MAC0422 - Sistemas Operacionais
Bacharelado em Ciência da Computação
IME/USP – Segundo Semestre de 2011
Terceiro Exercício-Programa

1 Travessia do Amazonas

Você foi contratado para coordenar a travessia do rio Amazonas entre a sua margem direita e esquerda. Para tanto, você dispõe de um único barco capaz de acomodar no máximo 3 (três) pessoas. O barco afundará se mais que 3 pessoas embarcarem. O objetivo deste exercício é familiarizá-lo com os conceitos e uso de mecanismos de sincronização em Linux. Você deverá modelar esse problema de sincronização e implementá-lo utilizando Linux, fazendo uso de processos e mecanismos `Unix IPC`. Você não deverá utilizar POSIX *threads* e seus respectivos mecanismos de sincronização.

1.1 O processo Passageiro

Cada passageiro será representado por um processo que será disparado a partir do prompt e terá como argumento de entrada a margem da onde ele deseja partir. Exemplo:

```
$ ./passageiro 0
passageiro PID=1123 esperando na margem ESQUERDA.
```

onde 0 representa a margem esquerda e 1 a margem direita. Você deve imprimir o identificador do processo que representa a pessoa que espera pelo barco e sua margem de origem. O processo `Passageiro` deve ser estruturado da seguinte forma, para facilitar a correção do seu EP:

```
#define ESQUERDA 0
#define DIREITA  1

void embarca(int margem)
{
  /* Aqui o passageiro embarca na margem especificada se possível,
    ou espera o barco chegar à sua margem do rio */
}

void desembarca(int margem)
{
  /* Aqui o passageiro desembarca do barco vindo da margem especificada
    e realiza quaisquer outras tarefas para dar continuidade à viagem
    de outros passageiros */
}

void atravessa(int margem)
{
  /* O barco atravessa o rio a partir da margem especificada */
}

int main(int argc, char *argv[])
{
  /* lê margem de origem como parâmetro */
```

```
  embarca(partida);
  atravessa(partida);
  desembarca(partida);

  /* imprime passageiro saiu do pier */

  exit(0);
}
```

1.2 Implementação

Obviamente a sincronização deve ocorrer na chegada e saída do barco e a função `atravessa()` não possui papel no problema, devendo imprimir na tela do terminal que a travessia está em execução, em qual sentido, e quem está atravessando.

A função `embarca()` não deve retornar até que seja seguro para uma pessoa atravessar o rio em um determinado sentido (deve garantir que o barco não afundará e que ninguém cairá no rio caso o barco esteja na margem contrária). O barco só inicia a travessia se existirem exatamente 3 pessoas no barco na margem certa do rio (a margem que o barco está estacionado). Se o barco não aparecer em 10 segundos, o passageiro desiste da viagem, imprime este fato e realiza um outro percurso mais longo de ônibus.

A função `desembarca()` é chamada para indicar que o chamador terminou de atravessar o rio e pode preparar a travessia de outras pessoas que estão esperando.

O usuário poderá disparar tantos processos passageiros quanto desejar. Em particular, você precisará alocar um segmento de memória compartilhada entre todos os passageiros e deverá compartilhar também os semáforos. Para tanto você deverá também descobrir um jeito de comunicar o identificador destes recursos a todos os processos passageiros sem utilizar um processo coordenador!

Inicialmente, o primeiro passageiro que chegar deve criar todos os recursos necessários para que futuros passageiros possam interagir com o sistema de travessia. Adicionalmente, ele também decidirá de forma aleatória a margem em que o barco se encontra. Quando qualquer passageiro terminar sua viagem ou mesmo desistir dela, ele deve verificar o estado da simulação. Caso não exista nenhuma atividade (ninguém atravessando ou esperando pelo barco, ou mesmo desembarcando), ele deve imprimir este fato, se identificar, remover quaisquer recursos alocados previamente e sair do sistema sem deixar rastros.

Instruções de entrega

Você deve entregar todos os arquivos fonte (em C ou C++) e quaisquer outros arquivos adicionais (e.g. Makefile). Crie um arquivo compactado `.zip, .tar.gz, .tgz` ou `.bz2` contendo todos os arquivos pertencentes ao seu EP. Não inclua código executável, somente código fonte (.h, .c, .S) e arquivos textos contendo quaisquer descrições suas. Identifique o seu EP de acordo com o exemplo abaixo:

```
EP2-Marcel-Alexandre.zip
```

Não esqueça de adicionar um arquivo `LEIAME.TXT` contendo os nomes dos participantes da equipe (máximo 2 pessoas), números USP e demais instruções necessárias para compilar o seu programa e entender o seu EP. Você é responsável pela clareza destas instruções. Entregue o seu EP eletronicamente no site da disciplina até a data final. Envie somente um EP para a sua equipe.