UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA
DIM0124 – PROGRAMAÇÃO CONCORRENTE

Trabalho Prático

Mecanismos de Sincronização

Objetivo

O objetivo deste trabalho é estimular o projeto, implementação e avaliação de soluções para problemas por meio de programação concorrente, em especial colocando em prática os conceitos e mecanismos de sincronização de *threads*.

1 O Problema: Uma Lista Simplesmente Encadeada Concorrente

Considere uma lista simplesmente encadeada cujo acesso é compartilhado por três tipos de *threads*: o tipo B realiza operações de busca sobre a lista, o tipo I realiza operações de inserção de itens no final da lista e o tipo R realiza operações de remoção de itens a partir de qualquer posição da lista. *Threads* do tipo B meramente realizam operações de leitura sobre a lista e, portanto, podem ser executadas de forma simultânea com as outras. Por sua vez, as operações de inserção realizadas pelas *threads* do tipo I devem ser mutuamente exclusivas a fim de impedir que duas *threads* estejam inserindo itens no final da lista ao mesmo tempo. Por fim, no máximo uma *thread* do tipo R pode acessar a lista por vez para realizar remoção de itens e essa operação deve ser mutuamente exclusiva com relação às demais (busca e inserção).

Projete e implemente uma solução concorrente para o problema que satisfaça esses três tipos de exclusão mútua entre as operações realizadas pelas threads dos tipos B, I e R sobre a lista simplesmente encadeada compartilhada. Por questões de simplicidade, considere que a lista armazena apenas valores do tipo inteiro. A cada (tentativa de) operação, o programa deverá exibir na saída padrão a operação que está sendo realizada.

2 Tarefas

A tarefa central a ser realizada neste trabalho consiste em projetar e implementar uma solução concorrente para o problema anteriormente descrito utilizando conceitos e técnicas de programação concorrente, incluindo a criação, execução e sincronização de *threads* independentes e concorrentes. A solução poderá ser implementada utilizando facilidades providas pelas linguagens de programação C++, Java **ou** Python utilizando um dos mecanismos para sincronização de *threads* concorrentes estudados.

O desenvolvimento da solução deve de antemão visar pela busca de desenvolvimento de *software* de qualidade, isto é, funcionando correta e eficientemente, exaustivamente testado, bem documentado e com tratamento adequado de eventuais exceções. Mais ainda, a implementação deverá garantir corretude do programa com relação a concorrência e aplicar de forma adequada os conceitos e mecanismos de sincronização.

Além da implementação da solução, deverá ser elaborado um relatório escrito simples descrevendo, pelo menos:

- como a solução foi projetada;
- a lógica de sincronização utilizada, em termos dos mecanismos empregados e como ela é feita entre os fluxos de execução do programa;
- como é garantida a corretude da solução com relação a concorrência, em termos de seu resultado, garantia de exclusão mútua e ausência de condições de *deadlock* e *starvation*;
- eventuais dificuldades encontradas durante o desenvolvimento, e;
- instruções para compilação e execução do programa.

3 Autoria e política de colaboração

O trabalho poderá ser feito **individualmente ou em equipe composta por no máximo dois estudantes**, sendo que, neste último caso, é importante, dentro do possível, dividir as tarefas igualmente entre os integrantes da equipe. O trabalho em cooperação entre estudantes da turma é estimulado, sendo aceitável a discussão de ideias e estratégias. Contudo, tal interação não deve ser entendida como permissão para utilização de (parte de) código fonte de outras equipes, o que pode caracterizar situação de plágio. Trabalhos copiados em todo ou em parte de outras equipes ou da Internet serão sumariamente rejeitados e receberão nota zero.

4 Entrega

Todos os códigos fonte resultantes da implementação deste trabalho, sem erros de compilação e devidamente testados e documentados, deverão ser submetidos como um único arquivo compactado no formato `.zip` **até as 23h59 do dia 15 de janeiro de 2022** através da opção *Tarefas* na Turma Virtual do SIGAA. Juntamente com os códigos fonte, o arquivo compactado deverá também conter o relatório escrito, preferencialmente em formato *Adobe Portable Format* (PDF). Se for o caso, é possível fornecer, no campo *Comentários* do formulário eletrônico de submissão da tarefa, o endereço de um repositório remoto destinado ao controle de versões, porém esta opção não exclui a necessidade de submissão dos arquivos via SIGAA.

5 Avaliação

A avaliação deste trabalho será feita principalmente sobre os seguintes critérios: (i) utilização correta dos conceitos de *threads* estudados; (ii) a corretude da execução das soluções implementadas, tanto com

relação a funcionalidades quanto a concorrência; (iii) a aplicação de boas práticas de programação, incluindo legibilidade, organização e documentação de código fonte, e; (iv) qualidade do relatório produzido. Este trabalho possuirá nota máxima de 10,0 (dez) pontos.