

© 2002-2004 Volnys Bernal 1

Sincronização e Comunicação Problemas Clássicos

Volnys Borges Bernal
volnys@lsi.usp.br
<http://www.lsi.usp.br/~volnys>



© 2002-2004 Volnys Bernal 2

Agenda

- Problemas Clássicos:
 - ❖ Produtor-consumidor
 - ❖ Leitores e escritores
 - ❖ Jantar dos filósofos
 - ❖ Barbeiro sonolento

© 2002-2004 Volnys Bernal 3

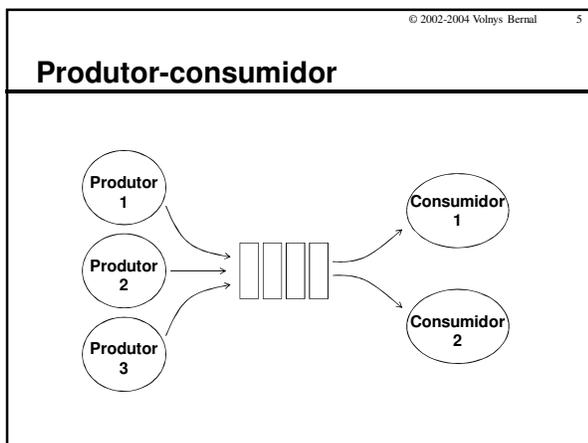
Produtor-Consumidor



© 2002-2004 Volnys Bernal 4

Produtor-consumidor

- Um sistema é composto por entidades produtoras e entidades consumidoras.
- Entidades produtoras
 - ❖ Responsáveis pela produção de itens que são armazenados em um buffer (ou em uma fila)
 - ❖ Itens produzidos podem ser consumidos por qualquer consumidor
- Entidades consumidoras
 - ❖ Consomem os itens armazenados no buffer (ou na fila)
 - ❖ Itens consumidos podem ser de qualquer produtor



© 2002-2004 Volnys Bernal 6

Produtor consumidor

- Exercício:
 - ❖ Resolva o problema produtor-consumidor supondo ambiente operacional preemptível e semáforos.

© 2002-2004 Volnys Bernal 7

Leitores e Escritores



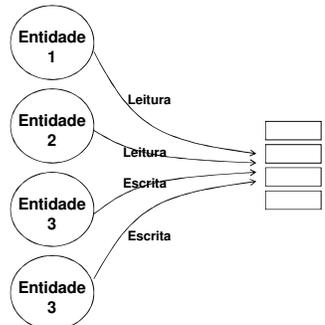
© 2002-2004 Volnys Bernal 8

Leitores e escritores

- ❑ Um sistema com uma base de dados é acessado simultaneamente por diversas entidades. Estas entidades realizam dois tipos de operações:
 - ❖ Leitura
 - ❖ Escrita
- ❑ Neste sistema é aceitável a existência de diversas entidades lendo a base de dados.
- ❑ Porém, se um processo necessita escrever na base, nenhuma outra entidade pode estar realizando acesso à base.

© 2002-2004 Volnys Bernal 9

Leitores e escritores



© 2002-2004 Volnys Bernal 10

Leitores e escritores

- ❑ Exercício:
 - ❖ Resolva o problema dos leitores e escritores supondo:
 - Ambiente operacional (preemptível/não preemptível): _____
 - Processadores (monoprocessador/multiprocessador): _____

```

nleitura = 0
nescrita = 0

Leitura()
{
}

Escrita()
{
}
    
```

© 2002-2004 Volnys Bernal 11

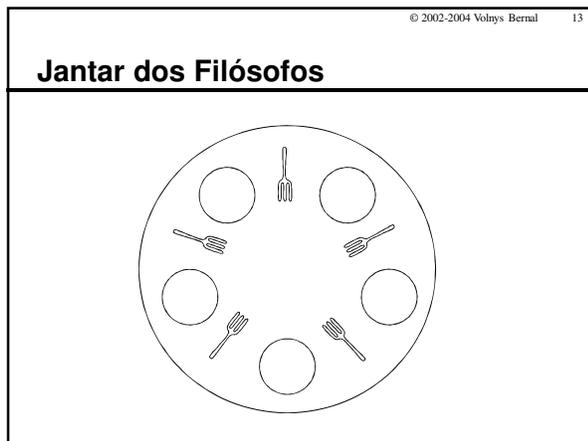
Jantar dos Filósofos



© 2002-2004 Volnys Bernal 12

Jantar dos Filósofos

- ❑ Cinco filósofos estão sentados ao redor de uma mesa circular para o jantar.
- ❑ Cada filósofo possui um prato para comer espaguete.
- ❑ Como o espaguete é muito escorregadio, é necessário a utilização de dois garfos.
- ❑ Entre cada par de pratos existe um garfo.



© 2002-2004 Volnys Bernal 14

Jantar dos Filósofos

- ❑ A vida do filósofo consiste na alternância de períodos de alimentação e reflexão.
- ❑ Quando um filósofo fica com fome, ele tenta pegar os garfos a sua volta (garfos a sua esquerda e direita), em qualquer ordem, um de cada vez.
- ❑ Se o filósofo conseguir pegar os dois garfos ele inicia seu período de alimentação. Após algum tempo ele devolve os garfos a sua posição original e retorna ao período de reflexão

© 2002-2004 Volnys Bernal 15

Jantar dos Filósofos

- ❑ Pergunta:
 - ❖ É possível escrever um programa para possibilitar que cada filósofo possa realizar suas atividades sem problemas?
- ❑ Problemas?
 - ❖ Se todos os filósofos resolverem comer ao mesmo tempo?

© 2002-2004 Volnys Bernal 16

Jantar dos filósofos

- ❑ Exercício:
 - ❖ Resolva o problema do jantar dos filósofos supondo:
 - Ambiente operacional (preemptive/não preemptive): _____
 - Processadores (monoprocessador/multiprocessador): _____

```

Filosofo ()
{

}

```

© 2002-2004 Volnys Bernal 17

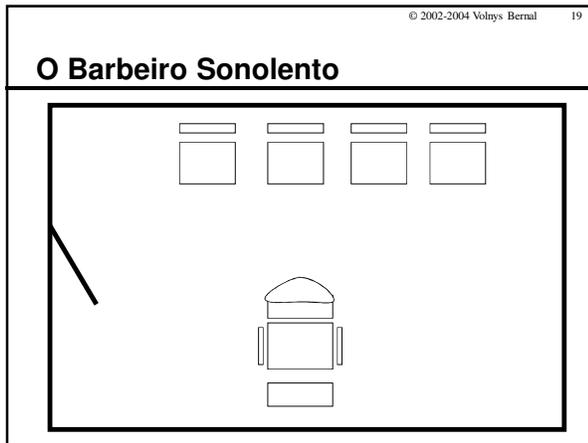
Problemas Clássicos:

O Barbeiro Sonolento

© 2002-2004 Volnys Bernal 18

O Barbeiro Sonolento

- ❑ Uma barbearia possui:
 - ❖ 1 barbeiro
 - ❖ 1 cadeira de barbeiro
 - ❖ N cadeira para espera de clientes
- ❑ Se, em um determinado momento, não houverem clientes para serem atendidos, o barbeiro dorme.
- ❑ Quando um cliente chega, ele acorda e atende o cliente.
- ❑ Quando um cliente chega e o barbeiro estiver atendendo um cliente, ele aguarda sua vez sentado na cadeira de espera.
- ❑ Quando um cliente chega e não existem cadeiras de espera disponíveis, o cliente vai embora.



© 2002-2004 Volnys Bernal 20

Barbeiro sonolento

□ Exercício:

- ✦ Resolva o problema dos barbeiro sonolento supondo:
 - Ambiente operacional (preemptivo/não preemptivo): _____
 - Processadores (monoprocessador/multiprocessador): _____

```
nclientes = 0

Barbeiro ()
{

}

Cliente ()
{

}
```