$\text{prod}.A.i = \langle \prod j : 0 \leqslant j < i : A.j \rangle$.

Const N : Int; A : Array[0, N) of Int;
Var r : Int;
$\{N \geqslant 0\}$
S
$\{r = \langle \text{summ } i : 0 \leqslant i < N \wedge \text{prod}.A.i < A.i : A.i \rangle\}$

a)La suma de aquellos elementos, cuya productoria de los elementos anteriores sea menor al elemento a sumar.

Derivación:

Paso 1 (invariante)
Proponemos la técnica cambio de variable por constante:
$INV = r = \langle \text{summ } i : 0 \leqslant i < n \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge 0 \leq n \leq N$
B = n < N
¿Vale INV ^ ¬B → Q? veamos.
$r = \langle \text{summ } i : 0 \leqslant i < n \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge 0 \leq n \leq N \wedge n \geq N$
≡{ por lógica
$r = \langle \text{summ } i : 0 \leqslant i < n \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge 0 \leq n \leq N \wedge (n > N \vee n = N)$
≡{ distributiva
$r = \langle \text{summ } i : 0 \leqslant i < n \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge (0 \leq n \leq N \wedge n > N) \vee (0 \leq n \leq N \wedge n = N)$
≡{principio de no contradicción
$r = \langle \text{summ } i : 0 \leqslant i < n \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge (\text{false} \vee (0 \leq n \leq N \wedge n = N))$
≡{ logica, neutro de disyuncion
$r = \langle \text{summ } i : 0 \leqslant i < n \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge 0 \leq n \leq N$

luego INV ^ ¬B equivale a Q, por lo que resulta válida el invariante y la guarda

Paso 2 (Inicializamos)
Inicializamos las dos variables que tenemos y proponemos S1= r,n := E,F
veamos {P} S1 {INV}
asumimos P
wp.(r,n := E,F).(INV)
≡{ def de wp
$E = \langle \text{summ } i : 0 \leqslant i < F \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge 0 \leq F \leq N$
≡{ forzamos rango vacío con F = 0
$E = \langle \text{summ } i : 0 \leqslant i < 0 \wedge \text{prod}.A.i < A.i : A.i \rangle \wedge 0 \leq 0 \leq N$
≡{ rango vacio, lógica
E = 0 ^ 0 ≤ N
≡{ hipótesis, elegimos E = 0
true

Luego S1 = r,n := 0,0

Paso 3 (cota) Proponemos t = N - n ya que N es una constante y podemos hacer crecer n para que la cota decrezca, y si llegara a cero la guarda se habrá hecho falsa, para ello veamos INV ^ B → t ≥ 0

Asumimos INV ^ B ≡ r = ⟨ summ i : 0 ⩽ i < n ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ n < N
N - n ≥ 0
≡{ aritmética
N ≥ 0
≡{ hipótesis
true

Paso 4 (cuerpo del ciclo) Buscamos el cuerpo del ciclo que haga decrecer t y preserve INV, proponemos la asignación S2 = (r,n := E,n+K)
asumimos INV ^ B ≡ r = ⟨ summ i : 0 ⩽ i < n ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ n < N
wp.(r,n := E,n+K).(INV)
≡{ def de wp
E = ⟨ summ i : 0 ⩽ i < n+K ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ n+K ≤ N
≡{ queremos recorrer cada posición del arreglo por lo que proponemos k = 1
E = ⟨ summ i : 0 ⩽ i < n+1 ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ n+1 ≤ N
≡{ por lógica
0 ≤ n+1 ≤ N es lo mismo que
0 ≤ n+1 ^ n + 1 ≤ N
0 ≤ n ^ n < N
0 ≤ n < N
≡{ luego por hipótesis
E = ⟨ summ i : 0 ⩽ i < n+1 ∧ prod.A.i < A.i : A.i ⟩ ^ true
≡{ neutro de ^, logica en el rango}
E = ⟨ summ i : 0 ⩽ i ≤ n ∧ prod.A.i < A.i : A.i ⟩
≡{ logica en el rango
E = ⟨ summ i : (0 ⩽ i < n v i = n) ∧ prod.A.i < A.i : A.i ⟩
≡{ distributiva
E = ⟨ summ i : (0 ⩽ i < n ^ prod.A.i < A.i) v (i = n ^ prod.A.i < A.i) : A.i ⟩
≡{ partición de rango
E = ⟨ summ i : 0 ⩽ i < n ^ prod.A.i < A.i : A.i ⟩ + ⟨ summ i : i = n ^ prod.A.i < A.i : A.i ⟩
≡{ hipótesis, rango unitario y condición
E = r + (prod.A.n < A.n) → A.n
        ¬(prod.A.n < A.n) → 0

Como no podemos seguir programando, fortalecemos el invariante
E = r + (r2 < A.n) → A.n   Con r2 = ⟨ ∏ j : 0 ⩽ j < n : A.j ⟩
        ¬(r2 < A.n) → 0
Nuevo invariante

INV’ ≡ INV ^ r2 = ⟨ ∏ j : 0 ⩽ j < n : A.j ⟩

Inicializamos nuevamente proponemos S1‘ = (r,r2,n := E,F,D)
asumimos P
wp.(r,r2,n := E,F,D).(INV’)
≡{ def de wp
E = ⟨ summ i : 0 ⩽ i < F ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ F ≤ N ^ r2 = ⟨ ∏ j : 0 ⩽ j < F : A.j ⟩
≡{ proponemos rango vacío con F = 0
E = 0 ^ 0 ≤ 0 ≤ N ^ r2 = 1
≡{ logica, hipotesis
E = 0 ^ r2 = 1
≡{ elegimos convenientemente
true
Luego S1’ = r,r2,n := 0,1,0

Cuerpo del ciclo nuevamente, proponemos S2’ = r,r2,n := E,F,n+K
asumimos INV’ ^ B o equivalentemente
r = ⟨ summ i : 0 ⩽ i < n ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ n < N ^ r2 = ⟨ ∏ j : 0 ⩽ j < n : A.j ⟩
wp.(r,r2,n := E,F,n+K).(INV’)
≡{ def de wp
E = ⟨ summ i : 0 ⩽ i < n+K ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ n+K ≤ N ^ F = ⟨ ∏ j : 0 ⩽ j < n+K : A.j ⟩
≡{ proponemos aumentar n en 1 nuevamente
<u>E = ⟨ summ i : 0 ⩽ i < n+1 ∧ prod.A.i < A.i : A.i ⟩ ^ 0 ≤ n+1 ≤ N</u> ^ F = ⟨ ∏ j : 0 ⩽ j < n+K : A.j ⟩
≡{ mismos pasos que antes
E = r + (r2 < A.n) → A.n
    ¬(r2 < A.n) → A.n
^ F = ⟨ ∏ j : 0 ⩽ j < n+1 : A.j ⟩
≡{ lógica en el rango
F = ⟨ ∏ j : 0 ⩽ j < n v j = n : A.j ⟩
≡{ partición de rango
F = ⟨ ∏ j : 0 ⩽ j < n : A.j ⟩ * ⟨ ∏ j : j = n : A.j ⟩
≡{ hipótesis, rango unitario
F = r2 * A.n
≡{ elegimos convenientemente
E = r + (r2 < A.n) → A.n
    ¬(r2 < A.n) → 0

Planteamos un condicional separando por casos
Caso r2 < A.n
E = r + A.n
≡{ elegimos convenientemente
true
Caso ¬(r2 < A.n)
E = r

≡{ elegimos convenientemente
true

Finalmente:
Const N : Int; A : Array[0, N) of Int;
Var r : Int;
$\{N \geqslant 0\}$
r,r2,n := 0,1,0
<u>do</u>(n < N) →
    S1
    <u>if</u>(r2 < A.n) →
        r,r2,n := r + A.n, r2 * A.n, n+1
    else(r2 ≥ A.n) →
        r,r2,n := r, r2 * A.n, n+1
    <u>fi</u>
<u>od</u>
    S2
$\{r = \langle \text{summ } i : 0 \leqslant i < N \wedge prod.A.i < A.i : A.i \rangle\}$

Testeamos con
[1,2,6,11,2] = 2 + 6 = 8

La suma de aquellos elementos, cuya productoria de los elementos anteriores sea menor al elemento a sumar.

r	r2	n	estado
0	1	0	S1
0	1	1	S1
2	2	2	S1
8	12	3	S1
8	132	4	S1
8	264	5	S2

Funciona!!!!