Exercícios práticos
Threads e Mutex

INE5410 - Programação Concorrente
Prof. Márcio Castro

1 Dicas úteis

Para realizar os exercícios a seguir, você necessitará de:

- Um editor de texto para escrever o seu código: escolha o editor de sua preferência (vim, emacs, nano, pico, gedit, ...)
- Um compilador: usaremos o GCC (GNU C Compiler).
- Um terminal: para compilar e executar o seu programa.

A sintaxe para compilar um programa em C é a seguinte:

```
$ gcc -o <nome_arquivo_binario> <nome_arquivo_contendo_o_código>
```

Por exemplo: para criar um programa chamado meu_programa a partir de um código em C chamado meu_programa.c faça:

```
$ gcc -o meu_programa meu_programa.c
```

Se tudo ocorrer bem, ao final da compilação será gerado um arquivo binário chamado meu_programa. Para executá-lo, digite:

```
$ ./meu_programa
```

Você deverá incluir as seguintes bibliotecas nos seus códigos:

```
#include <unistd.h>
#include <sys/types.h>
#include <stdio.h>
#include <pthread.h>
```

2 Exercícios

Agora, vamos exercitar a criação de threads no Linux utilizando a API POSIX Threads (PThreads). Para compilar um programa com a biblioteca PThreads você deverá adicionar o parâmetro -lpthread no momento da compilação do seu programa:

```
$ gcc -std=c11 -o meu_programa meu_programa.c -lpthread
```

Exercício 1 Escreva um programa em C que cria uma worker thread usando PThreads. A thread criada deverá imprimir na tela a frase "Nova thread criada. TID = XX!", onde XX é o identificador da thread. A main thread deverá aguardar que a thread criada imprima a frase na tela antes de terminar. Dica: use a função pthread_self() para retornar o ID da thread.

Exercício 2 Transforme o código do exercício anterior em um código genérico que cria n threads. Da mesma forma que o exercício anterior, a main thread deverá aguardar a finalização de todas as worker threads antes de terminar. O número de threads deverá ser definido no código em uma constante chamada MAX_THREADS. Por exemplo, deverão ser criadas 5 threads se a constante for definida da seguinte forma: #define MAX_THREADS 5.

Exercício 3 Tendo como base a solução para o exercício anterior, crie uma variável global do tipo inteiro inicialmente contendo o valor 0 (int contador_global = 0;). Modifique a função executada pelas threads para que cada thread realize **100 operações de incremento nesta variável global** (contador_global++) **sem a utilização de mutex**. Após o término da execução das threads, a main thread deverá imprimir o valor armazenado na variável contador_global. **Em uma execução correta, o valor impresso deverá ser igual a 100 vezes o número de threads criadas**, ou seja contador_global = 100 * MAX_THREADS. Execute várias vezes este programa com 2, 4, 8, 32 e 128 threads e observe o valor impresso a cada vez. O que acontece?

3 Mutex

Exercício 4 Modifique a sua solução do exercício anterior para agora proteger a região crítica (contador_global++) com um mutex. **Somente o incremento da variável deverá ser protegido com o mutex**. Execute várias vezes este programa com 2, 4, 8, 32 e 128 threads e observe o valor impresso a cada vez. O que aconteceu?