

## Bakery Algorithm

Critical section for  $n$  processes

- Before entering its critical section, process receives a number. Holder of the smallest number enters the critical section.
- If processes  $P_i$  and  $P_j$  receive the same number, if  $i < j$ , then  $P_i$  is served first; else  $P_j$  is served first.
- The numbering scheme always generates numbers in increasing order of enumeration; i.e., 1,2,3,3,3,3,4,5...

## Bakery Algorithm (Cont.)

- Notation  $<\equiv$  lexicographical order (ticket #, process id #)
  - $(a,b) < (c,d)$  if  $a < c$  or if  $a = c$  and  $b < d$
  - $\max(a_0, \dots, a_{n-1})$  is a number,  $k$ , such that  $k \geq a_i$  for  $i = 0, \dots, n-1$
- Shared data

**var** *choosing*: **array**  $[0..n-1]$  **of** *boolean*;  
*number*: **array**  $[0..n-1]$  **of** *integer*;

Data structures are initialized to *false* and 0, respectively

## Bakery Algorithm (Cont.)

**repeat**

```
choosing[i] := true;  
number[i] := max(number[0], number[1], ..., number[n - 1]) + 1;  
choosing[i] := false;  
for j := 0 to n - 1  
  do begin  
    while choosing[j] do no-op;  
    while number[j] ≠ 0  
      and (number[j], j) < (number[i], i) do no-op;  
  end;
```

critical section

```
number[i] := 0;
```

remainder section

**until** *false*;