Homework assignment 5

T-106.420 Concurrent Programming

Family name: **Chaparro González**
First name: **Diego**
Student number: **59881P**
dchaparro@acm.org

11th December 2002

Contents

1 Exercise

```
monitor Semaphores {

        int np = 0;
        int nv = 0;
        int p = 0;
        int v = 0;
        cond posP;
        cond posV;
        cond enoughV;

        procedure PsemP () { # P() of operation P
                while (p == 0) wait (posP);
                while (np == nv) wait (enoughV);
                p = p - 1;
        }

        procedure VsemP () { # V() of operation P
                p = p + 1;
                signal (posP)
                np = np + 1;
        }

        procedure PsemV () { # P() of operation V
                while (v == 0) wait (posV);
                v = v - 1;
        }

        procedure VsemV () { # V() of operation V
                v = v + 1;
                signal (posV);
                nv = nv + 1;
                signal (enoughV);
        }
}
```

The signaling discipline that I've used is Signal and Wait (SW).

2 Exercise

2.1 a)

I think that this solution is correct.

2.2 b)

I think that this solution is incorrect. The problem is that the two kind of processes (bear and bees) use the same semaphore to enter in their critical sections. For example, if the bear starts to execute first, it passes the semaphore mutex and then it will sleep for the semaphore full. Then the other processes start their execution and they will sleep in the instruction P(mutex) forever, because the mutex is equal to 0. The processes will sleep forever.

2.3 c)

I think that this solution is incorrect. The two semaphores are initialized to 0, and when the processes (bear and bees) start their execution, the first instruction that they execute is a P() operation, and they will never enter in their critical sections because the initial value of the semaphore is 0. The processes will sleep forever.