Programação Concorrente

Odorico Machado Mendizabal

Universidade Federal de Santa Catarina – UFSC
Departamento de Informática e Estatística – INE

Processos

Nesta e nas próximas aulas

- Identificar os componentes de um processo e ilustrar como eles são representados e gerenciados por um SO
- Descrever como processos são criados e terminados em um SO (ciclo de vida de um processo)
- Desenvolvimento de programas usando chamadas ao sistema adequadas
- Descrever e comparar mecanismos de comunicação entre processos

Processo é uma instância de um programa

Um processo é um “programa em execução”

Programa

Código executável (algoritmo)

Ex.: Navegador Web, editor de texto, visualizador de imagem

Entidade passiva

Processo

É uma instância de um programa carregado na memória
Faz uso da CPU para execução de suas instruções
Faz uso de memória para armazenamento de dados
Faz uso de dispositivos de E/S

Ex.: Um documento em edição, filme sendo exibido por um visualizador de vídeo, etc.

Entidade ativa

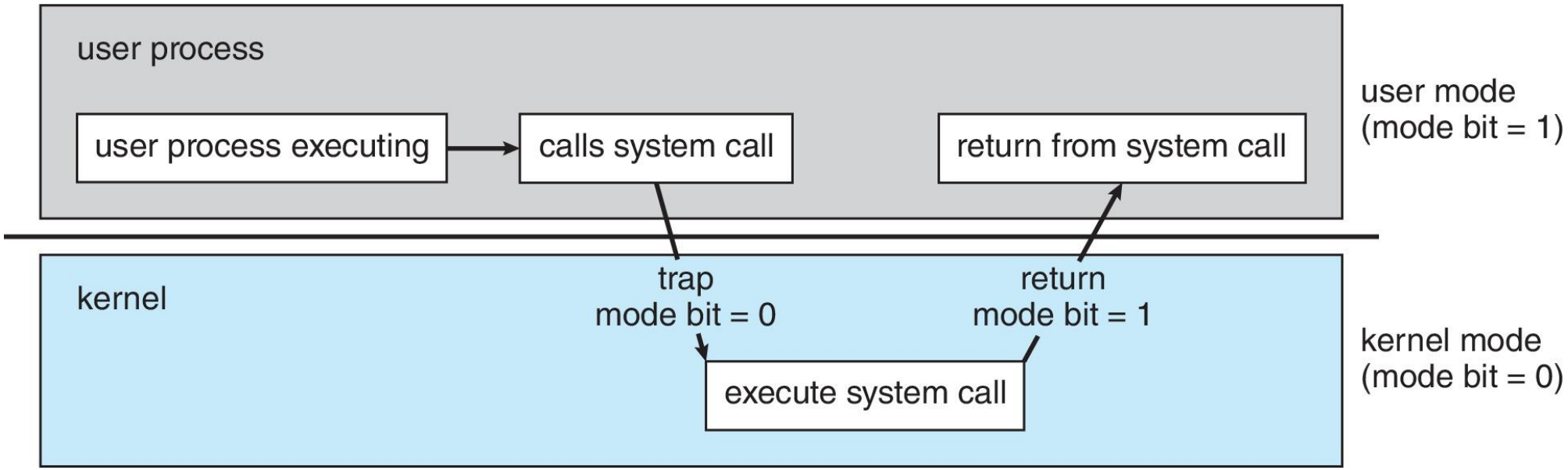
Conceitos fundamentais

- Processo
  - Uma instância de um programa em execução
  - Consiste de um espaço de endereço e uma ou mais *threads* de controle
- Espaço de endereçamento
  - Processos executam em espaço de endereços independente da memória física da máquina
- Proteção
  - O SO e HW são protegidos de programas do usuário e programas do usuário são isolados entre si através da tradução de espaço de endereço virtual para a espaço de endereçamento física
- Modo privilegiado / Modo usuário
  - Muitos projetos permitem que acesso ao HW e estruturas internas ocorra apenas no modo privilegiado (*kernel mode*)

Visão de um processo (execução dual)

Modo de execução dual: modo usuário e modo núcleo (*kernel*)

Transições do modo usuário para o modo núcleo ocorrem devido à:
- *Software*:
  - TRAP (chamada ao sistema)
  - Exceção
- Interrupções
  - Acesso à dispositivos de E/S



Espaço de endereçamento de um processo

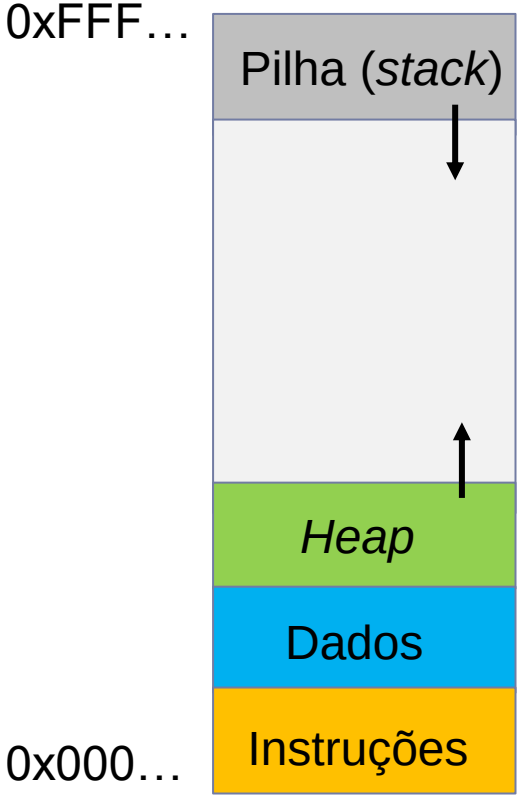
Programa

Código executável, mas é uma entidade passiva.
Instanciado pelo SO na forma de um processo

Espaço de endereçamento de um processo

Programa

Código executável, mas é uma entidade passiva.
Instanciado pelo SO na forma de um processo



O processo é carregado na memória contendo segmentos distintos para:

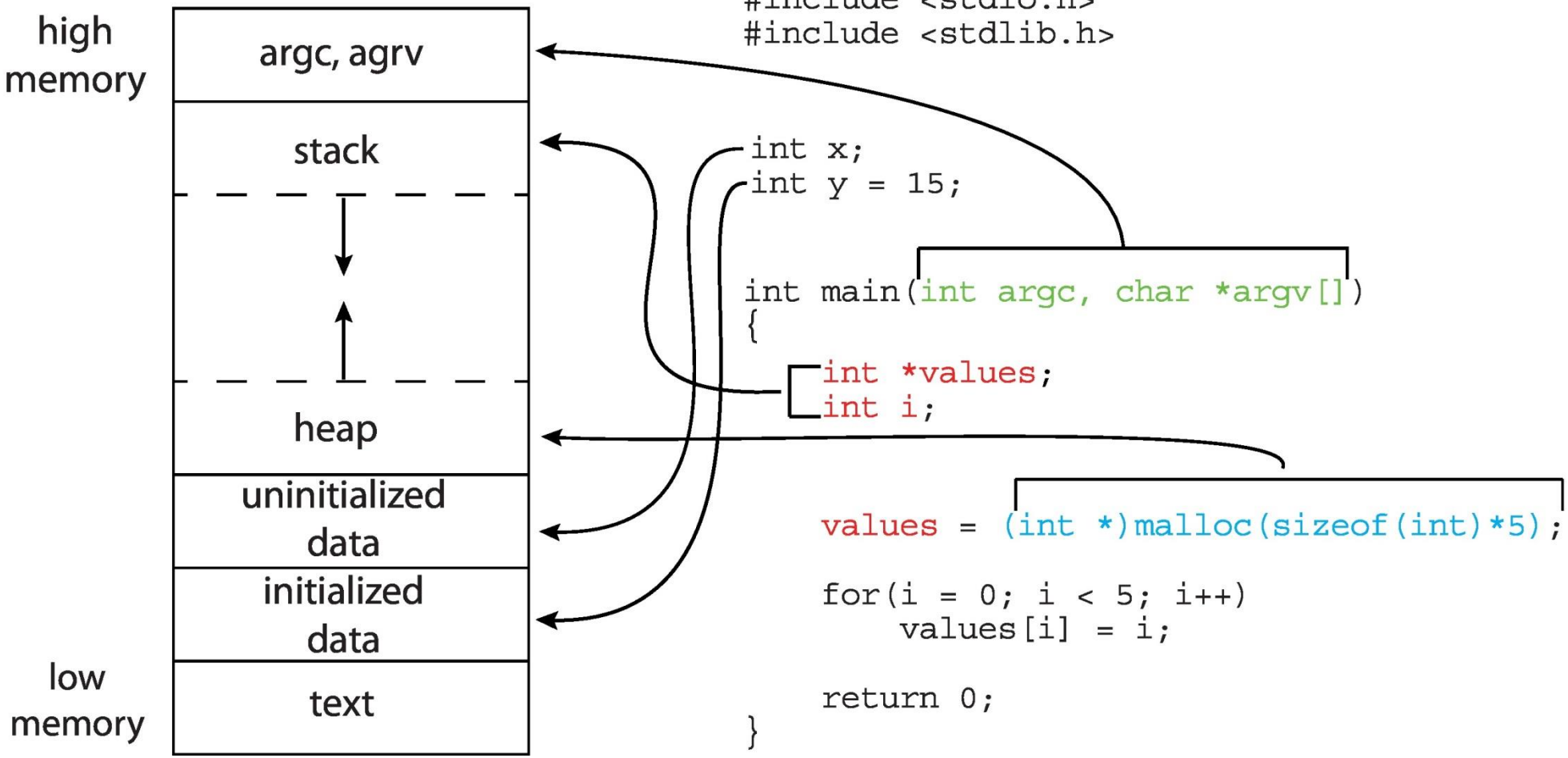
Instruções (texto): contém o código do programa

Dados: contém variáveis globais (declaradas e inicializadas estaticamente no programa)

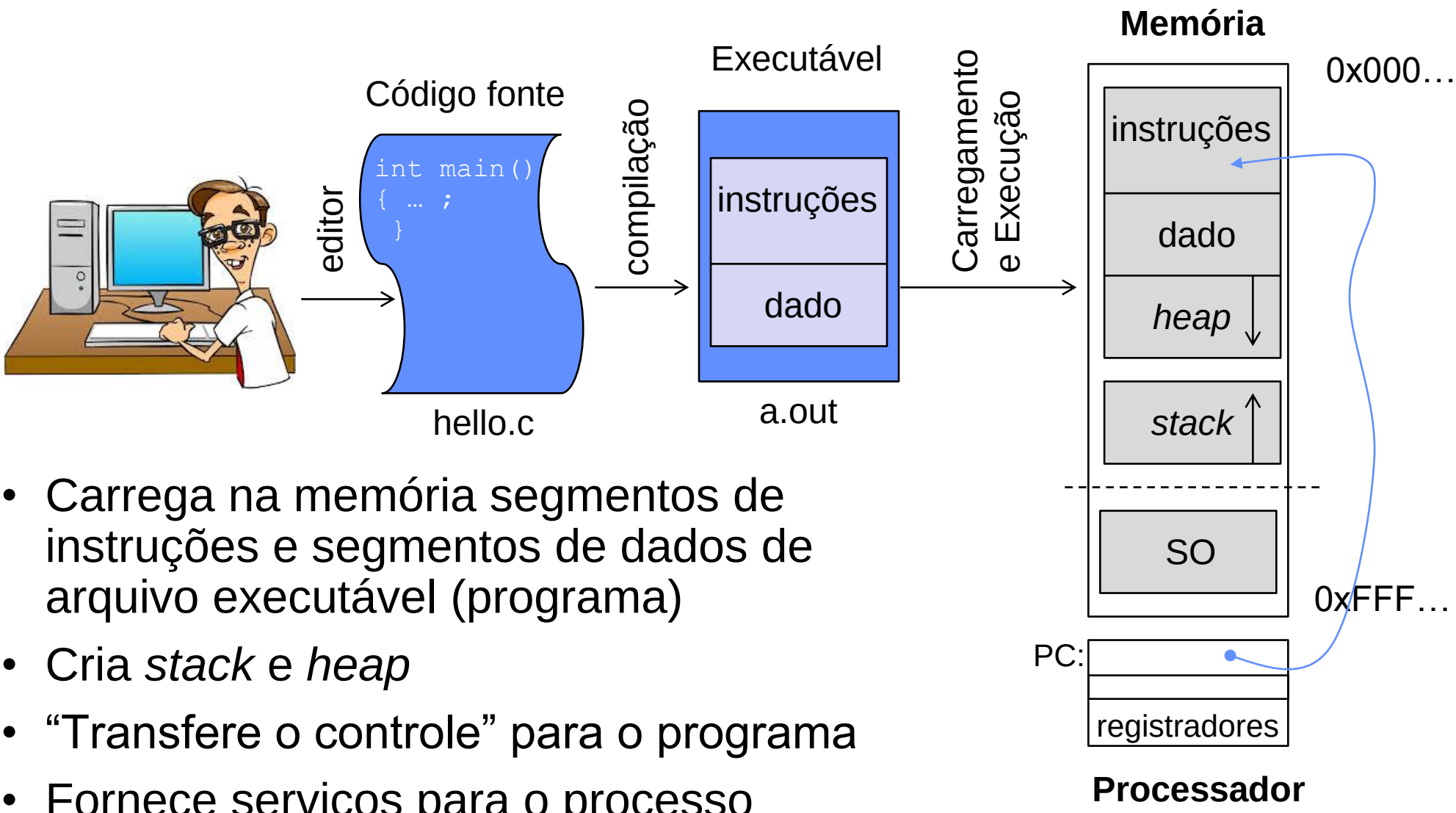
Heap: memória alocada dinamicamente, em tempo de execução

Pilha: contém dados temporários:
- variáveis locais: escopo existente apenas durante a execução de um procedimento
- endereços de retorno

Foto da memória de um programa em C



SO e a execução de programas (visão geral)



- Carrega na memória segmentos de instruções e segmentos de dados de arquivo executável (programa)
- Cria *stack* e *heap*
- “Transfere o controle” para o programa
- Fornece serviços para o processo
- Proteção ao processo e ao SO

Processos e Usuários

- Cada processo é associado ao usuário que o criou
  - Processos herdam as permissões do usuário a ele associado
    - Ex. Podem alterar arquivos que o usuário tem acesso e serão restrição de acesso à arquivos que o usuário não tem acesso
- Usuários especiais podem ser criados com finalidades específicas
  - Ex. Execução de *deamons* ou serviços do sistema
    - normalmente executam em plano de fundo (*background*)
    - Ex. `syslogd`, `sshd`, etc.
- Processos administrador têm acesso irrestrito (usuário *root*, ou administrador)

Processos: Execução

- *Foreground*
  - Processo é exibido na tela (terminal ou GUI)
  - Usuário interage com o processo
    - Fornece entradas para a execução do processo
    - Observa saídas
  - Ex. navegadores, editor de texto, compilador, etc.

- *Background*
  - Processo é executado em plano
  - Não exige interação direta com o usuário (execução autônoma)
  - Ex. sincronização de arquivos remotos, download de e-mails ou atualizações do sistema, registro de log de eventos do SO

Processos: Execução no Linux

- *Foreground*
  - Execução do processo:

`./nome_programa (binário ou shell script)`

- *Background*
  - Execução do processo em background (`&`):

`./nome_programa (binário ou shell script)` **`&`**

Para alternar um processo entre *foreground* e *background*, pode-se usar os comandos `fg` e `bg`

O comando `jobs` mostra os comandos em background

Próxima aula

- Ciclo de vida de um processo
- Criação e finalização de processos
- Gerenciamento de processos

Referências

Parte destes slides são baseadas em material de aula dos livros:

- *OLIVEIRA, Rômulo Silva de; CARISSIMI, Alexandre da Silva.; TOSCANI, Simão Sirineo. Sistemas operacionais. 4. ed. Porto Alegre: Bookman, 2010. xii, 374p. (Livros didáticos, n.11) ISBN 9788577805211*

- *SILBERSCHATZ, Abraham.; GAGME, Greg; GALVIN, Peter B. Sistemas operacionais com Java. Rio de Janeiro: Elsevier, 2008. 673 p. ISBN 9788535224061*

- *TANENBAUM, Andrew S. Sistemas operacionais modernos. 3. ed. Rio de Janeiro (RJ): Prentice-Hall do Brasil, 2010. xiii, 653p. ISBN 9788576052371*