CIS 452 02 – Assignment 6 – Gabe Baksa

Peterson's solution is a method that can be used to allow two processes to enter their critical section exclusively (only one process in their critical section at a time). For this in-class assignment you will create the pseudo code for two processes: "Process A" and "Process B". The two variables shared between the processes are flag and turn. Each process should attempt to enter its critical section and print "Hello World\n" using Peterson's solution.

Once you have the pseudocode written you will need to simulate the execution of each of the processes. Two simulations are expected, one in which process A will enter its critical section first and one in which process B will enter its critical section. Please use a different color font to represent each of the processes.

For example, if I have two processes that are context switched between them after the first process outputs "Hello World\n" with the printf statement.
**********Your work starts below here! **********

Program A:

```
boolean test_and_set (boolean *target)
        {
            boolean returnValue = *target;
            *target = TRUE;
            return returnValue:
        }
do {
        while (test_and_set(&lock))  { }       // do nothing
              /* critical section */
        lock = false;
              /* remainder section */
     } while (true);
```

Program B:

```
boolean test_and_set (boolean *target)
        {
            boolean returnValue = *target;
            *target = TRUE;
            return returnValue:
```

```
        }
do {
        while (test_and_set(&lock))  { }     // do nothing
              /* critical section */
        lock = false;
              /* remainder section */
     } while (true);
```

Copy and paste lines of code to simulate the execution of commands that will demonstrate how two processes use test and set for a shared variable ‘lock’.

lock = false;

```
do {
        while (test_and_set(&lock)) //returns false and sets lock to true
{
            boolean returnValue = *target;
            *target = TRUE;
            return returnValue:
        }
 do {
        while (test_and_set(&lock))  //lock is already true so returns true
{
            boolean returnValue = *target;
            *target = TRUE;
            return returnValue:
        }
{} //doing nothing because lock is true
/* critical section */
        lock = false; //does critical and sets lock
 do {
        while (test_and_set(&lock))  //lock is now false so sets it to true
{
            boolean returnValue = *target;
            *target = TRUE;
            return returnValue:
        }
/* critical section */
```

```
        lock = false;
/* remainder section */
/* remainder section */
```