A Brief Proof to Starvation Freedom

```
// Alice

A_wants = true;
turn = B;
while (B_wants && turn == B);
frob(&x);   // critical section
A_wants = false
```

```
// Bob

B_wants = true;
turn = A;
while (A_wants && turn == A);
borf(&x);  // critical section
B_wants = false
```

Theorem: (Peterson's algorithm guarantees starvation freedom) While Alice wants to execute her critical section, Bob cannot execute his critical section twice in a row, and vice versa.

Proof: Assume for the purpose of contradiction that Alice wanted to execute her critical section but found Bob execute his critical section twice in a row.

Consider the state after Bob finished his first visit to his critical section and exit, he set `B_wants = false` , now we have:

$$\text{turn} = \text{B}$$
$$\text{A\_wants} = \text{true}$$
$$\text{B\_wants} = \text{false}$$

Then Alice's while loop was false which allowed her to pass through.

However, Bob might attempt to enter his critical section before Alice had started to run `frob` , he set `B_wants = true` and `turn = A` , then Bob's while loop is true and he had to spin and wait, while Alice's while loop is still false so she could enter her critical section safely.

Contradiction.