(squash) Let $L$ be a list variable assigned a non-empty list. Reassign it so that any run of two or more identical items is collapsed to a single item.

After trying the question, scroll down to the solution.
§ Let \( i \) be a natural variable used to index \( L \). The program is
\[
P \iff i := 1 . \ Q
Q \iff \text{if } i = \#L \text{ then } \text{ok}
\text{else if } L \ i = L(i-1) \text{ then } L := L((0;..i) ; (i+1;..\#L))
\text{else } i := i+1 \ fi \ Q \ fi
\]

Now we need to define specifications \( P \) and \( Q \), and then prove the two refinements. Part of the specification says that \( L \) and \( L' \) have the same items in them.
\[ L(0,..\#L) = L'(0,..\#L') \]
Another part of the specification is that in \( L' \) no two adjacent items are equal.
\[ \neg \exists j: 1,..\#L' \cdot L' j = L'(j-1) \]
The rest of the specification is that the items of \( L' \) are in the same order as in \( L \). I don't know how to formalize that. So I'll prove what I can.

Let's start with
\[ P = Q = L(0,..\#L) = L'(0,..\#L') \]
Here's the proof of the first refinement, starting with the right side.
\[
i := 1 . \ Q \iff i := 1 . \ L(0,..\#L) = L'(0,..\#L') \iff L(0,..\#L) = L'(0,..\#L') \iff P
\]
Now the last refinement, by cases. First case:
\[
i = \#L \land \text{ok} \iff i = \#L \land L' = L \land i' = i \implies Q
\]
Middle case:
\[
i + \#L \land L \ i = L(i-1) \land (L := L((0;..i) ; (i+1;..\#L))) \iff Q \implies Q
\]
Last case:
\[
i + \#L \land L \ i = L(i-1) \land L := L((0,..\#L) ; (i+1;..\#L)) \iff Q \implies Q
\]

Now redefine
\[ P = \neg \exists j: 1,..\#L' \cdot L' j = L'(j-1) \]
\[ Q = \neg \exists j: i,..\#L' \cdot L' j = L'(j-1) \]
Here's the proof of the first refinement, starting with the right side.
\[
i := 1 . \ Q \iff i := 1 \cdot \neg \exists j: i,..\#L' \cdot L' j = L'(j-1) \iff \neg \exists j: 1,..\#L' \cdot L' j = L'(j-1) \iff P
\]
Now the last refinement, by cases. First case:
\[
i = \#L \land \text{ok} \iff i = \#L \land L' = L \land i' = i \implies Q \iff \text{null domain}
\]
Middle case:
\[
i + \#L \land L \ i = L(i-1) \land (L := L((0;..i) ; (i+1;..\#L))) \iff Q \implies Q
\]
Last case:
\[ i \neq \#L \land L \neq L(i-1) \land (i := i + 1. \ Q) \]
\[ \Rightarrow \ UNFINISHED \]

The recursive time is \#L − 1. Redefine

\[ P \Leftarrow t' = t + \#L - 1 \]
\[ Q \Leftarrow t' = t + \#L - i \]

and insert the time increment

\[ P \Leftarrow i := 1. \ Q \]
\[ Q \Leftarrow \text{if } i = \#L \text{ then ok} \]
\[ \quad \text{else if } L \neq L(i-1) \text{ then } L := L((0, i) ; (i + 1, \#L)) \]
\[ \quad \text{else } i := i + 1 \text{ fi} \quad t := t + 1. \ Q \text{ fi} \]

UNFINISHED