



Bash

Automação com scripts

Entendendo comandos





\$ command -o argument ... | comando &
command --option argument ... && comando

Variáveis





```
$ var="sou um valor"
```

```
$ export var="sou um valor persistente"
```

```
$ source varFile
```

Como criar, editar,
buscar e listar
arquivos/pastas





ls

Lista arquivos da pasta.

Principais opções:

-a - Lista todos os arquivos(inclusive ocultos) .

Obs: -A não lista os óbvios . e ..

-l - Lista mais informações sobre o arquivo, incluindo permissões e owner.

-h - Transforma tamanhos de arquivos para kB, MB e GB.



cat <files>

Mostra o conteúdo dos arquivos no terminal.

Principais opções:

-v - Mostra caracteres ocultos do arquivo.

Dica: Caso o arquivo seja muito grande, pode-se redirecionar a saída do comando para o comando more



mv <source> <dest>

Move um arquivo da origem para o destino. Sobrescreve o destino caso exista ou cria o arquivo caso contrário. Serve para renomear arquivos.

Principais opções:

- f - Não pede confirmação para sobrescrever o arquivo destino.
- i - Pergunta antes de sobrescrever.
- t <pasta> - Move todos os argumentos <source> para a pasta.



```
cp <source> <dest>
```

Faz uma cópia de source em dest.

Principais opções:

- f - Sobrescreve dest sem confirmação.
- r - Copia diretórios recursivamente.



Criar arquivos

É possível criar arquivos em branco de 3 formas:

\$ > <arquivo>

\$ touch <arquivo>

\$ cat > <arquivo>



locate <name>

Pesquisa o arquivo ou pasta name em todo o computador. É possível que seja necessário rodar o comando updatedb para atualizar o índice de pesquisa.

-b - Pesquisa apenas arquivos com este nome



grep <pattern> <address>

Busca o padrão fornecido nos arquivos do endereço

Principais opções:

- r - Percorre os subdiretórios recursivamente
- e - Usa RegEx nos padrões
- c - Mostra apenas a contagem dos matches

Alterar Permissões de Arquivos





sudo <cmd>

Executa o comando com permissões de superusuário.

Principais opções:

-u <user> - Usuário com o qual o comando será executado.



chown <user> <file>

Altera o owner do arquivo para o usuário fornecido.

Principais opções:

- R - altera recursivamente os arquivos do diretório



chmod <mod> <file>

Altera as permissões do arquivo para o modo fornecido.

Principais opções:

Modo: XXXX ou XXX

1: Execução

2: Escrita

4: Leitura

Pode-se usar a soma

ou

1: Sticky

2: SGID

4: SUID

Pode-se usar a soma



Exercício

Faça um script que faz um backup comprimido de uma lista de pastas informadas como argumentos.

Para acessar os argumentos passados a uma função/comando pode-se usar \$0, \$1... e @\$ para acessar todos como array.

Gerenciamento de pacotes





apt

Gerenciamento de pacotes, instalação de programas

apt install -> Instala um novo pacote

apt remove -> Remove um pacote instalado

apt update -> Atualiza o repositório de pacotes

apt upgrade -> Atualiza pacotes instalados

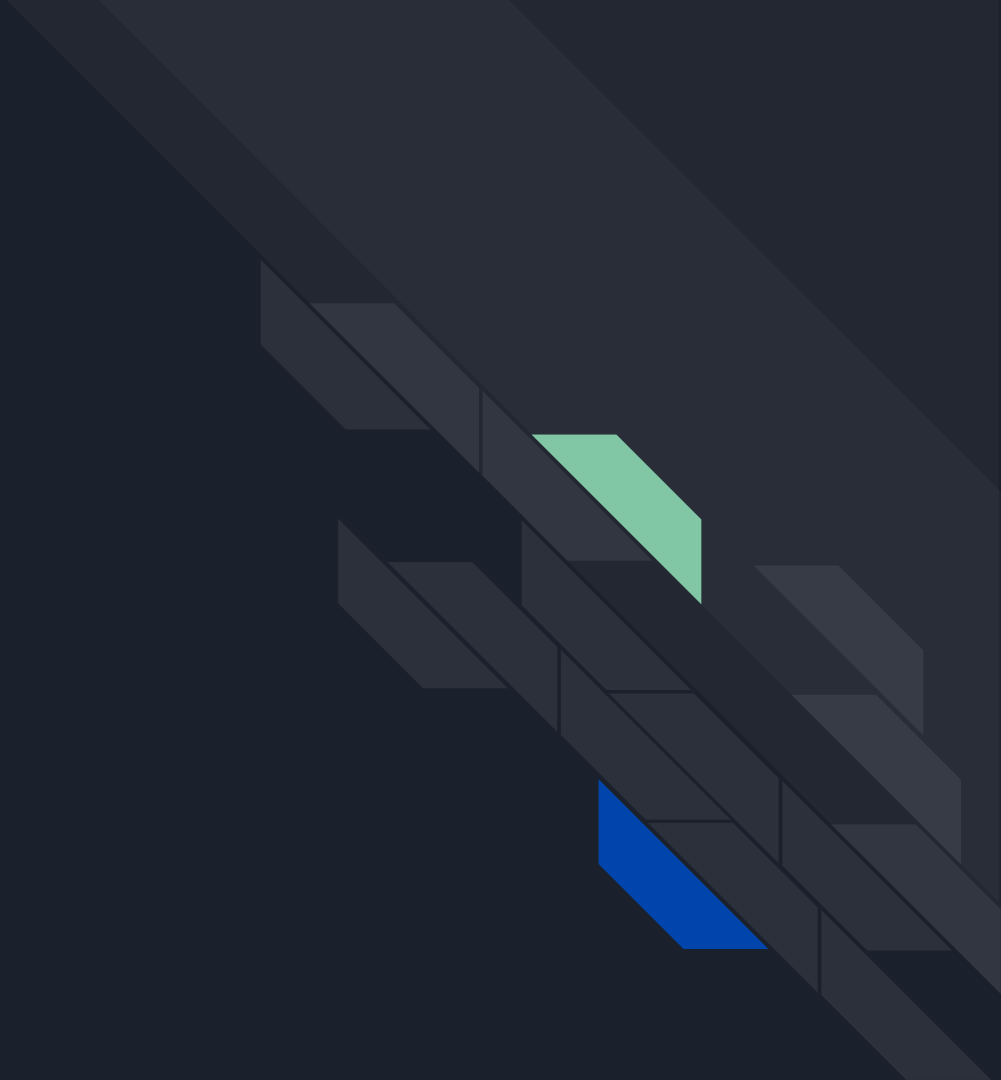


Exercício

Instale os pacotes necessários para desenvolver uma aplicação na linguagem de sua escolha.

Instale os pacotes necessários para desenvolver uma aplicação em python.

Rede e Internet





wget <url>

Baixa o conteúdo da url para a pasta atual.

-O <file> - Define o arquivo de output



curl <url>

Executa uma requisição HTTP, FTP, IMAP, POP3 ou SMTP

Principais opções:

-d <cont> - Define o conteúdo da requisição, da mesma forma que um clique em submit de uma página web

-H <cont> - Define os headers da requisição

-X <req> - Define o tipo de requisição de acordo com o protocolo

Ex: `$ curl -X POST https://us-central1-sinuous-wording-252015.cloudfunctions.net/function-1 -H "Content-Type:application/json" -d '{"name":"Hello"}'`



ping <endereço>

Envia requisições ICMP e verifica acesso e latência. É capaz de realizar resolução de nome via DNS.

Principais opções:

-c <n> - número de pacotes a enviar.

Ex: \$ ping -c 3 Google.com



traceroute <endereço>

Analisa roteadores no caminho entre a origem e o endereço

Ex: \$ traceroute google.com



nslookup <url>

Realiza a resolução de nome da url informada

Ex: \$ nslookup google.com

Redirecionamento de Entrada e Saída





`<cmd_1> | <cmd_2>`

Direciona a saída do comando 1 para a entrada do comando 2.

Ex.:

`ls -l | grep -c "^d"`

Conta o número de diretórios na pasta atual



cmd > file

Direciona a saída do comando para o arquivo. Pode direcionar toda a saída ou apenas parte dela com os seletores associados.

> Sobrescreve o arquivo com a saída. Não direciona a saída de erros

>> Escreve no final do arquivo a saída.

&> ou &>> Direciona a saída e os erros para o arquivo

2> ou 2>> Direciona apenas os erros para o arquivo



cmd < file

Direciona o arquivo para a entrada padrão do comando.

Concatenar comandos





`<cmd_1> && <cmd_2>`

Retorna verdadeiro se ambos são verdadeiros. As otimizações do Linux fazem com que o segundo comando só execute se o primeiro for bem sucedido.

Ex.:

```
mkdir teste && cd teste
```

Só entra no diretório se ele pode ser criado



`<cmd_1> || <cmd_2>`

Retorna verdadeiro se ao menos um é verdadeiro. As otimizações do Linux fazem com que o segundo comando só execute se o primeiro for mal sucedido.

Ex.:

```
cd test || {mkdir test && cd test}
```

Caso o diretório não exista, cria e entra nele. Caso contrário, apenas entra no diretório

Prioridade de execução





``cmd`` e `${cmd}`

Executa os comandos antes do comando externo, mesmo que o comando esteja dentro da *escape sequence* “...”.

Ex.:

```
echo "Hello, ${whoami}"
```

Obs.: Todas as versões do shell suportam ```, mas nem todas suportam `${}`

Processamento de texto





sed

Realiza processamento e edição de texto linha a linha. Para edição multilinhas, pesquise sobre awk.

Ex. de uso para substituição:

```
sed -e "s/<padrao_substituido>/<substituicao>/g"
```



Exercício

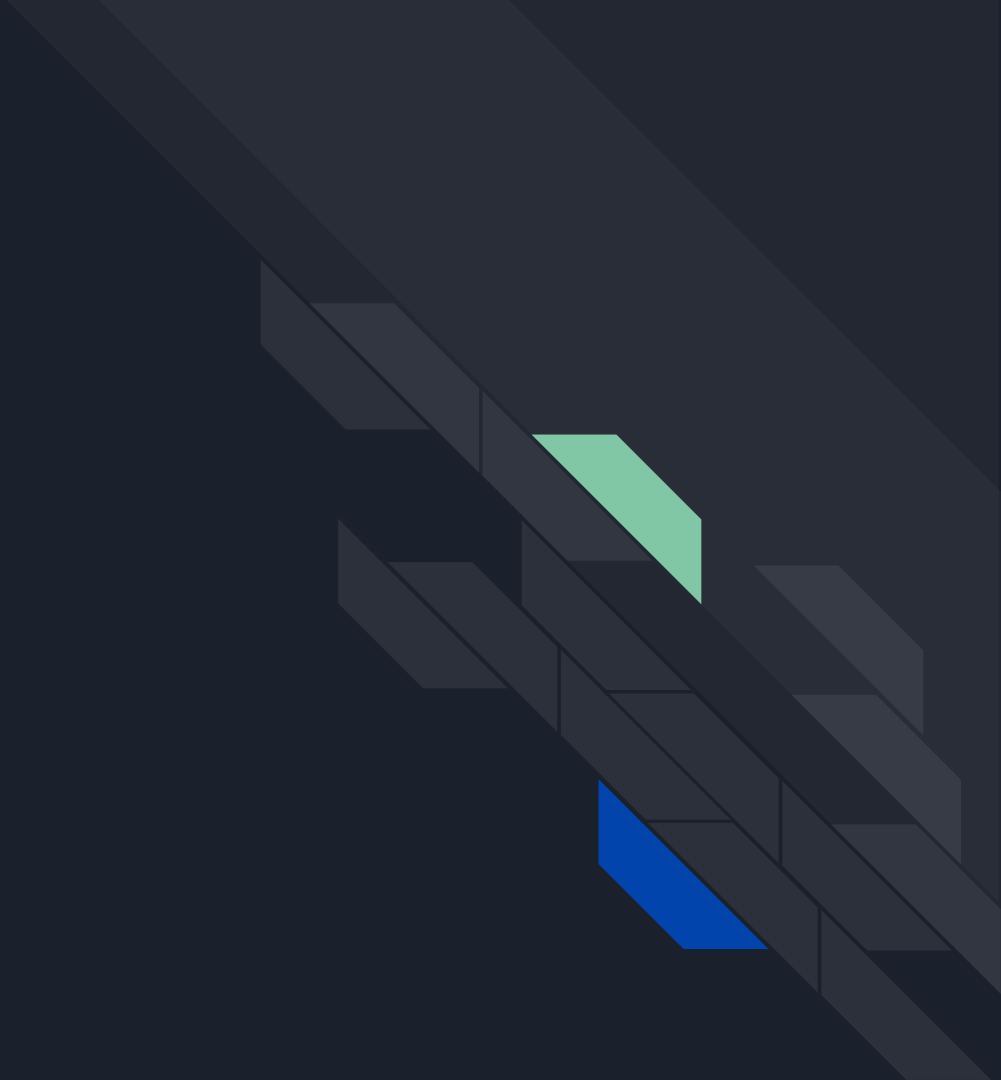
Crie um script que recebe um nome de projeto de web api, cria a estrutura básica de pastas, inicia o repositório git baixando templates de um outro repositório e substituindo valores quando apropriado e instala as dependências básicas. (Faça para o python e para a linguagem de sua escolha. No python inclua venv ou solução similar e fastapi ou solução similar)


Inclua também um gerenciador de tarefas do projeto estilo npm/yarn que possui comandos para rodar as principais tarefas do projeto: lint, test, execução, build...

Dúvidas?



Git





git init

Inicializa um repositório vazio no diretório atual



git remote add <name> <url>

Adiciona o endereço remoto com o nome informado e a url informada.

É utilizado para definir a origin na criação do repo geralmente.

Ex.:

```
git remote add origin https://github.com/user/repo
```



```
git add <files>
```

Adiciona os arquivos informados ao repositório.



git commit <file>

Cria o commit acrescentando o(s) arquivo(s) selecionados.

Principais opções:

- a - Envia todos os arquivos alterados no repositório
- m "msg" - Define a mensagem de commit (Ser descritivo é tudo)



git push (-u origin master)

Envia os commits locais para o repositório remoto.

A parte do comando entre parênteses é utilizado na primeira utilização do push para definir o endereço remoto.

Caso haja conflito, os arquivos serão marcados e devem ser editados e um commit de merge deve ser realizado.



git checkout <branch>

Alterna para a branch especificada. Mudanças que não foram incluídas em um commit serão mantidas caso a branch atual tenha uma versão mais nova ou de mesma idade que a branch alvo. Caso contrário, você deverá fazer o commit da mudança antes de mudar de branch.

Principais opções:

- b - Cria a branch caso não exista.