \neq \langle\rangle \land delta\_list[1].delta > 0$
                                $\text{THEN } \land delta\_list' = [delta\_list \text{ EXCEPT } ![1].delta = delta\_list[1].delta$
                                $\text{ELSE } \land \text{TRUE}$
                                       $\land \text{UNCHANGED } delta\_list$
                          $\land idx' = [idx \text{ EXCEPT } ![self] = random\_num(1, Len(delta\_list') + 1)]$
                          $\land \text{IF } idx'[self] \leq Len(delta\_list')$
                                $\text{THEN } \land delta' = [delta \text{ EXCEPT } ![self] = random\_num(0, delta\_list'[id$
                                       $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“reload\_insert1”}]$
                                $\text{ELSE } \land delta' = [delta \text{ EXCEPT } ![self] = random\_num(0, DeltaRange)]$
                                       $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“reload\_insert\_end”}]$
                          $\land \text{UNCHANGED } \langle running, stack, name\_, name, head,$
                                              $to\_be\_reloaded\rangle$

$reload\_insert1(self) \triangleq$ $\land pc[self] = \text{“reload\_insert1”}$
                          $\land delta\_list' = [delta\_list \text{ EXCEPT } ![idx[self]].delta = delta\_list[idx[self]].delta -$
                          $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“reload\_insert2”}]$
                          $\land \text{UNCHANGED } \langle running, stack, name\_, name, idx,$
                                              $delta, head, to\_be\_reloaded\rangle$

$reload\_insert2(self) \triangleq$ $\land pc[self] = \text{“reload\_insert2”}$
                          $\land delta\_list' = InsertAt(delta\_list, idx[self], [delta \mapsto delta[self], name \mapsto name[$
                          $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“end\_reload\_timer”}]$
                          $\land \text{UNCHANGED } \langle running, stack, name\_, name, idx,$
                                              $delta, head, to\_be\_reloaded\rangle$

$reload\_insert\_end(self) \triangleq$ $\land pc[self] = \text{“reload\_insert\_end”}$
                          $\land delta\_list' = Append(delta\_list, [delta \mapsto delta[self], name \mapsto name[self]])$
                          $\land \text{TRUE}$
                          $\land pc' = [pc \text{ EXCEPT } ![self] = \text{“end\_reload\_timer”}]$
                          $\land \text{UNCHANGED } \langle running, stack, name\_, name, idx,$
                                              $delta, head, to\_be\_reloaded\rangle$
```

```
$end\_reload\_timer(self) \triangleq \land pc[self] = \text{"end\_reload\_timer"}$
    $\land pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc]$
    $\land idx' = [idx \text{ EXCEPT } ![self] = Head(stack[self]).idx]$
    $\land delta' = [delta \text{ EXCEPT } ![self] = Head(stack[self]).delta]$
    $\land name' = [name \text{ EXCEPT } ![self] = Head(stack[self]).name]$
    $\land stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])]$
    $\land \text{UNCHANGED } \langle delta\_list, running, name\_, head,$
            $to\_be\_reloaded \rangle$

$reload\_timer(self) \triangleq start\_reload\_timer(self) \lor reload\_insert1(self)$
    $\lor reload\_insert2(self) \lor reload\_insert\_end(self)$
    $\lor end\_reload\_timer(self)$

$start\_executor \triangleq \land pc[\text{"executor"}] = \text{"start\_executor"}$
    $\land \text{IF } delta\_list \neq \langle\rangle \land delta\_list[1].delta > 0$
        $\text{THEN } \land delta\_list' = [delta\_list \text{ EXCEPT } ![1].delta = delta\_list[1].delta - 1]$
        $\text{ELSE } \land \text{TRUE}$
              $\land \text{UNCHANGED } delta\_list$
    $\land pc' = [pc \text{ EXCEPT } ![\text{"executor"}] = \text{"execute"}]$
    $\land \text{UNCHANGED } \langle running, stack, name\_, name, idx, delta,$
            $head, to\_be\_reloaded \rangle$

$execute \triangleq \land pc[\text{"executor"}] = \text{"execute"}$
    $\land \text{IF } delta\_list \neq \langle\rangle \land delta\_list[1].delta = 0$
        $\text{THEN } \land head' = Head(delta\_list)$
              $\land delta\_list' = Tail(delta\_list)$
              $\land \land name\_' = [name\_ \text{ EXCEPT } ![\text{"executor"}] = head'.name]$
                $\land stack' = [stack \text{ EXCEPT } ![\text{"executor"}] = \langle [procedure \mapsto \text{"timer\_callback"},$
                                                     $pc \mapsto \text{"save\_timer"},$
                                                     $name\_ \mapsto name\_[\text{"executor"}]]$
                                                     $\circ\, stack[\text{"executor"}]]$
              $\land pc' = [pc \text{ EXCEPT } ![\text{"executor"}] = \text{"start\_callback"}]$
        $\text{ELSE } \land pc' = [pc \text{ EXCEPT } ![\text{"executor"}] = \text{"reload"}]$
              $\land \text{UNCHANGED } \langle delta\_list, stack, name\_, head \rangle$
    $\land \text{UNCHANGED } \langle running, name, idx, delta, to\_be\_reloaded \rangle$

$save\_timer \triangleq \land pc[\text{"executor"}] = \text{"save\_timer"}$
    $\land to\_be\_reloaded' = Append(to\_be\_reloaded, head.name)$
    $\land pc' = [pc \text{ EXCEPT } ![\text{"executor"}] = \text{"execute"}]$
    $\land \text{UNCHANGED } \langle delta\_list, running, stack, name\_, name, idx,$
            $delta, head \rangle$

$reload \triangleq \land pc[\text{"executor"}] = \text{"reload"}$
    $\land \text{IF } to\_be\_reloaded \neq \langle\rangle$
        $\text{THEN } \land \land name' = [name \text{ EXCEPT } ![\text{"executor"}] = to\_be\_reloaded[1]]$
                $\land stack' = [stack \text{ EXCEPT } ![\text{"executor"}] = \langle [procedure \mapsto \text{"reload\_timer"},$
```

```
                                                                                                    $pc \mapsto$ "reload2",
                                                                                                    $idx \mapsto idx[$"executor"$]$,
                                                                                                    $delta \mapsto delta[$"executor"$]$,
                                                                                                    $name \mapsto name[$"executor"$]]\rangle$
                                                                                                    $\circ\ stack[$"executor"$]]$
                                          $\land\ idx' = [idx$ EXCEPT $![$"executor"$] = defaultInitValue]$
                                          $\land\ delta' = [delta$ EXCEPT $![$"executor"$] = defaultInitValue]$
                                          $\land\ pc' = [pc$ EXCEPT $![$"executor"$] =$ "start_reload_timer"$]$
                                    ELSE  $\land\ pc' = [pc$ EXCEPT $![$"executor"$] =$ "start_executor"$]$
                                          $\land$ UNCHANGED $\langle stack, name, idx, delta\rangle$
                          $\land$ UNCHANGED $\langle delta\_list, running, name\_, head, to\_be\_reloaded\rangle$

$reload2 \triangleq$ $\land\ pc[$"executor"$] =$ "reload2"
           $\land\ to\_be\_reloaded' = Tail(to\_be\_reloaded)$
           $\land\ pc' = [pc$ EXCEPT $![$"executor"$] =$ "reload"$]$
           $\land$ UNCHANGED $\langle delta\_list, running, stack, name\_, name, idx, delta,$
                        $head\rangle$

$executor \triangleq start\_executor \lor execute \lor save\_timer \lor reload \lor reload2$

$Next \triangleq executor$
           $\lor\ (\exists\ self \in ProcSet : timer\_callback(self) \lor reload\_timer(self))$

$Spec \triangleq \land\ Init \land \Box[Next]_{vars}$
         $\land\ \land\ \mathrm{SF}_{vars}(executor)$
           $\land\ \mathrm{SF}_{vars}(timer\_callback($"executor"$))$
           $\land\ \mathrm{SF}_{vars}(reload\_timer($"executor"$))$

 END TRANSLATION
```