Sistemas Operacionais - Trabalho 1

Objetivos

1. Utilizar programação *multithreading* e comunicação entre processos.
2. Praticar técnicas de sincronização de processos.

Problema 1: Jantar dos Canibais

Suponha que um grupo de N canibais serve-se a partir de uma grande travessa que comporta M porções. Quando alguém quer comer, ele(ela) se serve da travessa, a menos que ela esteja vazia. Neste caso, o canibal acorda o cozinheiro e espera até que o cozinheiro coloque mais M porções na travessa. Desenvolva um código que:

- Modele os canibais e o cozinheiro como *threads*;
- Implemente e sincronize as ações dos canibais e do cozinheiro.

A solução deve evitar *deadlock* e deve acordar o cozinheiro apenas quando a travessa estiver vazia. Suponha um longo jantar, onde cada canibal continuamente se serve e come, sem se preocupar com as demais coisas na vida de um canibal. Modele N e M como parâmetros de entrada.

Problema 2: O seu *mutex*

O problema anterior pode ser solucionado com o uso de semáforos contadores e *mutex* (ou semáforo binário). Para o *mutex*, substitua seu uso com uma implementação em software das primitivas *acquire()* (ou *lock()*) e *release()* (ou *unlock()*) a ser desenvolvida pelo grupo. Para isso, utilize o algoritmo de Dekker ou o algoritmo de Peterson, generalizado para N processos. Os semáforos contadores devem ser deixados como na solução original.

Entrega: O trabalho pode ser realizado em grupos de 2 ou 3 alunos. Junto com a implementação das soluções para os problemas, deve ser entregue um relatório de 2 ou 3 páginas explicando detalhes da implementação e dificuldades encontradas. A entrega do trabalho deve ser realizada pelo Moodle (código fonte e relatório).