MAC 438 – Programação Concorrente

Instituto de Matemática e Estatística – Primeiro Semestre de 2003

Primeira Prova – 6 de maio de 2003

Nome: ____

Assinatura: ____

Nº USP: ______________

Instruções:

1. Preencha o cabeçalho acima;
2. A prova deve ser resolvida em folhas separadas (com o seu nome);
3. A prova pode ser resolvida a lápis;
4. Esta é uma prova rápida: **Duração: 1h15min**

1. **Questão:** (valor 1.5 pontos) Compartilhamento de recursos. Nós vimos durante o curso que existem várias formas de compartilhamento de recursos. Para se compartilhar um recurso s necessitamos de no mínimo das seguintes primitivas: `Acquire` - solicita o recurso e `Release` - devolve o recurso.

   Proponha uma forma segura de implementar a seguinte interface:

```
interface UsoExclusivo {
  Recurso Aquire();
  void Release();     //  podem receber parâmetros, se nescessário
}
```

   Os processos que recebem o recurso **não** são confiáveis. Isto é, podem tentar usá-lo antes de recebê-lo, após liberar um recurso, através de `release` eles podem tentar acessá-lo, e eles podem tentar liberar o recurso sem tê-lo. A sua implementação deve impedir que dois processos possam acessar um dado recurso ao mesmo tempo. Explique a razão pela qual a sua implementação funciona.

2. **Questão:** (valor 1.5 pontos) Uma das formas de se implementar um monitor em C é através do uso de semáforos e variáveis de condição. Observe o seguinte pseudo-código C (as chamadas `P`, `V`, `signal` e `wait` são as chamadas padrão da biblioteca de P_threads). Pode-se pensar de maneira análogo em Java (uma alternativa ao uso de `synchronized`).

```
int mutex = 1;
cond fila;
void g() {                    void f() {
  P(mutex);                     P(mutex);
  wait(cond);                   h();
  h();                          wait(cond);
  V(mutex);                     V(mutex);
}                             }
void h() {                    void libera() {
  P(mutex);                     P(mutex);
  wait(cond);                   signal(cond);
  V(mutex);                     V(mutex);
}                             }
```

   Quê possível problema pode acontecer ? Como resolvê-lo ?

3. **Questão:** (valor 2.0 pontos) Escreva o código correspondente ao seguinte monitor usando Java ou C.

```
monitor Barber_Shop {
  int barber = 0, chair = 0, open = 0;
  cond barber_available; #signaled when barber > 0
  cond chair_occupied;   #signaled when chair > 0
  cond door_open;        #signaled when open > 0
  cond customer_left;    #signaled when open == 0

  procedure get_haircut() {
    while (barber == 0)
      wait(barber_available);
    barber = barber - 1;
    chair = chair + 1;
    signal(chair_occupied);
    while (open == 0)
      wait(door_open);
    open = open - 1;
    signal(customer_left);
   }
   procedure get_next_customer() {
     barber = barber + 1;
     signal(barber_available);
     while (chair == 0)
       wait(chair_occupied);
       chair = chair - 1;
   }
   procedure finished_cut() {
     open = open + 1;
     signal(door_open);
     while (open > 0)
       wait(customer_left);
   }
}
```