216 (diagonal) Some points are arranged around the perimeter of a circle. The distance from each point to the next point going clockwise around the perimeter is given by a list. Write a program to find two points that are farthest apart.

After trying the question, scroll down to the solution.

§ If necessary, we will assume that there are at least two points. We want to find two points that most nearly make a diagonal. Equivalently, we want to find two points that are farthest apart around the perimeter. Equivalently, we want to find a segment of the list whose sum is most nearly half the total list sum. Let the list be L (a constant). We can indicate the two points by the final values of natural variables m and n. Formally, the problem is P where

$$P \;=\; \forall i,j\cdot\ 0{\le}i{\le}j{\le}\#L \;\Rightarrow\; abs\,((\Sigma L) - 2{\times}\Sigma L[m';..n']) \le abs\,((\Sigma L) - 2{\times}\Sigma L[i;..j])$$

We introduce natural variables k and l to indicate a segment of the list, variable c to be twice the sum of the segment $k;..l$, variable d to be twice the sum of the "best" segment $m;..n$ so far, and variable s to be the sum of the entire list (the perimeter). Formally

$$\begin{aligned} A \;=\;\; & (\forall i,j\cdot\ 0{\le}i{\le}j{\le}l{\le}\#L \;\wedge\; 0{\le}i{\le}k{\le}l{\le}\#L \\ & \qquad \Rightarrow\; abs\,((\Sigma L) - 2{\times}\Sigma L[m;..n]) \le abs\,((\Sigma L) - 2{\times}\Sigma L[i;..j])) \\ & \wedge\; (c = 2{\times}\Sigma L[k;..l]) \;\wedge\; (d = 2{\times}\Sigma L[m;..n]) \;\wedge\; (s = \Sigma L) \end{aligned}$$

Now the refinements.

$P \;\Leftarrow\; k{:=}\,0.\ l{:=}\,0.\ c{:=}\,0.\ m{:=}\,0.\ n{:=}\,0.\ d{:=}\,0.\ s{:=}\,\Sigma L.\ A{\Rightarrow}P$

$s{:=}\,\Sigma L \;\Leftarrow\;$ see book pages 44 and 67

$A{\Rightarrow}P \;\Leftarrow$

if $l{=}\#L \wedge c{\le}s$ **then** ok
else **if** $c{\le}s$ **then** $c{:=}\,c + 2 \times L\,l.\ \ l{:=}\,l{+}1$ **else** $c{:=}\,c - 2 \times L\,k.\ \ k{:=}\,k{+}1$ **fi**.
if $abs\,(s{-}c) < abs\,(s{-}d)$ **then** $m{:=}\,k.\ \ n{:=}\,l.\ \ d{:=}\,c$ **else** ok **fi**.
$A{\Rightarrow}P$ **fi**

For the time, insert $t{:=}\,t{+}1$ in front of the recursive call, and

replace P by $t' \le t + 3{\times}\#L$
replace $s{:=}\,\Sigma L$ by $t' = t + \#L$
replace $A{\Rightarrow}P$ by $t' \le t + 2{\times}\#L - k - l$