Write a program to flatten a list. The result is a new list just like the old one but without the internal structure. For example,

\[ L = \{ [3; 5]; 2; [5; [7]; \textit{nil}] \} \]
\[ L' = [3; 5; 2; 5; 7] \]

Your program may employ a test \( Li: \text{int} \) to see if an item is an integer or a list.

§

Define \( \text{flat} \ L \) to mean that list \( L \) is flat.

\[ \text{flat} \ L = \forall i: 0,..\#L: L[i]: \text{int} \]

or more succinctly

\[ \text{flat} \ L = L: [*\text{int}] \]

Define \( \text{sim} \ L \ M \) to mean that lists \( L \) and \( M \) have the same items in the same order, though they may have different internal structure. Formally, it is easier to define \( \text{sim} \) for all strings. Let \( s \), \( t \), and \( u \) be strings, and let \( i \) and \( j \) be integers.

\[ \text{sim} \ s \ t = \text{sim} \ t \ s \]
\[ \text{sim} \ s \ t \land \text{sim} \ t \ u \Rightarrow \text{sim} \ s \ u \]
\[ \text{sim} \ s \ \text{nil} = s=\text{nil} \]
\[ \text{sim} \ (i; s) \ (j; t) = i=j \land \text{sim} \ s \ t \]
\[ \text{sim} \ (s; [t]; u) \ (s; t; u) \]

Define specifications \( P \) and \( Q \) as

\[ P = \text{flat} \ L' \land \text{sim} \ L \ L' \]
\[ Q = L'[0;..k]=L[0;..k] \land \text{flat} \ (L'[k;..\#L']) \land \text{sim} \ (L[k;..\#L]) \land (L'[k;..\#L']) \]

Then the refinements are

\[ P \iff k:=0. \ Q \]
\[ Q \iff \text{if } k=\#L \text{ then ok} \]
\[ \quad \text{else if } L \text{ k: \text{int} then } k:=k+1. \ Q \]
\[ \quad \text{else } L:=L[0;..k]; \ L \text{ k };; \ L[k+1;..\#L]. \ Q \text{ fi fi} \]