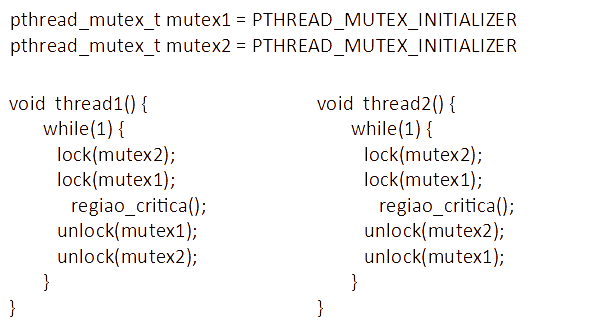
Infraestrutura de Software 2021.2 - Final

1. Explique as definições associadas a sistemas operacionais: (i) máquina estendida; (ii) gerenciador de recursos; e (iii) modo *kernel* (núcleo) (1,0)

2. Explique o conceito de *threads*  e sua implementação em modo usuário e modo núcleo (1,5)

3. Em um sistema com múltiplos processadores, explique as diferentes formas que um sistema operacional pode executar (1,5)

4. O código abaixo pode ter *deadlock (*impasse)? Explique usando as condições necessárias (2,5)



**5. Implemente usando pthreads** um sistema de controle de vacinação. Este sistema é composto por 3 *threads*. **(i)** Uma *thread secsaude* representa a secretaria de saúde que coordena as vacinações com unidades de saúde (***unisaude*).** A *thread secsaude* executa em um laço infinito e só pode iniciar uma nova iteração após a sincronização com as unidades de saúde (***thread unisaude)*.** Após a *thread secsaude* sincronizar com todas *threads* *unisaude*, *secsaude* e *unisaude* podem continuar a execução. **(ii)** Uma ***thread unisaude*** representa uma unidade de saúde e executa em um laço infinito. Primeiramente, ela precisa obter 40 vacinas com o depósito (***thread deposito***). Se o depósito estiver vazio, deve dormir e aguardar uma notificação de disponibilização de novas vacinas. Após obter as vacinas, precisa sincronizar com a *thread secsaude* e, em seguida, iniciar uma nova iteração. (iii) A ***thread deposito***deve disponibilizar 10 vacinas e esta é a quantidade máxima do estoque. Se o estoque estiver cheio, deve dormir e aguardar o uso do estoque. Quando alguma posição do estoque estiver livre, deve preenchê-lo (sempre respeitando a quantidade máxima de 10 unidades).**Obs: Não precisa mostrar a criação das threads, mas você deve demonstrar a utilização de outros recursos necessários para a resposta. (3,5).**int pthread\_mutex\_unlock(pthread\_mutex\_t \*mutex); int pthread\_mutex\_lock(pthread\_mutex\_t \*mutex); int pthread\_cond\_wait(pthread\_cond\_t \*cond, pthread\_mutex\_t \*mutex); int pthread\_cond\_signal(pthread\_cond\_t \*cond);int pthread\_cond\_broadcast(pthread\_cond\_t \**cond*); int pthread\_join(pthread\_t *thread*, void \*\**value\_ptr*); int pthread\_barrier\_wait(pthread\_barrier\_t \**barrier*);