

---

# Conhecendo o computador

Semana 0 - Aula 1

---

---

# Surgimento

- Durante a segunda guerra mundial, Alan Turing desenvolve um modelo **teórico** de uma máquina capaz de seguir instruções e realizar cálculos complexos, chamada de Máquina de Turing
  - Protótipo usado para quebrar códigos de guerra alemães
  - Filme: O Jogo da Imitação
-

---

# Máquina de Turing

- Elementos:
    - Fita que pode ser lida, escrita ou movimentada
    - Tabela que determina o que escrever e pra que lado movimentar a fita, dependendo do que for lido
    - Peça que lê, escreve e movimenta a fita
  - Com esses elementos, é possível calcular **qualquer** coisa que os computadores de hoje também conseguem
-

---

# E hoje em dia?

- Máquinas poderosas
  - Múltiplas funcionalidades
    - Internet
    - Jogos
    - Trabalho (textos, planilhas, apresentações)
    - Automação de tarefas
  - Simples e intuitivo, focado no usuário “leigo”
  - Máquina que executa programas
-

---

# Analogia da cozinha

- Programa: sequência de instruções a serem executadas a fim de produzir um resultado
  - Receita: sequência de ações a serem realizadas a fim de produzir um prato
-

---

# Analogia da cozinha

- Memória: armazena programas e dados utilizados por eles
  - Armário: armazena receitas e ingredientes utilizados por elas
-

---

# Analogia da cozinha

- Processador: Lê os programas e executa as instruções, buscando e gravando dados na memória
  - Cozinheiro: Lê as receitas e cozinha os pratos, pegando e guardando ingredientes no armário
-

---

# Sistemas operacionais

- Programas responsáveis por gerenciar o computador
  - Organiza vários programas rodando ao mesmo tempo
  - Cria o sistema de arquivos e pastas
  - Apresenta *interface* para o computador, o que permite interação do usuário
  - Seção especial do armário (memória) que guarda os materiais administrativos do restaurante: regras de limpeza e organização da cozinha, cardápio, etc.
-



---

# Interfaces

- Inicialmente, todos os comandos eram por texto, por meio de uma *Command Line Interface* (CLI), ou *Interface de Linha de Comando*
  - Com o tempo, foram desenvolvidas interfaces mais amigáveis e intuitivas, chamadas de *Graphical User Interface* (GUI), ou *Interface Gráfica*
  - CLIs são acessadas por meio de um Terminal
-

---

# O Terminal

---

---

# O que é o terminal

- Interface que permite interação com o computador por meio de comandos (CLI)
  - Existem vários terminais:
    - Bash
    - CMD
    - PowerShell
    - GitBash
  - Cada terminal possui seus comandos específicos e diferentes funcionalidades
-

---

# Sistemas operacionais e seus terminais

- Windows foi um sistema desenvolvido com foco na interface gráfica. Portanto, os terminais disponíveis são, em geral, menos poderosos
  - Sistemas Unix (Linux e Mac) herdaram o terminal da época em que não existiam interfaces gráficas, o *Bash*
  - No Windows, vamos utilizar o GitBash, um terminal que simula o *Bash*
-

---

---

# Por que usar o terminal?

- Alguns programas não possuem interface gráfica
  - Permite automatização de tarefas.
  - Força maior entendimento por parte do usuário
  - Mais rápido
-

---

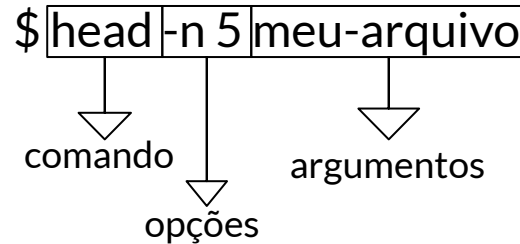
# Um comando no *bash*

- Um comando nada mais é que um programa que executa alguma ação.
  - Pode receber argumentos e opções.
-

---

# Exemplo

- Visualizar as primeiras 5 linhas de um arquivo chamado *meu-arquivo*



---

# Comandos

---



---

# Comandos básicos

- *whoami*: Imprime o seu nome de usuário na tela



```
whoami # retorna o nome do usuário atual ex: 'seuUsuario'
```

---

# Comandos básicos

- *echo*: Imprime algo no terminal



```
echo "Hello World" # imprime Hello World no terminal
```

---

# Comandos básicos

- *man*: Apresenta o manual dos comandos. Pode (e deve) ser usado quando queremos saber o que um comando faz e quais são as opções e argumentos que podemos usar.



```
man echo # mostra o manual do comando echo
```

---

# Comandos básicos

- *pwd*: É uma sigla que representa "print working directory". Mostra a pasta em que você atualmente se encontra.



`pwd` # retorna a pasta que o terminal está atualmente ex: `'/c/Users/seuUsuario'`

---

---

# Comandos básicos

- `cd`: Comando que vem de "Change Directory", significando trocar de diretório (pasta).




```
cd ./minha-pasta # troca o diretório atual para a subpasta 'minha-pasta'  
cd - # volta para a última pasta  
cd # vai para a pasta "home" do usuário atual  
cd ../ # vai para a pasta acima da atual
```

---

# Comandos básicos

- `ls`: Lista os arquivos e pastas presentes na pasta atual.



```
ls # retorna nome de arquivos e pastas presentes na pasta atual
ls -a # retorna nome de arquivos e pastas, incluindo os ocultos (cujo nome começa com `.`)
ls -l # retorna informações de arquivos e pastas, incluindo tamanho, proprietário e outras informações
ls -la # soma dos dois modificadores anteriores
```

---

---

# Manipulando arquivos e pastas

- *touch*: Cria um arquivo.



```
touch index.html # criará um arquivo chamado index.html na pasta atual
```

---

# Manipulando arquivos e pastas

- *mkdir*: Comando que vem da abreviação de "make directory". Cria pastas.



```
mkdir minha-pasta # cria uma pasta chamada 'minha-pasta' no diretório atual
```



---

# Manipulando arquivos e pastas

- *rm*: Comando que vem da palavra "remove", possibilita apagar arquivos, **SEM CONFIRMAÇÃO E CHANCE DE REVERTER**.

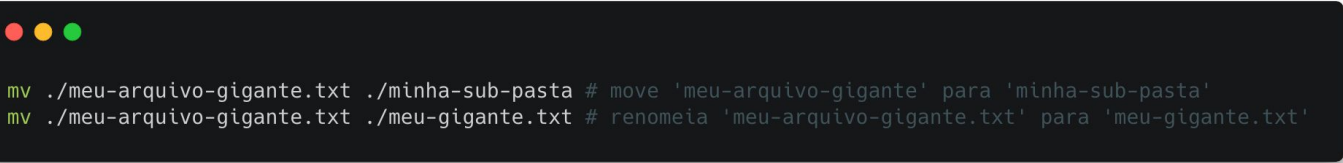


```
rm ./meu-arquivo-gigante.txt # remove imediatamente o arquivo 'meu-arquivo-gigante.txt'  
rm -r ./minha-pasta # remove Recursivamente todos os arquivos e sub-pastas da 'minha-pasta'
```

---

# Manipulando arquivos e pastas

- *mv*: Comando que vem da palavra "move". Move arquivos de uma pasta para outra. Também é utilizado para renomear arquivos.



```
mv ./meu-arquivo-gigante.txt ./minha-sub-pasta # move 'meu-arquivo-gigante' para 'minha-sub-pasta'  
mv ./meu-arquivo-gigante.txt ./meu-gigante.txt # renomeia 'meu-arquivo-gigante.txt' para 'meu-gigante.txt'
```

---

---

# Manipulando arquivos e pastas

- *cp*: Comando que vem da palavra "copy". Copia arquivos de uma pasta para outra.



```
cp ./meu-arquivo-gigante.txt ./minha-sub-pasta # copia 'meu-arquivo-gigante' para 'minha-sub-pasta'
```

---

# Manipulando arquivos e pastas

- *cat*: Vem de *concat*, pois concatena tudo que está no arquivo e imprime na tela.



```
cat meu-arquivo-gigante.txt # imprime o conteúdo do arquivo 'meu-arquivo-gigante.txt'
```

---

# Manipulando arquivos e pastas

- *head*: Imprime as primeiras linhas de um arquivo na tela. Podemos passar o número de linhas com a flag *-n*. O padrão é 10 linhas.



```
head meu-arquivo-gigante.txt # imprime as 10 primeiras linhas do arquivo 'meu-arquivo-gigante.txt'
head -n 20 meu-arquivo-gigante.txt # imprime as 20 primeiras linhas do arquivo 'meu-arquivo-gigante.txt'
```

---

# Manipulando arquivos e pastas

- *head*: Imprime as primeiras linhas de um arquivo na tela. Podemos passar o número de linhas com a flag *-n*. O padrão é 10 linhas.



```
tail meu-arquivo-gigante.txt # imprime as 10 últimas linhas do arquivo 'meu-arquivo-gigante.txt'
tail -n 20 meu-arquivo-gigante.txt # imprime as 20 últimas linhas do arquivo 'meu-arquivo-gigante.txt'
```

---

# Manipulando arquivos e pastas

- *grep*: Permite buscar o conteúdo de um arquivo.



```
grep Future4 ./lista-de-empresas.txt # Busca pela palavra Future4 no arquivo lista-de-empresas.txt e  
imprime toda a linha encontrada
```

---

# Mão na massa

---



## Ex 1. Navegar até a pasta *aula1*



```
pwd # imprime o diretório atual  
ls # mostra todos os arquivos da pasta atual  
cd aula1 # entra na pasta aula1
```

Ex 2. Criar um arquivo chamado *meu-arquivo* e uma pasta *minha-pasta*, os dois na pasta *aula1*



```
touch meu-arquivo # cria um arquivo chamado meu-arquivo  
mkdir minha-pasta # cria uma pasta chamada minha-pasta
```

Ex 3. Copiar o *meu-arquivo* para dentro da *minha-pasta*



```
cp meu-arquivo minha-pasta # copia meu-arquivo para minha-pasta
```

Ex 4. Renomear o arquivo *meu-arquivo* para *meu-arquivo2*



```
mv meu-arquivo meu-arquivo2 # renomeia o meu-arquivo para meu-arquivo2
```

Ex 5. Mover o arquivo *meu-arquivo2* para a *minha-pasta*, com o nome *meu-arquivo3*, com só um comando



```
mv meu-arquivo2 minha-pasta/meu-arquivo3 # move o arquivo para dentro da minha-pasta, e renomeia para meu-arquivo3
```

Ex 6. Entrar na *minha-pasta* e apagar o arquivo *meu-arquivo3*



```
cd minha-pasta
```

```
rm meu-arquivo3 # apaga o arquivo meu-arquivo3
```

Ex 7. Voltar para a pasta *aula1* e apagar a *minha-pasta*



```
cd ..  
rm minha-pasta # gera um erro, pois é uma pasta e não um arquivo  
rm -r minha-pasta # apaga a pasta recursivamente, ou seja, a pasta e tudo que está dentro
```

Ex 8. Entrar na pasta *future4* e imprimir na tela o conteúdo do arquivo *notas*



```
cd future4
```

```
cat notas # imprime o conteúdo do arquivo notas
```



Ex 9. Imprimir as 10 primeiras e as 5 últimas linhas do arquivo *notas*



```
head notas  
tail -n 5 notas
```

Ex 10. Descobrir a sua nota, buscando pelo seu nome



```
grep "seu-nome" notas
```

Ex 11. Descobrir nomes e notas dos 2 alunos seguintes a você



```
grep -A 2 "seu-nome" notas
```

---

# Revisão

---

- 
- Computador surgiu como uma máquina para fazer cálculos seguindo uma sequência de instruções
  - Computadores atuais são compostos basicamente pelos seguintes elementos:
    - Memória, que guarda dados e programas (armário)
    - Processador, que lê a memória e executa os programas (cozinheiro lendo o livro e cozinhando as receitas)
    - Sistema operacional, que gerencia o processador para ler os programas corretamente, e fornece uma interface para o usuário interagir com o computador
  - Sistemas operacionais vêm com dois tipos de interfaces: linha de comando (terminal) e gráfica (a que estamos acostumados)
-

- 
- 
- Utilizamos o terminal pois:
    - Alguns programas só estão disponíveis por ele
    - Possibilidade de automatização
    - Melhor entendimento de como as coisas funcionam no computador
    - Maior rapidez
  - Interagimos com o terminal por meio de comandos, que podem receber argumentos e opções.
-