IPRJ – SISTEMAS OPERACIONAIS

LISTA DE EXERCÍCIOS 03

1) O que é uma condição de corrida? Descreva um exemplo de situação que leva a uma condição de corrida.

2) O que é necessário para evitar uma condição de corrida/disputa?

3) Qual é o principal problema existente em todos os métodos de exclusão mútua com espera ociosa?

4) Descreva os processos exclusão mútua atráves da instrução TSL e atráves da solução de Peterson. Qual a principal diferença entre essas duas abordagens?

5) Descreva o processo de exclusão mútua realizado através de semáforos.

6) Qual a principal diferença entre semáforo e mutex?

7) Implemente o problema do Produtor-Consumidor utilizando duas threads produtoras e três threads consumidoras. Utilize semáforos para sincronizar o acesso das threads a região critica.

8) Implemente o jogo "*pedra, papel e tesoura*" utilizando threads como jogadores. Como entrada, tem-se a quantidade de rodadas que serão jogadas. Como saída, deve-se apresentar uma mensagem mostrando o placar final. Utilize mutex para sincronizar o acesso das threads a região critica.

   Lembre-se que *pedra* ganha de *tesoura*, *papel* ganha de *pedra* e *tesoura* ganha de *papel*.

Para sortear a jogada das threads utilize a função `int rand(void);` da biblioteca `stdlib.h`. A função `rand` retorna um número aleatório em um determinado intervalo. Exemplo:

```
x = rand() % 100;  /* x vai receber um valor entre 0 e 100 */
```

Para garantir que novos números aleatórios sejam sorteados em cada execução do programa é necessário executar a função `srand` antes de sortear os números. Exemplo:

```
srand(time(NULL));
```

Para poder utilizar essas funções é necessário incluir no programa as bibliotecas `stdlib.h` e `time.h`.