MAC0438 – Programação concorrente – 1s2012
EP3
Data de Entrega: 18/06/2012

Prof. Daniel Macêdo Batista

1 Problema

Neste EP você deverá simular uma montanha russa. Considere que existem diversas threads passageiros chegando e `m` ($m > 1$) threads carros. Os passageiros esperam para ocupar lugares em um dos `m` carros, sendo que cada carro tem capacidade para `C` passageiros. Entretanto o carro só pode fazer uma volta quando ele estiver cheio. Cada passageiro tem que conseguir andar na montanha-russa duas vezes (depois que ele termina uma volta, considere que seria como se ele estivesse chegando pela primeira vez na fila). Depois disso ele vai embora.

Os passageiros entram nos carros por ordem de chegada, com exceção de passageiros que possuam um bilhete dourado. Esses passageiros sempre podem passar na frente dos passageiros que não tenham o bilhete dourado. Se em um dado momento há mais de um passageiro com bilhete dourado, eles devem se organizar na ordem de chegada.

Há apenas uma pista na montanha russa, portanto não pode haver ultrapassagem entre os carros.

2 Requisitos

2.1 Monitor e semáforos

Neste EP você deverá implementar um monitor para sincronizar os passageiros e os carros. Você deverá implementar **todas** as operações básicas do monitor usando semáforos. EPs que não implementem todas as operações com semáforos receberão nota ZERO. Mesmo que você não utilize todas as operações, você deverá implementá-las e indicar no LEIAME em que arquivo e em quais linhas estão implementadas as operações. Lembrando que as operações são: `empty(cv)`, `wait(cv)`, `wait(cv,rank)`, `signal(cv)`, `signal_all(cv)`, `minrank(cv)`

As operações descritas acima são as operações básicas do monitor. Já as operações utilizadas para a sincronização entre carros e passageiros devem ser:

- `pegaCarona` que deve ser chamada pelos passageiros;
- `carrega` que deve ser chamada pelos carros;
- `descarrega` que deve ser chamada pelos carros.

Você pode criar outras operações para sincronização das threads, porém estas acima são obrigatórias. EPs que não implementem todas as três operações acima terão nota ZERO.

2.2 Linguagem

Seu programa deve ser escrito em C, C++ ou java. Programas escritos em outras linguagens terão nota ZERO.

2.3 Entrada e Saída

O seu EP deverá imprimir na saída padrão as seguintes informações quando eventos especiais acontecerem:

- Quantidade de passageiros esperando para entrar nos carros;
- Lista com o ID dos passageiros esperando para entrar nos carros, em ordem de chegada (passageiros com bilhete dourado deverão ter seus IDs precedidos de 'D') seguidos de ':K', onde K é a quantidade de vezes que o passageiro já andou na montanha-russa;
- Quantidade de passageiros dentro dos carros. Cada carro deve ter um ID que precisa ser impresso;
- Lista com o ID dos passageiros dentro de cada carro (passageiros com bilhete dourado deverão ter seus IDs precedidos de 'D');
- Informação sobre se cada carro está parado ou em movimento.

Os eventos especiais são os seguintes:

- Início do passeio de algum carro;
- Fim do passeio de algum carro;
- Chegada de um passageiro na fila.

O tempo gasto por um carro para completar a volta deve ser fixo e igual a 100 u.t. (unidades de tempo). Seu programa deverá receber como argumento a taxa de chegada com a qual os passageiros chegam. Ou seja, se você passar a taxa de chegada 0,01, isso significa que o intervalo entre a chegada de 1 passageiro e outro é de 100 u.t.. Seu programa também precisará receber como argumento os valores de C e de m.

3 Sobre a entrega

Você deve entregar um arquivo .tar.gz contendo os seguintes itens:

- fonte(s), Makefile (ou similar), arquivo LEIAME.

O desempacotamento do arquivo .tar.gz **deve** produzir um diretório contendo os itens. O nome do diretório **deve** ser ep3-membros_da_equipe. Por exemplo: ep3-joao-maria.

A entrega do .tar.gz **deve** ser feita através do Paca.

O EP pode ser feito individualmente ou em dupla.