Universidade Federal do Rio Grande do Norte
Instituto Metrópole Digital
IMD0036 – Sistemas Operacionais

Exercícios

Sincronização de threads

Prof. Gustavo Girão
girao@imd.ufrn.br

Instruções

- Consultas são permitidas exceto:
  - Consulta ao colega
  - Cópia integral de códigos excetuando os fornecidos pelo professor via SIGAA
- Cada exercício vale 0,5 ponto
  - TOTAL: 1,5 ponto

Exercício 1

- Dadas duas matrizes quadradas A e B de dimensão M, um algoritmo de multiplicação de marizes que gera a matriz resultante C é dado por:

```
for(linha=0;linha<M;linha++)
        for(coluna=0;coluna<M;coluna++){
                somaprod=0;
                for(i=0;i<M;i++)
                        somaprod+=A[linha][i]*B[i][coluna];
                C[linha][coluna]=somaprod;
        }
```

- Crie um programa em C que receba como parâmetro (utilizando argc, argv) o valor de *M* e que:
  - Crie ***M*** threads.
  - Cada thread **x** (onde **0** <= **x** < ***M***):
    - Realiza o calculo para a geração de UMA linha da matriz e;
    - Garanta exclusão mútua acessando a matriz B;

Exercício 1 - exemplo

- Se *n*=3:

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][0]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

Cálculo:
C[0][0] = (A[0][0] * B[0][0]) + (A[0][1] * B[1][0]) + (A[0][2]*B[2][0])

Feito pela **Thread 0** pois é o cálculo da **linha 0** da matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][0]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

Cálculo:
C[0][0] = (3 * 2) + (8 * 8) + (5*1)

Feito pela **Thread 0** pois é o cálculo da **linha 0** da matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][0]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

Cálculo:
C[0][0] = (3 * 2) + (8 * 8) + (5*1)
C[0][0] = 6 + 64 + 5

Feito pela **Thread 0** pois é o cálculo da **linha 0** da matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][0]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

75		

Cálculo:
C[0][0] = (3 * 2) + (8 * 8) + (5*1)
C[0][0] = 6 + 64 + 5
C[0][0] = 75

Feito pela **Thread 0** pois é o cálculo da **linha 0** da matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][1]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

75		

Cálculo:
C[0][1] = (A[0][0] * B[0][1]) + (A[0][1] * B[1][1]) + (A[0][2]*B[2][1])

ainda feito pela **Thread 0** pois **ainda** é o cálculo da **linha 0** da matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][1]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

75		

Cálculo:
C[0][1] = (3 * 1) + (8 * 5) + (5*0)

ainda feito pela **Thread 0** pois **ainda** é o cálculo da **linha 0** da matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][1]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

75		

Cálculo:
C[0][1] = (3 * 1) + (8 * 5) + (5*0)
C[0][1] = 3 + 40 + 0

ainda feito pela **Thread 0** pois **ainda** é o cálculo da **linha 0** da matriz C

Exercício 1 - exemplo

- Se *n*=3:
- Para gerar o elemento C[0][1]

Matriz A

3	8	5
0	12	1
4	4	2

Matriz B

2	1	7
8	5	11
1	0	6

Matriz C

75	43	

Cálculo:
C[0][1] = (3 * 1) + (8 * 5) + (5*0)
C[0][1] = 3 + 40 + 0
C[0][1] = 43

ainda feito pela **Thread 0** pois **ainda** é o cálculo da **linha 0** da matriz C

Exercício 2

- Faça um código C que resolva o problema do produtor, consumidor assumindo que:
- O tamanho do buffer é passado por parâmetro (argc, argv)
- Existem 5 threads:
  - 3 consumidoras e 2 produtoras
  - Garanta exclusão mútua no acesso ao buffer, **independentemente se a thread é consumidora ou produtora** e;
  - Faça com que qualquer thread produtora que tente escrever no buffer quando cheio durma e acorde quando o buffer tiver pelo menos uma posição livre e;
  - Qualquer thread consumidora que tente ler no buffer quando vazio durma e acorde quando o buffer tiver pelo menos um elemento dentro dele.

Exercício 3

- Implemente uma solução para o problema dos **Leitores-Escritores**. Sua solução implementada em C deve receber 2 parâmetros (por meio de argc, argv): O número de leitores e o número de gravadores. Este problema simula a tentativa de acesso simultâneo de duas solicitações distintas a uma base de dados: para ler ou para escrever. Entretanto, podem haver diversos leitores e diversos gravadores.

Exercício 3

- Garanta a exclusão mútua nesse caso, considerando:
  - A "base de dados" é simplesmente representada por uma variável inteira. Leitores lêem-a, Escritores incrementam-a.
  - Quando um gravador precisa ter acesso à base de dados, ele o faz de maneira exclusiva (i.e. nenhum outro leitor ou gravador tem acesso).
  - Quando um leitor precisa ter acesso à base de dados, outros leitores podem fazê-lo ao mesmo tempo. Entretanto, caso UM escritor queira ter acesso, nenhum leitor pode fazê-lo.
  - O escritor faz chamada a uma função "escrevendo" que simplesmente espera 2 segundos (*sleep(2)*), escreve um novo valor na variável e imprime "Escritor #x: escrevendo dado y" (onde x é um identificador da thread e y é o novo valor da variável).
  - O leitor faz chamada a uma função "lendo" que simplesmente espera 1 segundo (*sleep(1)*), lê o novo valor da variável e imprime "Leitor #x: lendo dado y" (onde x é um identificador da thread e y é o valor atual da variável).