SISTEMI OPERATIVI IIN/IEL/IDT
INFORMATICA INDUSTRIALE E SISTEMI OPERATIVI IDI
SISTEMI DI ELABORAZIONE P.O.
prova scritta preliminare del 20.11.2003

Nome: _____ Cognome: _____

In un sistema sono in esecuzione K processi di due tipi diversi, rispettivamente di tipo P_1 e di tipo P_2. Il sistema mette a disposizione tre tipi di risorse: L risorse di tipo A, M risorse di tipo B e N risorse di tipo C.
I processi di tipo P_1 concorrono all'uso di risorse di tipo A e tipo C. I processi di tipo P_2 concorrono all'uso di risorse di tipo B e tipo C. In particolare:

- l'attività dei processi di tipo P_1 si alterna tra le chiamate useResourceAC() e nonCriticalSection(). La funzione useResourceAC() richiede una risorsa di tipo A ed una di tipo C;
- l'attività dei processi di tipo P_2 si alterna tra le chiamate useResourceBC() e nonCriticalSection(). La funzione useResourceBC() richiede una risorsa di tipo B ed una di tipo C.

Le chiamate useResourceAC(), useResourceBC() possono essere invocate solo se le risorse necessarie al loro funzionamento sono disponibili: altrimenti il processo si sospende in attesa che queste si rendano disponibili.

Mostrare in pseudo-codice come possa avvenire, attraverso l'uso di semafori e variabili condivise, la sincronizzazione tra i due tipi di processi descritti considerando che tra i due tipi di processo non è stabilita alcuna priorità

Traccia:

P_1	P_2

while(TRUE) {
???
useResourceAC();
???
nonCriticalSection();
}
	while(TRUE) {
	???
	useResourceBC();
	???
	nonCriticalSection();
	}

SISTEMI OPERATIVI IIN/IEL/IDT
INFORMATICA INDUSTRIALE E SISTEMI OPERATIVI IDI
SISTEMI DI ELABORAZIONE P.O.
soluzione test scritto del 20.11.2003

Variabili condivise e semafori

- mutex(1), semaforo di mutua esclusione;
- SAC(0), SBC(0), semafori sulle risorse condivise
- countA = L, countB = M, countC = N, contatori sul numero delle risorse dei tre tipi
- countSAC = 0, countSBC = 0, contatori sul numero di processi sospesi in attesa delle risorse.

P_1

```
while (true) {
   wait(mutex);
   if (countA==0 || countC==0) {          // sospensione in assenza di risorsa A o C
        countSAC++;
        signal(mutex);
        wait(SAC);
   }
   else {              // acquisizione delle risorse
        countA--;
        countC--;
        signal(mutex);
   }

   useResourceAC();              // uso della risorsa

   wait(mutex);
   if (countSAC>0) {              // risveglio di un processo di tipo1 in attesa
        countSAC--;
        signal(SAC);
   }
   else if (countSBC>0 && countB>0) {          // risveglio di un processo di tipo2 in attesa
        countA++;            // la risorsa A è liberata
        countB--;            // la risorsa B è allocata
        signal(SBC);
        }
   else {       // rilascio senza risveglio se non ci sono processi sospesi per la risorsa
        countA++;
        countC++;
        }
        signal(mutex);
}
```

P_2

```
while (true) {
    wait(mutex);
    if (countB==0 || countC==0) {           // sospensione in assenza di risorsa B o C
        countSBC++;
        signal(mutex);
        wait(SBC);
    }
    else {              // acquisizione delle risorse
        countB--;
        countC--;
        signal(mutex);
    }

    useResourceBC();                // uso delle risorse

    wait(mutex);
    if (countSBC>0) {               // risveglio di un processo di tipo2 in attesa
        countSBC--;
        signal(SBC);
    }
    else if (countSAC>0 && countA>0) {          // risveglio di un processo di tipo1 in attesa
        countA--;    // la risorsa A è allocata
        countB++;    // la risorsa B è liberata
        signal(SAC);
        }
    else {      // rilascio senza risveglio se non ci sono processi sospesi per la risorsa
        countB++;
        countC++;
    }
    signal(mutex);
}
```