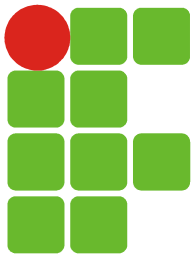


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
BAHIA
Campus Salvador

# Comunicação entre processos

# Comunicação entre processos (-- motivação --)

- Processos em execução no sistema operacional podem ser:
  - Independentes:
    - Quando não podem ser afetados pela execução de outro processo
  - Cooperantes
    - Quando podem ser afetados pela execução de outro processo
- Já sabendo que compartilhamento causa problemas, é mais fácil criar processos independentes!!

# Comunicação entre processos (-- motivação --)

- Entretanto, é extremamente desejável criar um ambiente com processos cooperantes!

- Porque?
  - Compartilhamento de informações
  - Aumento da velocidade de computação
  - Modularidade
  - Dar suporte a execução de várias tarefas

- Processos cooperantes requerem comunicação entre processos (*Interprocess communication – IPC*)

# Comunicação entre processos (-- definição --)

- Mecanismo que permite aos processos trocarem dados ou informações.
- Comunicação entre processos não usa interrupção!
- Frequentemente é feita de duas formas:
  - Troca de mensagens
  - Compartilhamento de memória

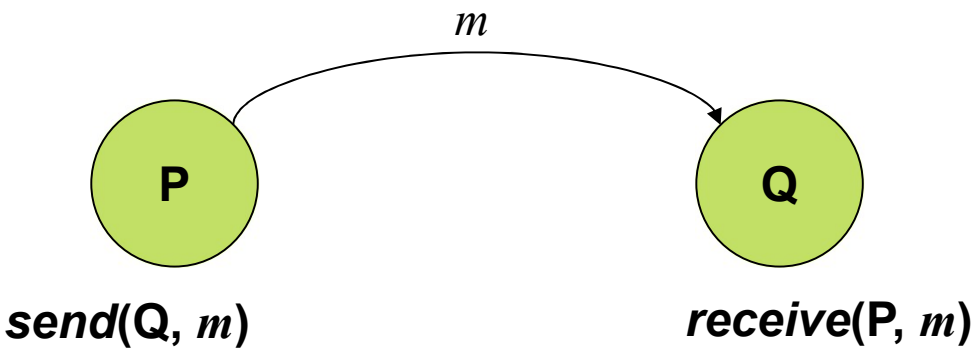
# Comunicação entre processos (-- troca de mensagens --)

- Se pensarmos numa arquitetura centralizada, os processos estão na mesma máquina.
  - Diferentes processos têm acesso aos mesmos recursos.
- O que acontece se a arquitetura do sistema for distribuída? (um *chat*, por exemplo)
- Como os processos podem se comunicar?

# Comunicação entre processos (-- troca de mensagens --)

- Processos podem se comunicar por troca de mensagens.
  - Frequentemente quando estão em diferentes máquinas e precisam compartilhar dados
- A troca de mensagens é feita baseada em duas ***primitivas***:
  - *send()*
  - *receive()*
- Mensagens podem ter tamanho fixo ou variável
- Se dois processos precisam se comunicar, deve haver um *link* entre eles.

# Comunicação entre processos
# (-- troca de mensagens --)

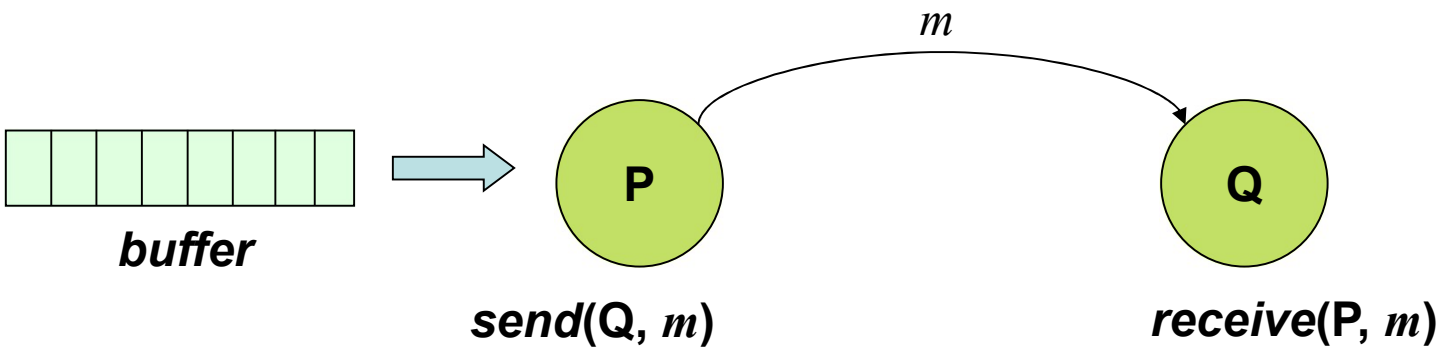


# Comunicação entre processos (-- troca de mensagens --)

- Troca de mensagens por sincronização:
  - Blocking send: processo que envia a mensagem fica bloqueado até a confirmação do recebimento
  - Nonblocking send: processo envia a mensagem e vai executar a próxima instrução
  - Blocking receive: receptor fica bloqueado até que a mensagem esteja disponível
  - Nonbloking receive: o receptor devolve uma mensagem válida ou nula.

# Comunicação entre processos (-- troca de mensagens --)

- Troca de mensagens por bufferização:
  - *Zero capacity*
  - *Bouded-capacity*
  - *Unbouded-capacity*



# Comunicação entre processos (-- compartilhamento de memória --)

- Processos devem definir uma área de memória que será compartilhada;
- Por padrão o sistema operacional não permite que um processo acesse outro processo!
- Como resolver???

# Comunicação entre processos (-- compartilhamento de memória --)

- Caso dois processos desejem compartilhar memória, ambos precisam assumir as consequências de não considerar as restrições do sistema operacional

- TODA COMUNICAÇÃO ENTRE PROCESSOS PRECISA DA ARBITRAGEM DO SISTEMA OPERACIONAL

# Comunicação entre processos (-- compartilhamento de memória --)

- Processos trocam informações através de leituras e escritas numa área compartilhada;
- O sistema operacional não controla esta operação!
- O que os processos precisam garantir??

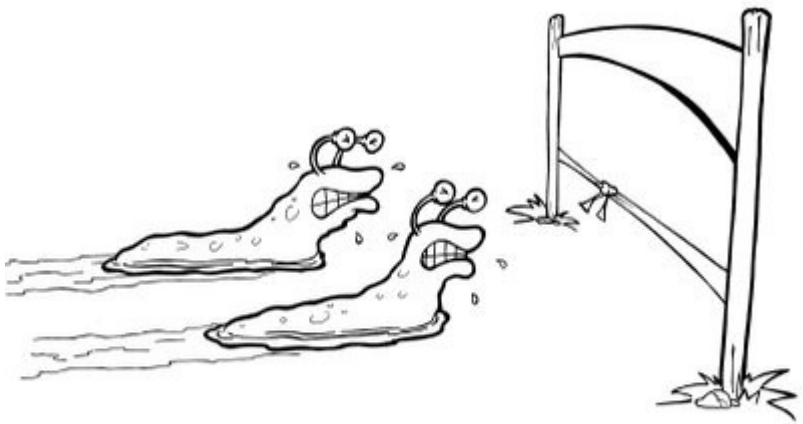
# Para pensar um pouco...

*O que acontece quando dois processos querem escrever na mesma área de memória no mesmo instante?*

# Comunicação entre processos (-- *Race condition* --)

- Em alguns sistemas operacionais, processos cooperantes frequentemente compartilham algum dispositivo de armazenamento.
  - Arquivos
  - Memória
  - Disco

# Comunicação entre processos (-- *Race condition* --)



- Dois processos podem tentar ler ou escrever dados num espaço compartilhado, e o resultado final depende de quem está executando naquele momento.

# Comunicação entre processos (-- *Race condition:* exemplo ilustrativo --)

- Um exemplo ilustrativo:
  - Suponha duas *threads*, que alteram o valor da variável $x$

$T_1$: x := x + 1
$T_2$: x := x + 2

Considere x = 2

$T_1$ ➔ $T_2$ : x = 5
$T_2$ ➔ $T_1$ : x = 5

$T_1$: x := x + 1
$T_2$: x := x * 2

Considere x = 2

$T_1$ ➔ $T_2$ : x = 6
$T_2$ ➔ $T_1$ : x = 5