We want to find the smallest number in \(0\ldots n\) with property \(p\). Linear search solves the problem. But evaluating \(p\) is expensive; let us say it takes time \(1\), and all else is free. The fastest solution is to evaluate \(p\) on all \(n\) numbers concurrently, and then find the smallest number that has the property. Write a program without concurrency for which the sequential to concurrent transformation gives the desired computation.

After trying the question, scroll down to the solution.
We introduce array $A: [n*bin]$. We define the desired result $R$, invariant $I$, and helper specification $P$ as follows.

$R = \neg(\exists j: 0..h \cdot p_j) \land (p \cdot h' \lor h'=n)$

$I = \forall j: 0..i \cdot A_j = p_j$

$P = I_n \land \neg(\exists j: 0..h' \cdot p_j) \Rightarrow R$

Now the program is

$R \iff I_0 \Rightarrow I_n \cdot h := 0. P$

$I_0 \Rightarrow I_n \iff \text{for } i := 0..n \text{ do } I_i \Rightarrow I'(i+1) \text{ od}$

$I_i \Rightarrow I'(i+1) \iff A_i := p_i$

$P \iff \text{if } h=n \text{ then ok else if } A \text{ then ok else } h := h+1. P \text{ fi fi}$

The $n$ iterations of the for-loop can be executed concurrently.

We can express the result of the sequential to concurrent transformation at source as follows.

$R \iff I_0 \Rightarrow I_n \cdot h := 0. P$

$I_0 \Rightarrow I_n \iff i := 0. I_i \Rightarrow I'_n$

$I_i \Rightarrow I'_n \iff \text{if } i=n \text{ then ok else if } A \text{ then ok else } (i := i+1. I_i \Rightarrow I'_n) \text{ fi}$

$P \iff \text{if } h=n \text{ then ok else if } A \text{ then ok else } h := h+1. P \text{ fi fi}$

To understand the execution, it might help to unroll the recursion a little: in the refinement of $I_i \Rightarrow I'_n$, replace the recursive call $I_i \Rightarrow I'_n$ by what it calls:

$\text{if } i=n \text{ then ok else } A \text{ then ok else } (i := i+1. I_i \Rightarrow I'_n) \text{ fi}$

And maybe do the same once more.