



Nome:

.....

ENGENHARIA INFORMATICA – UNIVERSIDADE DO MINHO

Exame de Sistemas Distribuídos

21 de julho de 2021 – Duração: 2h00

Número:

<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0	<input type="checkbox"/> 0
<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1	<input type="checkbox"/> 1
<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2	<input type="checkbox"/> 2
<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3	<input type="checkbox"/> 3
<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4	<input type="checkbox"/> 4
<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5	<input type="checkbox"/> 5
<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6	<input type="checkbox"/> 6
<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7	<input type="checkbox"/> 7
<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8	<input type="checkbox"/> 8
<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9	<input type="checkbox"/> 9

Instruções: Preencha o nome e o número de aluno nesta folha pintando complementemente as caixas correspondentes a cada algarismo; em cada pergunta de escolha múltipla há sempre uma ou mais respostas certas; para as assinalar pinte completamente as caixas correspondentes; não use as áreas sombreadas; preencha também o nome e número em cada folha de exame adicional.

Grupo I

Responda a este grupo no próprio enunciado.

1. Em programas concorrentes que usam primitivas de sincronização para delimitar e controlar o acesso a secções críticas, a sua eficiência depende:

- ☐ da duração em absoluto de cada uma das secções críticas
- ☐ da probabilidade de diferentes threads tentarem usar a mesma secção crítica ao mesmo tempo
- ☐ da eficiência das primitivas de sincronização usadas para exclusão mútua
- ☐ apenas do número de threads que são iniciados ao mesmo tempo

2. Considere o problema de sincronização de relógios físicos em servidores dispersos na Internet e a sua resolução com o algoritmo *Network Time Protocol* (NTP). Este algoritmo:

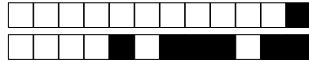
- ☐ não necessita de uma rede de difusão para funcionar
- ☐ torna dispensável a existência de uma referência de tempo fiável
- ☐ é indispensável para a realização de tarefas de coordenação tais como a exclusão mútua distribuída
- ☐ funciona melhor quando os atrasos na rede entre os servidores envolvidos são simétricos (i.e., iguais nos dois sentidos)

3. Considere um sistema distribuído em que há migração de código do servidor, que guarda e manipula estado persistente, para um cliente interativo, com uma interface do utilizador. Esta migração:

- ☐ contribui principalmente para a redução da quantidade de dados transferidos na rede
- ☐ é indispensável para a transparência de acesso
- ☐ contribui principalmente para a redução da latência percebida pelo utilizador
- ☐ resulta num compromisso entre a eficiência de execução do código migrado e a segurança do servidor

4. O protocolo *2-phase commit* (2PC) pode bloquear em caso de falha, sendo incapaz de produzir um resultado, quando:

- ☐ o coordenador falha depois de enviar a proposta no início da primeira fase
- ☐ o coordenador falha antes de enviar a decisão no início da segunda fase
- ☐ um participante falha antes de enviar uma resposta na primeira fase
- ☐ uma maioria de participante falham a seguir a enviar uma resposta na primeira fase



5. Identifique o principal desafio na implementação de *middleware* de invocação remota que suporte clientes com vários *threads* e invocações concorrentes ao mesmo servidor. Descreva sucintamente uma solução.

☐ 0 ☐ .1 ☐ .2 ☐ .3 ☐ .4 ☐ .5 ☐ .6 ☐ .7 ☐ .8 ☐ .9 ☐ 1 *cotação*

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Grupo II

Responda a cada pergunta deste grupo numa folha de exame separada.

6. Considere a gestão de uma fila de espera para vacinação, assumindo que cada frasco de vacina serve para NUM utentes e que como tal só pode ser usado quando NUM utentes estiverem prontos. Pretende-se que escreva em Java, fazendo uso de primitivas de controlo de concorrência, uma classe que implemente a seguinte interface para ser usada por *threads* que representam utentes ou o fornecedor de vacinas:

```
interface ControloVacinas {  
    void pedirParaVacinar();  
    void fornecerFrascos(int frascos)  
}
```

Quando um utente chega e pretende ser vacinado deve invocar o método `pedirParaVacinar`, que irá bloquear até estarem reunidas as condições de início de vacinação. O método `fornecerFrascos(int frascos)` sinaliza a entrada de mais frascos na unidade de saúde. Considere que o sistema arranca com 0 frascos de vacinas disponíveis.

Valorização: Garanta que os utentes não são ultrapassados no acesso à vacina tendo em conta a ordem de chegada. Minimize a necessidade de acordar *threads*.

7. Implemente o programa servidor usando *threads*, *sockets* TCP, e a classe desenvolvida na pergunta anterior.