**Infraestrutura de Software - 1º EE (2019.1)**

1. Sistemas operacionais podem ser uma máquina abstrata e proveem uma interface denominada chamada ao sistema (*system call*). **O que é chamada ao sistema (system call) e como funciona (incluindo etapas internas do sistema operacional)**? (2,0)

2. No código abaixo, deadlock pode ocorrer? Explique mostrando um exemplo e a relação com as 4 condições necessárias para que um deadlock ocorra. (2,5)



3. Explique a organização de um kernel micronúcelo (1,5).

4. Faça resumo de como funciona e a inicialização do *bootloader* de 2 etapasem um arquitetura x86*.* Quais são os cuidados necessários no código ASM?(ex: localização disco, memória...). (1,5)

6. **Implemente um sistema** de arrecadação de doação, no qual é composto por várias threads. Ao iniciar a execução, cada thread deve incrementar o contador *doação*, uma variável inteira global, em uma unidade. Ao concluir a execução, cada thread também deverá decrementar o contador em uma unidade. Assuma um mesmo código a ser usado pelas threads **e não se preocupe com o código a ser executado entre as manipulações do contador**. Após a manipulação do contador por uma thread, outras threads deverão ter acesso para incrementá-lo ou decrementá-lo. **Ademais, esse sistema possui condição de disputa? O sistema possui região crítica? Caso positivo, aponte**.

**Obs: Não precisa mostrar a criação das threads, mas você deve demonstrar a utilização de outros recursos necessários para a resposta. (2,5).**

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\_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*)