```
global
    $Addr\ head;\ Addr\ tail;$
    $\boxed{Set\langle Addr\rangle\ reg;\ Set\langle Elem\rangle\ elems;}$
assume
    $reg = \{head, tail, null\}$
$\wedge\ elems = \{head \rightarrow data, tail \rightarrow data\}$
$\wedge\ head \neq tail \wedge head \neq null \wedge tail \neq null$
$\wedge\ head \rightarrow data = -\infty \wedge tail \rightarrow data = +\infty$
$\wedge\ head \rightarrow next = tail \wedge tail \rightarrow next = null$
```

```
procedure MGC
    $Elem\ e$
begin
while true do
    $e :=$ havocListElem()
    nondet choice
        call Search($e$)
    or
        call Insert($e$)
    or
        call Remove($e$)
end while
end procedure
```

```
procedure Insert($e$)
    $Addr\quad prev$
    $Addr\quad curr$
    $Addr\quad aux$
begin
$prev := head$
$prev \rightarrow lock()$
$curr := prev \rightarrow next$
$curr \rightarrow lock()$
while $curr \rightarrow data < e$ do
    $aux := prev$
    $prev := curr$
    $aux \rightarrow unlock()$
    $curr := curr \rightarrow next$
    $curr \rightarrow lock()$
end while
if $curr \neq null \wedge curr \rightarrow data > e$ then
    $aux := malloc(e, null, \#)$
    $aux \rightarrow next := curr$
    $prev \rightarrow next := aux$
    $\boxed{\begin{array}{l} reg := reg \cup \{aux\} \\ elems := elems \cup \{e\} \end{array}}$
end if
$prev \rightarrow unlock()$
$curr \rightarrow unlock()$
return
end procedure
```

```
procedure Search($e$)
    $Addr\quad prev$
    $Addr\quad curr$
    $Addr\quad aux$
    $Bool\quad found$
begin
$prev := head$
$prev \rightarrow lock()$
$curr := prev \rightarrow next$
$curr \rightarrow lock$
while $curr \rightarrow data < e$ do
    $aux := prev$
    $prev := curr$
    $aux \rightarrow unlock()$
    $curr := curr \rightarrow next$
    $curr \rightarrow lock()$
end while
$found := (curr \rightarrow data = e)$
$prev \rightarrow unlock()$
$curr \rightarrow unlock()$
return $found$
end procedure
```

```
procedure Remove($e$)
    $Addr\quad prev$
    $Addr\quad curr$
    $Addr\quad aux$
begin
$prev := head$
$prev \rightarrow lock()$
$curr := prev \rightarrow next$
$curr \rightarrow lock()$
while $curr \rightarrow data < e$ do
    $aux := prev$
    $prev := curr$
    $aux \rightarrow unlock()$
    $curr := curr \rightarrow next$
    $curr \rightarrow lock()$
end while
if $curr \neq tail \wedge curr \rightarrow data = e$ then
    $aux := curr \rightarrow next$
    $prev \rightarrow next := aux$
    $\boxed{\begin{array}{l} reg := reg \setminus \{curr\} \\ elems := elems \setminus \{e\} \end{array}}$
end if
$prev \rightarrow unlock()$
$curr \rightarrow unlock()$
return
end procedure
```