Infraestrutura de Software 2023.2 - 1o EE

1. Instruções *test-and-set* (de CPUs) podem ser utilizadas em sistemas operacionais para implementar exclusão mútua (*mutex, semáforos)*. Como funciona? Essa abordagem funciona para sistemas multiprocessadores? Justifique (2,0)
2. O que é o contexto do processo? Por que a troca de contexto é uma ação importante no desempenho de escalonamento de processos? (2,0)
3. Quais as vantagens e desvantagens de um sistema operacional estruturado de forma monolítica ou micronúcleo? (2,0)
4. Como é possível garantir em um sistema com múltiplas *threads* ou processos a inexistência de *deadlocks*? Justifique usando as condições necessárias. (1,5)

5. Implemente uma API de jogos digitais usando *pthreads*. A biblioteca é composta por 4 *threads:* imagem, som, controle e comunicação . Assuma que cada *thread é* implementada em um laço infinito. Após uma iteração (repetição), cada *thread* deverá ficar parada esperando que as outras *threads* também terminem uma iteração. Após todas as *threads* concluírem uma repetição, elas continuam a respectiva execução e a espera deve ocorrer novamente ao fim da iteração. Considere um código qualquer dentro do laço de cada thread. **Além da implementação, explique brevemente se exclusão mútua é uma técnica apropriada para esse problema.** (2,5)

Obs: Não precisa mostrar a criação das threads, mas você deve demonstrar a utilização de outros recursos necessários para a resposta.

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\_mutex\_init(pthread\_mutex\_t \*mutex, const pthread\_mutexattr\_t \*attr);int pthread\_mutex\_destroy(pthread\_mutex\_t \*mutex); int pthread\_join(pthread\_t *thread*, void \*\**value\_ptr*); int pthread\_barrier\_wait(pthread\_barrier\_t \*barrier); int pthread\_barrier\_init(pthread\_barrier\_t \* barrier, const pthread\_barrierattr\_t \* attr, unsigned int count );