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

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

After trying the question, scroll down to the solution.
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 \( 0 \leq a \leq b \leq \#L \land (\Sigma L [a;..b]) = n \)”. The quantifier § turns a predicate into a bunch, and then \( \Sigma \) tells the size of the bunch, but unfortunately § works on only one variable, not two. Still, we can sum up the sizes. Formally,

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

Suppose the items of \( L \) are all \( 0 \), and \( n \neq 0 \). Then there are \((\#L+1) \times (\#L+2)/2\) segments with the right sum, so the best solution is probably quadratic. Let \( i, j, s, \) and \( c \) be natural variables. The desired result of the computation is \( R \), defined as

\[
R = c' = \Sigma a \cdot b \cdot 0 \leq a \leq b \leq \#L \land (\Sigma L [a;..b]) = n
\]

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

\[
A = c' = c + \Sigma a \cdot b \cdot 0 \leq i \leq b \leq \#L \land (\Sigma L [i;..b]) = n
B = i'=i \land c' = c + \Sigma b \cdot 0 \leq j \leq b \leq \#L \land s + (\Sigma L [j;..b]) = n
\]

Now the refinements are

\[
R \iff i=0. \ c:=0. \ A
A \iff j:=i. \ s:=0. \ B. \ \text{if} \ i=\#L \ \text{then} \ ok \ \text{else} \ i:=i+1. \ A \ fi
B \iff \text{if} \ s=n \ \text{then} \ c:=c+1 \ \text{else} \ ok \ \text{fi}.
\]

We prove the refinement of \( R \) by two substitutions. The refinement of \( A \) can be proven by cases. First:

\[
j:=i. \ s:=0. \ B. \ i=\#L \ \text{ok}
\]

substitutions in \( B \)

\[
i'=i \land c' = c + \Sigma b \cdot 0 \leq i \leq b \leq \#L \land (\Sigma L [i;..b]) = n. \ i=\#L \ \text{ok}
\]

remove sequential composition

\[
i'=i=\#L \land c' = c + \Sigma b \cdot 0 \leq i \leq b \leq \#L \land (\Sigma L [i;..b]) = n
\]

Since \( i=\#L \), the sum is just the single value when \( i=b=\#L \).

So it doesn’t change anything to put an \( a \) in there, \( i=a=b=\#L \).

\[
i'=i=\#L \land c' = c + \Sigma a \cdot b \cdot 0 \leq i \leq a \leq \#L \land (\Sigma L [a;..b]) = n
\]

\[
\Rightarrow A
\]

The other case is

\[
j:=i. \ s:=0. \ B. \ i=\#L \land (i:=i+1. \ A)
\]

substitutions into \( B \) and \( A \)

\[
i'=i \land c' = c + \Sigma b \cdot 0 \leq i \leq b \leq \#L \land (\Sigma L [i;..b]) = n.
\]

\[
c' = c + \Sigma a \cdot b \cdot 0 \leq i \leq+1 \leq a \leq \#L \land (\Sigma L [a;..b]) = n
\]

remove sequential composition

\[
c' = c + \Sigma b \cdot 0 \leq i \leq b \leq \#L \land (\Sigma L [i;..b]) = n
\]

\[
+ \Sigma a \cdot b \cdot 0 \leq i \leq+1 \leq a \leq \#L \land (\Sigma L [a;..b]) = n
\]

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

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

Together, they look at all segments starting at or after \( i \).

\[
\Rightarrow A
\]

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

\[
B \iff s=n \land j=\#L \land (c:=c+1)
\]

\[
B \iff s=n \land j=\#L \land (c:=c+1. \ s:=s+L. \ j:=j+1. \ B)
\]

\[
B \iff s>n \land ok
\]

\[
B \iff s<n \land j=\#L \land ok
\]

\[
B \iff s<n \land j=\#L \land (s:=s+L. \ j:=j+1. \ B)
\]
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.
no solution given