

Exclusão Mútua

Grupo de Sistemas Distribuídos
Universidade do Minho

1 Objectivos

Evitar *corridas*, pela utilização de mecanismos que garantam *exclusão mútua* na execução de *secções críticas*. Observar a ocorrência de *deadlock*, e implementar solução que garanta a sua ausência.

2 Mecanismos

Mecanismo nativo de exclusão mútua, via *lock* disponível em todos os objectos, em duas variantes:

- método `synchronized`, que usa o lock de `this`:
`synchronized T m(...) { ... }`
- bloco `synchronized`, que usa o lock de `obj`:
`synchronized (obj) { ... }`

O lock é reentrante, permitindo que uma thread invoque aninhadamente/recursivamente métodos/blocos sobre o mesmo objecto.

3 Exercícios propostos

1. Modifique o exercício anterior – incremento concorrente de um contador partilhado — de modo a garantir a execução correcta do programa.
2. Implemente uma classe `Banco` que ofereça os métodos de consulta, crédito e débito de valores sobre um número fixo de contas (com saldo inicial nulo). Utilize exclusão mútua ao nível do objecto `Banco`.
3. Acrescente o método `transferir` à classe `Banco` como composição das operações de débito e crédito de um valor sobre duas contas.
4. Reimplemente a classe `Banco` utilizando exclusão mútua ao nível das contas individuais.