3. For the fourth attempt (Algorithm 3.9):

   (a) Construct the tabular form of the scenario for starvation shown in Figure 3.5.

   (b) Explain why this scenario is considered to be starvation and not livelock as in the third attempt.

   (c) Construct a scenario in which one process is starved while the other enters its critical section infinitely often.

4. Peterson's algorithm for the critical section problem is based upon Dekker's algorithm but is more concise because it collapses two await statements into one with a compound condition. Prove the correctness of Peterson's algorithm by constructing a state diagram for an abbreviated version of the algorithm.

Algorithm 3.13: Peterson's algorithm

```
boolean wantp ← false, wantq ← false
integer last ← 1
```

p		q	
	loop forever		loop forever
p1:	non-critical section	q1:	non-critical section
p2:	wantp ← true	q2:	wantq ← true
p3:	last ← 1	q3:	last ← 2
p4:	await wantq = false or last = 2	q4:	await wantp = false or last = 1
p5:	critical section	q5:	critical section
p6:	wantp ← false	q6:	wantq ← false

(In our presentation, the compound condition is atomic because it is in one labeled line; the algorithm also works if the two conditions are evaluated non-atomically.)

5. (Buhr and Haridi [16]) Compare the use of the variable turn in Dekker's algorithm with the use of the variable last in Peterson's algorithm. What advantage does Dekker's algorithm have? Does it matter if the order of the assignments in the postprotocol is switched?