Sistemas Operacionais: Introdução

Wedson Almeida Filho
Samuel Xavier de Souza

Ementa

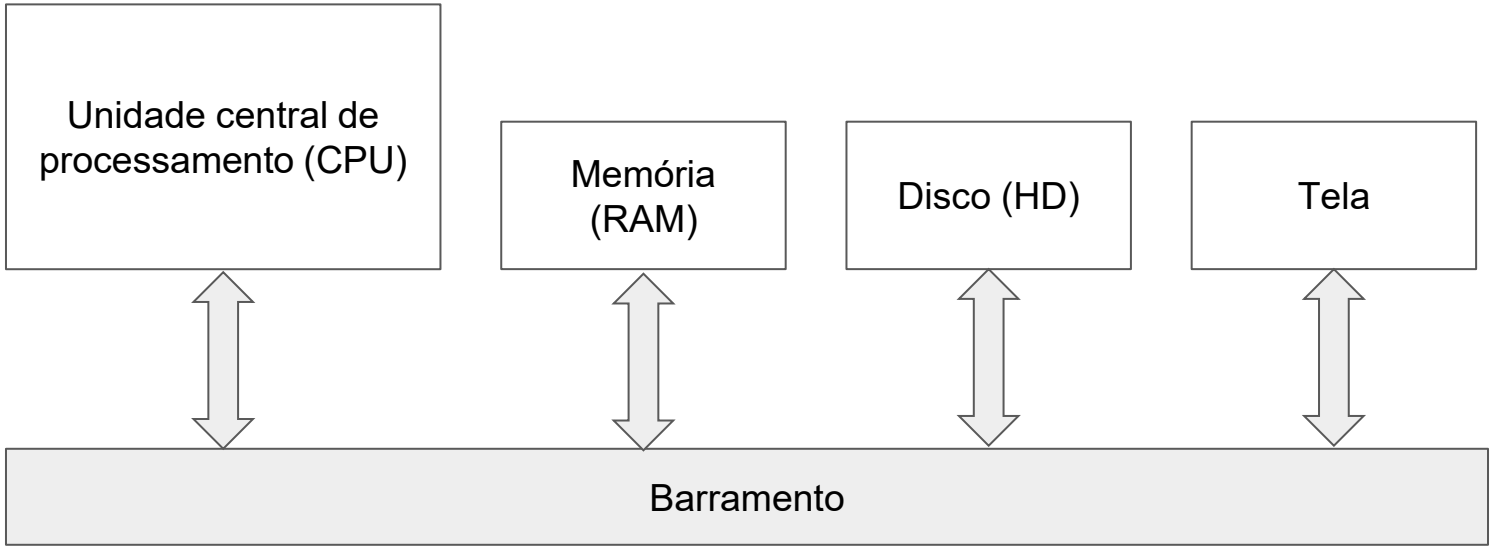
De acordo com o SIGAA:

1. Histórico e conceitos básicos.
2. Gerência de processos e programação concorrente.
3. Gerência de memória principal e auxiliar.
4. Gerência de dispositivos de entrada e saída.
5. Estudo de sistemas operacionais existentes.
6. Virtualização de Sistemas Operacionais.

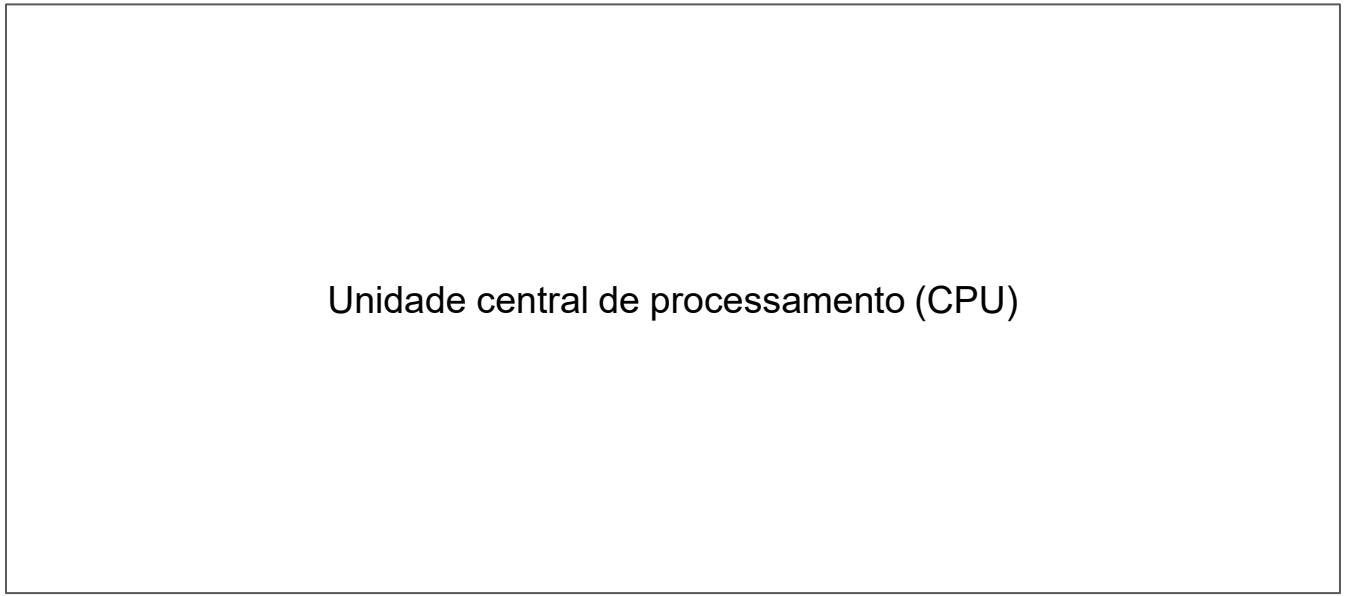
Objetivos desse tópico

- Familiarização com Conceitos Fundamentais
  - Modo Kernel vs. Modo Usuário
  - Interrupções
  - Chamadas de Sistema

Revisão: modelo de computador



Revisão: modelo de CPU



Revisão: modelo de CPU

Um laço como o seguinte:

```
loop {

        instr, instr_len = fetch_instr(pc)
        pc = pc + instr_len

        execute_instr(instr)
}
```

Revisão: modelo de CPU

Um laço como o seguinte:

```
loop {
        if irq_enabled then check_irq()

        instr, instr_len = fetch_instr(pc)
        pc = pc + instr_len

        execute_instr(instr)
}
```

Tarefa 1: Modo usuário e chamadas de sistema

Escreva um código mínimo em linguagem de montagem que contenha apenas uma instrução para atribuir um valor a um registrador qualquer da CPU. Compile-o com as flags apropriadas para evitar que o compilador adicione bibliotecas ou instruções extras. Ao executar o programa, você deverá observar uma falha de segmentação (segmentation fault).

- Explique por que essa falha ocorre.
- Corrija o código para que a falha não aconteça mais.
- Execute o programa com o comando strace antes e depois da correção e observe a diferença na sequência de chamadas de sistema.
- Explique por que foi necessário incluir uma chamada de sistema para encerrar corretamente o programa.
- Discuta brevemente quais seriam as implicações se os programas de usuário pudessem ser executados livremente, sem recorrer ao sistema operacional para acessar recursos básicos como o encerramento.

Enviar relatório em PDF anexando o código completo em sintaxe colorida.