UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELETRÔNICA
AUTOMAÇÃO EM TEMPO REAL
PROFESSOR: Constantino Seixas Filho

PROVA I (com consulta)

1) [30 pontos] Atravessando Baboons [Tanenbaum] 

No Parque Nacional de Kruger na África do Sul existe um canyon que separa uma colônia de baboons. Ligando os dois lados há uma corda estendida. Só existe uma maneira de um baboon atravessar de um lado para o outro: se deslocando pendurado na corda caminhando com as mãos.

As limitações existentes são:

- A corda só agüenta 5 baboons simultaneamente.
- Se dois baboons atravessam em direções opostas simultaneamente vão se encontrar em algum ponto, lutar pelo direito de passagem e cair no abismo. Por isso devemos assegurar que uma vez que um baboon ganhou o direito de atravessar nenhum outro baboon do lado oposto seja permitido.
- Não deve haver *deadlock* nem inanição.

Implemente um algoritmo em pseudo linguagem que resolva este problema usando semáforos.

```
enum dir {NORTE, SUL};

Thread baboon  (dir  direção) {
Loop {
     Descansa();
     Protocolo_para_atravessar();      // Quero atravessar
     Atravessa();
     Protocolo_de_complitude();        // Já atravessei
     }  // loop
}
```

2) [30 pontos] Questão de tempo ...

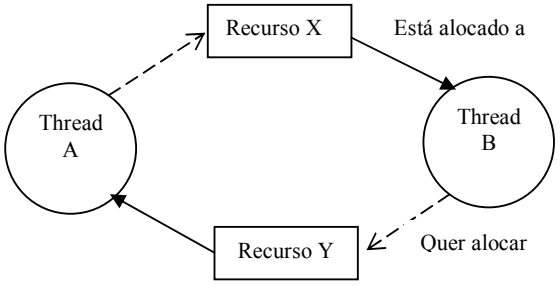
Uma thread espera por um evento (EvAtivação). Quando o evento acontece ela deve emitir um pulso de 140 ms ativando a variável BOOL bSet, inicialmente em 0. Após receber um evento EvTerminar a thread deve sair do loop e terminar. Programe a aplicação usando o conjunto de instruções do WNT.

Temporize utilizando Waitable Timers.

3) [20 pontos] Deadlock

Para que ocorra deadlock quatro condições são necessárias:

a) Exclusão mútua: o recurso não pode ser compartilhado.
b) Guarda e espera. Uma thread mantém um recurso enquanto espera que outro recurso fique livre.
c) Não preempção. Uma vez que uma thread adquire um recurso este recurso não pode lhe ser tomado.
d) Espera circular. Existe um ciclo no diagrama de alocação que mostra threads e recursos, com setas cheias ligando a thread ao recurso já alocado e pontilhada ligando a thread a um recurso desejado.



Baseado nestes critérios analise a situação seguinte e resolva:
Três threads A, B e C compartilham 3 recursos do tipo X, 3 do tipo Y e 2 do tipo Z. Cada recurso é alocado individualmente e em qualquer ordem.

X	Y	Z
3	3	2

Cada thread para executar precisa de 2 X, 2Y e 1Z.

X	Y	Z
2	2	1

Neste exato instante a thread A tem 2 recursos X e um Z. A thread B tem 1 X e 1Y e thread 3 não tem nenhum recurso alocado.

	X	Y	Z
A	2	0	1
B	1	1	0
C	0	0	0

Este sistema está em *deadlock* ? Este sistema pode ficar em *deadlock* ? Por que ? Mostre o grafo de alocação.

4) [20 pontos] Mostre os passos gerais para provar a inexistência de *deadlock* na solução do problema dos leitores e escritores com semáforos (página 134 do livro).

5) [10 pontos] Bônus !!!
Euclides demonstrou no livro IX, proposição 20 de "Os Elementos" que existem infinitos números primos. A demonstração usada é por absurdo.
Suponha que exista um número primo P maior que todos os outros. Considere agora o número composto N = 2 x 3 x 5 x 7 x 11 x ... x P
Agora tome N + 1.
Continue agora a demonstração de Euclides ...