Módulo 2 - Laboratório 4
Sincronização condicional com variáveis de condição

Computação Concorrente (MAB-117)
Prof. Silvana Rossetto

[1]Instituto de Computação/UFRJ

Introdução

O objetivo deste Laboratório é introduzir o uso de soluções de sincronização por condição, usando variáveis de condição oferecidas pela biblioteca Pthreads. Para cada atividade, siga o roteiro proposto e responda às questões colocadas.

Atividade 1

Objetivo: Introduzir o uso de *variáveis de condição* provido pela biblioteca Pthreads.

Roteiro:

1. Abra o arquivo **hellobye.c** e compreenda como a aplicação funciona (*acompanhe a vídeo-aula4 com a explanação da professora*).
2. Execute o programa **várias vezes**. O log de execução impresso na tela foi sempre o esperado? A condição lógica da aplicação foi atendida em todas as execuções?
3. Agora altere o número de threads A para 1. O que vai ocorrer na execução? O programa vai terminar? Por que?
4. Altere o número de threads A de volta para 2.
5. Agora altere o número de threads B para 2 e faça as correções necessárias no código para que a aplicação continue funcionando.

Atividade 2

Objetivo: Exemplificar o uso de *variáveis de condição* provido pela biblioteca Pthreads.

Roteiro:

1. Abra o arquivo **byehello.c** e compreenda como a aplicação funciona (*acompanhe a vídeo-aula5 com a explanação da professora*).
2. Execute o programa **várias vezes**. O log de execução impresso na tela foi sempre o esperado? A condição lógica da aplicação foi atendida em todas as execuções?

Atividade 3

Objetivo: Explorar características do funcionamento das operações sobre variáveis de condição em Pthreads.

Roteiro:

1. Abra o arquivo **printX.c** e compreenda como a aplicação funciona (*acompanhe a vídeo-aula6 com a explanação da professora*).
2. Execute o programa **várias vezes**. O log de execução impresso na tela foi sempre correto? A condição lógica da aplicação foi atendida em todas as execuções?
3. Podemos substituir a linha 50 pela linha 51? Justifique.
4. Comente a linha 50 e descomente a linha 51, e execute novamente a aplicação **várias vezes**. O log de execução impresso na tela foi sempre correto? A condição lógica da aplicação foi atendida em todas as execuções?

Atividade 4

Objetivo: Projetar e implementar um programa concorrente onde a ordem de execução das threads é controlada no programa.

Roteiro: Implemente um programa com 4 threads:

- A thread 1 imprime a frase "Fique a vontade."
- A thread 2 imprime a frase "Seja bem-vindo!"
- A thread 3 imprime a frase "Volte sempre!".
- A thread 4 imprime a frase "Sente-se por favor."

A ordem de impressão das mensagens deverá ser:

- A thread 2 deve sempre imprimir sua mensagem antes das threads 1 e 4.
- A ordem em que as threads 1 e 4 imprimem suas mensagens não importa, mas ambas devem sempre imprimir suas mensagens antes da thread 3.

  As saídas possíveis para o programa são:

```
(1)
Seja bem-vindo!
Sente-se por favor.
Fique a vontade.
Volte sempre!

(2)
Seja bem-vindo!
Fique a vontade.
Sente-se por favor.
Volte sempre!
```

Disponibilize o código implementado na **Atividade 4** em um ambiente de acesso remoto (GitHub ou GitLab). **Use o formulário de entrega desse laboratório para enviar o link do repositório do código implementado e listar dúvidas ou dificuldades encontradas.**