Write a program to find, in a given list of naturals, the number of segments whose sum is a given natural.

§

Let \( L \) be the given list, and \( n \) be the given natural. The first problem is to say formally “the number of segments in \( L \) whose sum is \( n \)”. Instead of “segments”, we can say “the number of naturals \( a \) and \( b \) such that \( 0 \leq a \leq b \leq \#L \wedge \Sigma \{ a;..b \} = n \)”. The quantifier \( \exists \) turns a predicate into a bunch, and then \( \ast \) tells the size of the bunch, but unfortunately \( \exists \) works on only one variable, not two. Still, we can sum up the sizes. Formally,

\[
\Sigma a, b \cdot 0 \leq a \leq b \leq \#L \wedge (\Sigma L [a;..b]) = n
\]

But that's ugly. To get a neater, more workable expression, add axioms \( T=1 \) and \( \bot=0 \) equating binary values and numbers. Now the number of segments is

\[
\Sigma a, b \cdot 0 \leq a \leq b \leq \#L \wedge (\Sigma L [a;..b]) = n
\]

Now the refinements are

I will need two more similar specifications \( A \) and \( B \), defined as

\[
\begin{align*}
A &= c' = c + \Sigma b \cdot 0 \leq i \leq b \leq \#L \wedge (\Sigma L [i;..b]) = n \\
B &= i'=i \wedge c' = c + \Sigma b \cdot 0 \leq j \leq b \leq \#L \wedge s + (\Sigma L [j;..b]) = n
\end{align*}
\]

Now the refinements are

\[
R \equiv \begin{cases} i:=0, c:=0. A & \text{if } i=\#L \text{ then } ok \\ i:=i+1. A \text{ fi} & \text{else}
\end{cases}
\]

The second sum looks at all segments starting at or after \( i+1 \).

The first sum looks at all segments starting at \( i \).

The other case is

\[
\begin{cases} j:=i, s:=0. B & \text{if } i=\#L \text{ then } ok \\ j:=j+1. B \text{ fi} & \text{else}
\end{cases}
\]

The refinement of \( B \) can be broken into various cases.

\[
B \equiv \begin{cases} s=n \wedge j=\#L \wedge (c:=c+1) & \text{if } s=n \wedge j=\#L \wedge (c:=c+1) \\ s:=s+Lj. j:=j+1. B & \text{else}
\end{cases}
\]

The second sum looks at all segments starting at or after \( i+1 \).

Together, they look at all segments starting at or after \( i \).
\[ B \iff s < n \land j = \#L \land (s := s + Lj, j := j + 1). \]

All five are very easy, so I leave them here. The disjunct \( s > n \) is not necessary for correctness. Without it, execution time is exactly \( \#L \times (\#L + 1)/2 \). With it, that's an upper bound. So for time,

- replace \( R \) with \( t' \leq t + \#L \times (\#L + 1)/2 \)
- replace \( A \) with \( i \leq \#L \implies t' \leq t + (\#L - i) \times (\#L - i + 1)/2 \land i' \leq \#L \)
- replace \( B \) with \( j \leq \#L \implies t' \leq t + \#L - j \land j' \leq \#L \land i' = i \)

Again, easy.

(b) Write a program to find, in a given list of positive naturals, the number of segments whose sum is a given natural.