203 (merge) Given two sorted lists, write a program to merge them into one sorted list.

After trying the question, scroll down to the solution.

§ Given two sorted lists L and M, we can define their merge $L\otimes M$. Let's give $\otimes$ precedence level 6. The most obvious definition is that $L\otimes M$ is the sorted permutation of $L;;M$. Another way is by the axioms

(a) $[nil] \otimes L \ = \ L \ = \ L \otimes [nil]$

(b) $x \le y \ \Rightarrow\ [x];;L \otimes [y];;M \ = \ [x] ;; (L \otimes [y];;M)$

(c) $x \ge y \ \Rightarrow\ [x];;L \otimes [y];;M \ = \ [y] ;; ([x];;L \otimes M)$

Axiom (a) defines merge when either list is empty, and axioms (b) and (c) define merge when both lists are nonempty. (The notation would be less cluttered with brackets if we were merging strings.) From these axioms we can prove

$L\otimes M \ = \ M\otimes L$

$L\otimes(M\otimes N) \ = \ (L\otimes M)\otimes N$

But we don't need them. Now for the program. Let the given lists be A and B (these are constants, not variables). Introduce variable C to accumulate the result. Introduce natural variables a and b as indexes in A and B. Then

$C' = A\otimes B \ \Longleftarrow\ a:= 0.\ \ b:= 0.\ \ C:= [nil].\ \ P$

where $P \ = \ C' \ = \ C;;(A[a;..\#A] \otimes B[b;..\#B])$. The proof of this first refinement is just 3 uses of the Substitution Law and $[nil]$ is the identity for $;;$. The other refinement is

```
P  ⇐  if $a=\#A \vee b=\#B$ then $C:= C;;A[a;..\#A];;B[b;..\#B]$
      else if $A\ a \le B\ b$ then $C:= C;;[A\ a].\ \ a:= a+1.\ \ P$
           else $C:= C;;[B\ b].\ \ b:= b+1.\ \ P$ fi fi
```

The first case to be proven is

$P \ \Leftarrow\ (a=\#A \vee b=\#B) \wedge (C:= C;;A[a;..\#A];;B[b;..\#B])$

use distributivity and antidistributivity

$=\quad (P \ \Leftarrow\ a=\#A \wedge (C:= C;;A[a;..\#A];;B[b;..\#B]))$ expand P

$\wedge\ \ (P \ \Leftarrow\ b=\#B \wedge (C:= C;;A[a;..\#A];;B[b;..\#B]))$ expand P

$=\quad (C' = C;;(A[a;..\#A] \otimes B[b;..\#B]) \ \Leftarrow\ a=\#A \wedge (C:= C;;A[a;..\#A];;B[b;..\#B]))$

$\wedge\ \ (C' = C;;(A[a;..\#A] \otimes B[b;..\#B]) \ \Leftarrow\ b=\#B \wedge (C:= C;;A[a;..\#A];;B[b;..\#B]))$

use context $a=\#A$ in first line and $b=\#B$ in second line

$=\quad (C' = C;;([nil] \otimes B[b;..\#B]) \ \Leftarrow\ a=\#A \wedge (C:= C;;[nil];;B[b;..\#B]))$

$\wedge\ \ (C' = C;;(A[a;..\#A] \otimes [nil]) \ \Leftarrow\ b=\#B \wedge (C:= C;;A[a;..\#A];;[nil]))$

$[nil]$ is identity for both $;;$ and $\otimes$.

$=\quad (C' = C ;; B[b;..\#B] \ \Leftarrow\ a=\#A \wedge (C:= C ;; B[b;..\#B]))$ specialization

$\wedge\ \ (C' = C ;; A[a;..\#A] \ \Leftarrow\ b=\#B \wedge (C:= C ;; A[a;..\#A]))$ specialization

$=\quad \top$

The middle case is

$a<\#A \wedge b<\#B \wedge A\ a \le B\ b \wedge (C:= C;;[A\ a].\ \ a:= a+1.\ \ P)$ replace P

$=\quad a<\#A \wedge b<\#B \wedge A\ a \le B\ b$

$\wedge\ (C:= C;;[A\ a].\ \ a:= a+1.\ \ C' = C;;(A[a;..\#A] \otimes B[b;..\#B]))$

Substitution

$=\quad a<\#A \wedge b<\#B \wedge A\ a \le B\ b \wedge (C' = C;;[A\ a];;(A[a+1;..\#A] \otimes B[b;..\#B]))$

now use merge axiom (b) in right-to-left direction

$=\quad a<\#A \wedge b<\#B \wedge A\ a \le B\ b \wedge (C' = C;;([A\ a];;A[a+1;..\#A] \otimes B[b;..\#B]))$ simpify

$=\quad a<\#A \wedge b<\#B \wedge A\ a \le B\ b \wedge (C' = C;;(A[a;..\#A] \otimes B[b;..\#B]))$ specialize

$\Longrightarrow\quad P$

The last case is just like the middle case.