

# Introdução à Computação

## hardware, software e sistema operacional

Alexandre Rademaker

# hardware e software I

A CPU é a unidade de processamento central, que coordena tudo que é executado em um computador.

Sequências de comandos armazenados na memória são enviados para a CPU para serem executados.

Mas um computador pode estar rodando vários programas ao mesmo tempo! E todos os programas precisam estar armazenados no computador para começar, certo?

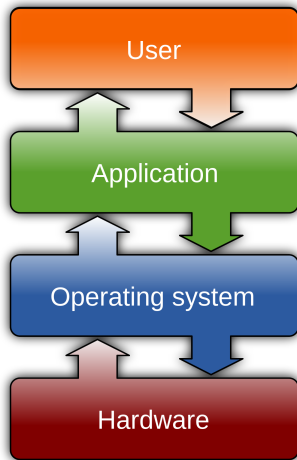
## hardware e software II

O **sistema operacional** é o programa que gerencia o computador, controlando como cada software usa cada parte do hardware do computador.

## hardware e software III

São exemplos de sistemas operacionais: Linux, Windows, MacOS, iOS, Android etc.

Eles gerenciam todos os outros programas que rodam no computador e quais dispositivos eles podem usar. Nele instalamos os programas.



## hardware e software IV

A idéia de vários programas rodando 'ao mesmo tempo' é criada pelo SO, que permite que cada programa em execução tenha uma fração de segundos da CPU por vez.

# sistema operacional I

Os sistemas operacionais modernos são acompanhados de vários programas quando instalados.

Existem os **utilitários**: gerenciador de arquivos, drivers para impressão, gerenciador de configurações, relógio, **interpretador de comandos** etc.

Existem os **aplicativos**: editores de texto, editores de imagem, navegadores etc.

Também podemos instalar novos programas desde que compatíveis com o SO.

# sistema operacional II

Os programas podem ser.

CLI (Command Line Interface) são aqueles que operam por meio de uma interface texto, onde o usuário interage digitando texto e recebendo texto como resposta, em um terminal.

GUI (Graphical User Interface) são aqueles onde o usuário interage por meio de elementos visuais, como botões, menus, janelas e ícones.

Neste curso vamos aprender a implementar programas CLI.

## sistema operacional III

Todo sistema operacional moderno (que roda em laptops e desktops) oferece dois utilitários muito importantes: gerenciador de arquivos (geralmente GUI) e interpretador de comandos (CLI).



## sistema operacional IV

O **gerenciador de arquivos** é usado para manipular os arquivos e diretórios armazenados no **sistema de arquivos** mantido pelo SO e armazenado no **hard-drive** do computador.

Exemplos: Windows Explorer, Finder do MacOS, Nautilus (Linux).

Diferente do que ocorre no iOS? Como?

# sistema operacional V

O **interpretador de comandos**, ou shell, permite ao usuário interagir com o sistema operacional por meio de uma interface de linha de comando. Ele é o responsável por ler os comandos digitados pelo usuário e executá-los, além de fornecer informações sobre os resultados.

O shell também gerencia as **variáveis de ambiente**, que são valores que podem ser acessados pelos programas em execução. Além disso, ele pode executar **scripts**, que são arquivos com uma sequência de comandos que podem ser executados em lote.

Exemplos: CMD (Windows), bash, zsh, etc.

# linha de comando I

Comandos básicos no Linux: `pwd`, `cd`, `ls`, `cp`, `mv`, `mkdir`, `rm`, `rmdir`, `touch`, `cat`, `tail`, `head`, `less`, `popd`, `pushd`, `ps`, `history`, `find`, `grep`, `df`, `diff`, `echo`, `man`, `tar` and `zip`, etc.

Tutorial sobre linha de comando.

Nomes de arquivos `/Users/ar/work/teste.txt` (completo, nome base e extensão).

Os diretórios `.` e `..`.

Opções e argumentos, `ls -l` ou `rm -rf`.

## linha de comando II

Em sistemas Unix/Linux, programas CLI produzem **códigos numéricos de retorno** para indicar como a execução ocorreu.

Outros programas ou scripts que chamam o programa em questão podem usar este código para determinar se o processo foi executado com êxito.

O código 0 indica sucesso. Outros códigos comuns: 1 para erro genérico. 2 para erro na linha de comando. 126 para alguma permissão negada. 127 para comando não encontrado e qualquer um acima de 128 para casos específicos.

Os códigos de retorno são normalmente documentados no manual do programa.

## linha de comando III

Quando executamos mais de um comando, os seguintes operadores podem ser usados:

A; B	executa A e depois B
A && B	B só será executado se A retornar zero
A    B	B só será executado se A retornar valor diferente de zero

## linha de comando IV

Entre um programa e o seu ambiente (interpretador de comandos) existem **canais de comunicação padrão**.

Estes canais são a entrada padrão (STDIN, por onde entram os dados), saída padrão (STDOUT, por onde saem os dados) e saída de erros (STDERR, usado para informar erros).

Geralmente estes fluxos de entrada e saída de um programa CLI são herdado do processo 'pai', o processo que evocou o programa, geralmente o shell.

Mas podemos **redirecionar** entrada e saídas para arquivos. Ex: `ls -l > lista.txt`.

<https://bit.ly/42303g9>

# Controle de Versão

Um controlador de versão é uma ferramenta de software que permite gerenciar as alterações feitas em arquivos de um projeto (normalmente um diretório) ao longo do tempo.

O mais famoso controlador de versão atualmente é o [Git](#), ver também <https://rogerdudler.github.io/git-guide/>.

O site GitHub <https://docs.github.com/pt> é o maior site para armazenamento de projetos opensource que usam Git.

Para saber mais: [vídeos](#) e [curso](#).