Introducing some notation

Finding the maximal element in array \( a \) is the problem. Let \( m \) be the variable to hold the max. Let the \texttt{co} statement specify some number of concurrent processes. Let \(<\) and \(>\) denote atomic actions. The goal of the program can then be expressed in predicate logic as:

\[
(\forall j : 1 \leq j \leq n : m \geq a[j]) \land (\exists j : 1 \leq j \leq n : m == a[j])
\]

Let’s look carefully at each successive version of this program, and see what we can discover regarding the issues of synchronization.

1 Sequential version

```c
int m = 0;
for [i = 0 to n-1] {
    if (a[i] > m)
        m = a[i];
}
```

2 Version 2

Let’s fully parallelize the loop with the \texttt{forall} statement, which executes all iterations in parallel:

```c
int m = 0;
forall [i = 0 to n-1] {
    if (a[i] > m)
        m = a[i];
}
```
3  Version 3

Let’s make the processes’ actions atomic:

```c
int m = 0;
forall [i = 0 to n-1] {
    if (a[i] > m)
        m = a[i];
}
```

4  Version 4

Let’s make only part of each processes’ actions atomic:

```c
int m = 0;
forall [i = 0 to n-1] {
    if (a[i] > m)
        m = a[i];
}
```

5  Version 5

Let’s combine parts of the last two versions:

```c
int m = 0;
forall [i = 0 to n-1] {
    if (a[i] > m)
        if (a[i] > m)
            m = a[i];
}
```