X4.0 (one-tailed if) We have a two-tailed **if** programming notation with the syntax

if binary expression **then** program **else** program **fi**

Most programming languages also have a one-tailed **if** with a syntax like

if binary expression **then** program **fi**

It is executed by first evaluating the binary expression, and then executing the program if and only if the binary expression's value was $\top$.

(a) Define the one-tailed **if** formally.

(b) Let n be a natural variable, and let t be time measured recursively. Define

$$countdown \;=\; n{\geq}0 \Rightarrow t'{=}t{+}n$$

Prove the refinement

$$countdown \;\Leftarrow\; \textbf{if } n{>}0 \textbf{ then } n{:=}\, n{-}1.\;\; t{:=}\, t{+}1.\;\; countdown \textbf{ fi}$$

After trying the question, scroll down to the solution.

(a) Define the one-tailed **if** formally.

§ A one-tailed **if** is equal to a two-tailed **if** with an *ok* **else**-part.

$$\textbf{if } b \textbf{ then } P \textbf{ fi} \;=\; \textbf{if } b \textbf{ then } P \textbf{ else } ok \textbf{ fi}$$
$$\textbf{if } b \textbf{ then } P \textbf{ fi} \;=\; b \wedge P \vee \neg b \wedge ok$$
$$\textbf{if } b \textbf{ then } P \textbf{ fi} \;=\; (b \Rightarrow P) \wedge (\neg b \Rightarrow ok)$$

(b) Let n be a natural variable, and let t be time measured recursively. Define

$$countdown \;=\; n \geq 0 \Rightarrow t' = t + n$$

Prove the refinement

$$countdown \;\Leftarrow\; \textbf{if } n > 0 \textbf{ then } n := n - 1.\ t := t + 1.\ countdown \textbf{ fi}$$

§ Rewriting the refinement with a two-tailed **if** :

$$countdown \;\Leftarrow\; \textbf{if } n > 0 \textbf{ then } n := n - 1.\ t := t + 1.\ countdown \textbf{ else } ok \textbf{ fi}$$

There are two cases to prove. First case:

	$n>0 \wedge (n := n-1.\ t := t+1.\ countdown) \Rightarrow countdown$	definition of *countdown*
$=$	$n>0 \wedge (n := n-1.\ t := t+1.\ n \geq 0 \Rightarrow t' = t+n) \Rightarrow countdown$	substitution law twice
$=$	$n>0 \wedge (n-1 \geq 0 \Rightarrow t' = t+1+n-1) \Rightarrow countdown$	simplify
$=$	$n>0 \wedge (n>0 \Rightarrow t' = t+n) \Rightarrow countdown$	discharge
$=$	$n>0 \wedge t' = t+n \Rightarrow countdown$	definition of *countdown*
$=$	$n>0 \wedge t' = t+n \Rightarrow (n \geq 0 \Rightarrow t' = t+n)$	portation
$=$	$n>0 \wedge t' = t+n \wedge n \geq 0 \Rightarrow t' = t+n$	specialization
$=$	$\top$	

Second case:

	$n=0 \wedge ok \Rightarrow countdown$	definitions of *ok* and *countdown*
$=$	$n=0 \wedge n' = n \wedge t' = t \Rightarrow (n \geq 0 \Rightarrow t' = t+n)$	portation
$=$	$n=0 \wedge n' = n \wedge t' = t \wedge n \geq 0 \Rightarrow t' = t+n$	simplify $n=0 \wedge n \geq 0$
$=$	$n=0 \wedge n' = n \wedge t' = t \Rightarrow t' = t+n$	context
$=$	$n=0 \wedge t' = t \Rightarrow t' = t$	specialization
$=$	$\top$	