Explain briefly the Peterson's algorithm and describe how it achieves mutual exclusion.

Peterson’s algorithm is an algorithm to allow processes to use shared resources without conflict and only with memory as communication. Competing processes can be divided into two parts, one is called critical section where the process access the shared resource and the second part is noncritical section where the process don’t access any shared processes. The mutual exclusion means that processes never can be in the critical section at the same time.

How Peterson’s algorithm is constructed. We have two processes, A and B. First in our processes they do the non-critical code (not shared processes). As we can see from the pseudo-code A and B can never be in the critical section at the same time. If A is inside critical section, flag[0] is true or turn = 1 which means that B is stuck in the busy wait. Also the variable turn can only be either 1 or 0 which is a condition for being in either A or B.

void A(){

 while(true){

 /\*\* do non critical code \*\*/

 flag[0] = true;

 turn = 1;

 while(flag[1] && turn == 1) { /\*\* do nothing, busy wait \*\*/ }

 /\*\* critical section \*\*/

 flag[0] = false;

 }

}

void B(){

 while(true){

 /\*\* do non-critical code \*\*/

 flag[1] = true;

 turn = 0;

 while(flag[0] && turn == 0) { /\*\* do nothing, busy wait \*\*/ }

 /\*\* critical section \*\*/

 flag[1] = false;

 }

}