UFOP-DECOM-BCC264 Nº 02/2021

Ouro Preto, 28 de Janeiro de 2021

µTP2

1- Este µTP pode ser feito exclusivamente em C ~~ou C++~~ (apenas C, ok?), preferencialmente em Linux (pode ser Windows)
2- Usar obrigatoriamente a biblioteca pthread (C++ não tem a biblioteca pthreads).
3- É individual

Veja o seguinte programa que está na página 73 do livro (capitulo2):

```
while (TRUE) {
    while (turn !=0)            /* laço */ ;
    critical_region( );
    turn = 1;
    noncritical_region( );
}
```

(a)

```
while (TRUE) {
    while (turn !=1)            /* laço */ ;
    critical_region( );
    turn = 0;
    noncritical_region( );
}
```

(b)

Figura 2.18 Solução proposta para o problema da região crítica. (a) Processo 0. (b) Processo 1. Em ambos os casos, não deixe de observar os ponto-e-vírgulas concluindo os comandos while.

4- Você deve entender o código (explicado no texto, na aula do dia 27-01-2021, e faz parte deste µTP também o entendimento), implementá-lo em "C" usando pthreads.

5- Você deve explicar, no entregável, o que é a região critica (veja que critcial_region() e noncritical_region()) e responder se o código da região critical_region() do algoritmo "a" tem que ser o mesmo código de critical_region() de "b". O mesmo para os códigos noncritial_region() de "a" e "b". Eles tem que ser os mesmos em "a" e "b" para este algoritmo funcionar?. Explique isto de tal forma a ficar claro as suas respostas e que você entendeu (isto valerá ponto nest µTP tmb)

6- 2ª tarefa: Refaça o algoritmo de µTP1, implementando região crítica como (similar) a figura acima. Obviamente não é o mesmo programa. É a implementação das funcionalidades de ProcUP() e ProcDown() (do µTP1) usando região crítica como mostrado na figura cima.

7- Explique o que é a função `pthread_join (biblioteca pthreads)` e por que eles resolve ou não resolve a condição de corrida (race condition). Afinal. O que ela (a função `pthread_join )` faz e não usá-la o que pode acontecer? Excplique o que é *race condition*?

8- Nos entregáveis, explique se as 4 leis foram satisfeitas nos dois códigos e por que (sim ou não) que estão na página 71 do livro:

> Embora essa solução impeça as condições de disputa, isso não é suficiente para que processos paralelos cooperem correta e eficientemente usando dados compartilhados. Precisamos satisfazer quatro condições para chegar a uma boa solução:
>
> 1. Dois processos nunca podem estar simultaneamente em suas regiões críticas.
> 2. Nada pode ser afirmado sobre a velocidade ou sobre o número de CPUs.
> 3. Nenhum processo executando fora de sua região crítica pode bloquear outros processos.
> 4. Nenhum processo deve esperar eternamente para entrar em sua região crítica.

9- Entregáveis
   a. Programas:
   b. vídeo mostrando o funcionamento de, no máximo, 3 minutos e mostrando que você fez e que você entendeu. Muito importante "passar" isto para mim, no vídeo.
   c. Texto formato PDF com as tarefas solicitadas de tal forma que eu também entenda que você fez e que você fez o solicitado (inclusive ter entendido o conteúdo da aula e também dos assuntos que estão no livro) e com referências.

10- Faça o download dos entregáveis (veja acima) no moodle.

Prof. Carlos Frederico M.C. Cavalcanti
DECOM/ICEB