# LISTAS – TÓPICOS SO

Nome: João Vítor Fernandes Dias

## Lista de exercícios 1 – Docker

### **\* 1 - Resumidamente, o que significa a palavra contêiner?**

Pode-se entender um container como outra forma de virtualização. VMs (que também são outras formas de virtualização) permitem que uma parte do hardware execute vários SO como software (?). VMs são adicionadas à máquina principal para que o poder desse hardware possa ser distribuído entre os diferentes usuários e apareça como máquinas ou servidores separados. Containers virtualizam o OS, separando-o em compartimentos virtualizados para rodar aplicações em container.

### **\* 2 - Cite algumas vantagens de dockerizar um programa/aplicação**

Um Docker Container permite que softwares sejam portáteis para qualquer sistema operacional que rode Linux ou Windows (Uma máquina rodando o SO Windows consegue rodar containers Linux usando uma máquina virtual (VM));

### **\* 3 - Execute "docker run --rm chuanwen/cowsay" e descreva o resultado! Pode repetir a execução se quiser.**

R: Uma imagem foi pesquisada, mas não encontrada, então começou-se a baixar os arquivos referentes à imagem pesquisada e depois surge uma vaca dizendo uma frase pré-definida aleatoriamente.

#### Cowsays

##### Cowsay 1

PS F:\UENF\4Periodo\Topicos> docker run --rm chuanwen/cowsay

Unable to find image 'chuanwen/cowsay:latest' locally

latest: Pulling from chuanwen/cowsay

99ad4e3ced4d: Pull complete

ec5a723f4e2a: Pull complete

2a175e11567c: Pull complete

8d26426e95e0: Pull complete

46e451596b7c: Pull complete

c337767f8c73: Pull complete

Digest: sha256:1f7a652a47fe7311c7e201644d44682e11e7ae4d3d7b03c1ce5c0df164de205c

Status: Downloaded newer image for chuanwen/cowsay:latest

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/ Please read the prospectus carefully \

\ before you invest or send money. /

--------------------------------------

\ ^\_\_^

\ (oo)\\_\_\_\_\_\_\_

(\_\_)\ )\/\

||----w |

|| ||

##### Cowsay 2

PS F:\UENF\4Periodo\Topicos> docker run --rm chuanwen/cowsay

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/ Memory fault -- \

| core...uh...um...core... Oh dammit, I |

\ forget! /

---------------------------------------

\ ^\_\_^

\ (oo)\\_\_\_\_\_\_\_

(\_\_)\ )\/\

||----w |

|| ||

##### Cowsay 3

PS F:\UENF\4Periodo\Topicos> docker run --rm chuanwen/cowsay

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

< Yow! I threw up on my window! >

-------------------------------

\ ^\_\_^

\ (oo)\\_\_\_\_\_\_\_

(\_\_)\ )\/\

||----w |

|| ||

##### Cowsay 4

PS F:\UENF\4Periodo\Topicos> docker run --rm chuanwen/cowsay

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

/ Trust your husband, adore your husband, \

| and get as much as you can in your own |

| name. |

| |

\ -- Joan Rivers /

-----------------------------------------

\ ^\_\_^

\ (oo)\\_\_\_\_\_\_\_

(\_\_)\ )\/\

||----w |

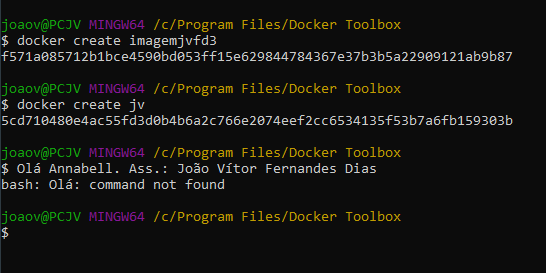
|| ||

PS F:\UENF\4Periodo\Topicos>

### **Lista de exercícios 1.1 – Docker (Extra)**

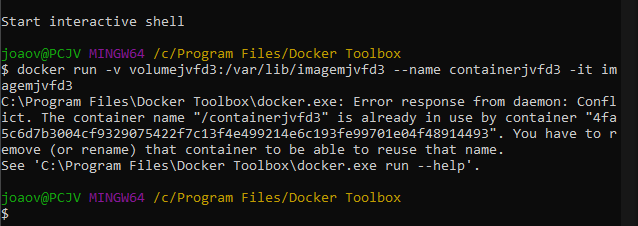
Já que temos a instalação do ambiente Docker completa, vamos então rodar e apresentar o print das seguintes tarefas. Lembra de colocar a identificação pessoal de cada um:

#### 1- Inicialize um container.



docker run -v <nome-do-volume>:/var/lib/<nome-da-imagem> --name <nome-do-conteiner> -it <nome-da-imagem>

docker run -v volumejvfd3:/var/lib/imagemjvfd3 --name containerjvfd3 -it imagemjvfd3



???

https://docs.docker.com/engine/reference/commandline/app\_init/

docker app init APP\_NAME [--compose-file COMPOSE\_FILE] [--description DESCRIPTION] [--maintainer NAME:EMAIL ...] [OPTIONS]

| Name, shorthand | Default | Description |
| --- | --- | --- |
| --compose-file |  | Compose file to use as application base (optional) |
| --description |  | Human readable description of your application (optional) |
| --maintainer |  | Name and email address of person responsible for the application (name:email) (optional) |
| --single-file |  | Create a single-file Docker Application definition |

???

https://docs.docker.com/engine/reference/commandline/create/

docker create [OPTIONS] IMAGE [COMMAND] [ARG...]

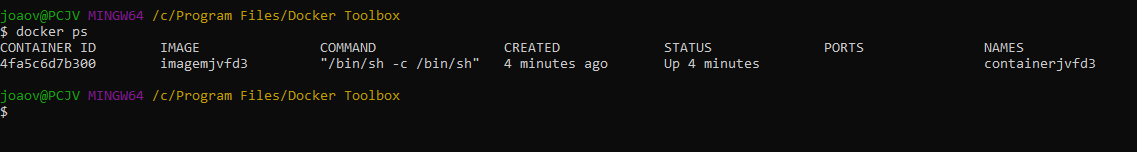
???

docker start <container\_id>

#### 2- Visualize os containers criados.

https://docs.docker.com/engine/reference/commandline/ps/

docker ps [OPTIONS]



Name, shorthand Default Description

--all , -a Show all containers (default shows just running)

--filter , -f Filter output based on conditions provided

--format Pretty-print containers using a Go template

--last , -n -1 Show n last created containers (includes all states)

--latest , -l Show the latest created container (includes all states)

--no-trunc Don't truncate output

--quiet , -q Only display container IDs

--size , -s Display total file sizes

https://docs.docker.com/engine/reference/commandline/container\_ls/

docker container ls [OPTIONS]

Name, shorthand Default Description

--all , -a Show all containers (default shows just running)

--filter , -f Filter output based on conditions provided

--format Pretty-print containers using a Go template

--last , -n -1 Show n last created containers (includes all states)

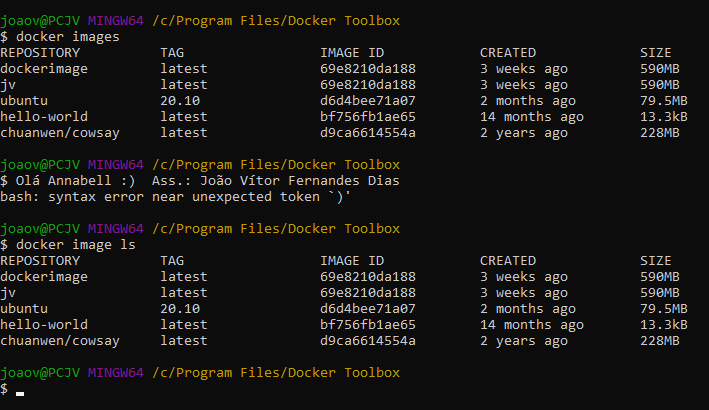
--latest , -l Show the latest created container (includes all states)

--no-trunc Don't truncate output

--quiet , -q Only display container IDs

--size , -s Display total file sizes

#### 3- Visualize as imagens.



https://docs.docker.com/engine/reference/commandline/images/

docker images [OPTIONS] [REPOSITORY[:TAG]]

docker images

Name, shorthand Default Description

--all , -a Show all containers (default shows just running)

--filter , -f Filter output based on conditions provided

--format Pretty-print containers using a Go template

--last , -n -1 Show n last created containers (includes all states)

--latest , -l Show the latest created container (includes all states)

--no-trunc Don't truncate output

--quiet , -q Only display container IDs

--size , -s Display total file sizes

Name, shorthand Default Description

--all , -a Show all images (default hides intermediate images)

--digests Show digests

--filter , -f Filter output based on conditions provided

--format Pretty-print images using a Go template

--no-trunc Don't truncate output

--quiet , -q Only show image IDs

https://docs.docker.com/engine/reference/commandline/image\_ls/

docker image ls [OPTIONS] [REPOSITORY[:TAG]]

Name, shorthand Default Description

--all , -a Show all images (default hides intermediate images)

--digests Show digests

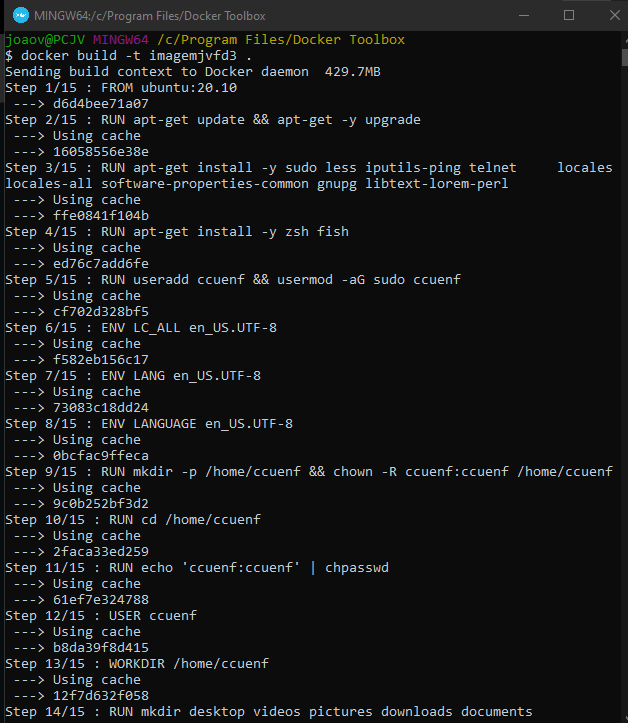
--filter , -f Filter output based on conditions provided

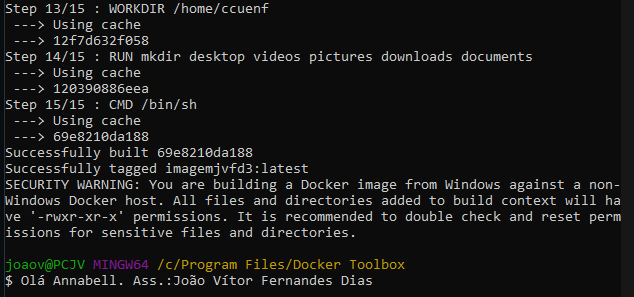
--format Pretty-print images using a Go template

--no-trunc Don't truncate output

--quiet , -q Only show image IDs

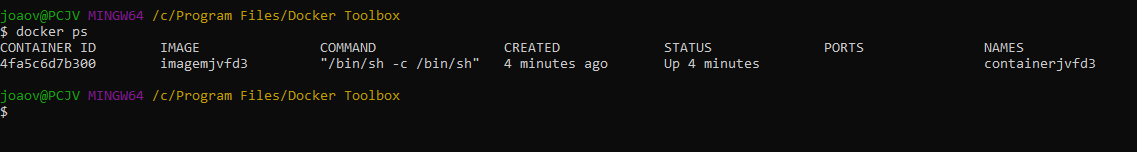
#### 4- Crie uma imagem a partir do Dockerfile.





**Criar imagem:** docker build -t <nome-da-imagem> .

#### 5- Execute essa imagem usando um volume criado.



**Criar volume:** docker volume create <nome-do-volume>

**Executar a imagem:** docker run -v <nome-do-volume>:/var/lib/<nome-da-imagem> --name <nome-do-conteiner> -it <nome-da-imagem>

**Criar imagem:** docker build -t <nome-da-imagem> .

**Criar volume:** docker volume create <nome-do-volume>

**Executar a imagem:** docker run -v <nome-do-volume>:/var/lib/<nome-da-imagem> --name <nome-do-conteiner> -it <nome-da-imagem>

**religar o container** docker start <nome-do-conteiner>

docker exec -it <nome-do-conteiner> /bin/sh

## Lista de exercícios 2 – Linux

### **1 - Explique a definição de “Linux”**

<https://4linux.com.br/o-que-e-linux/>

Linux é o nome dado apenas ao núcleo do sistema operacional, chamado de Kernel.

Kernel é um conjunto de instruções que controla como será usado o processador, a memória, o disco e dispositivos periféricos. É o software presente em todo sistema operacional que dita como o computador deve funcionar. O Kernel Linux foi criado pelo Linus Torvalds, com a primeira versão oficial lançada em 1991.

O Kernel por si só não tem utilidade prática. É preciso uma série de programas adicionais para seu uso efetivo, como interpretadores de comandos, compiladores para que seja possível o desenvolvimento de novos programas, editores de textos e assim por diante.

### **2 - Dado os diretórios da primeira tabela, correlacione-os com suas funções na segunda tabela:**

|  |  |
| --- | --- |
| opção | diretório |
| a | /dev |
| b | /etc |
| c | /usr |
| d | /opt |
| opção | função |
| C | Localização de arquivos, programas e bibliotecas do escopo de usuário ficam; não são dados essenciais |
| A | Diretório destino das mapeações de cada hardware |
| D | Aqui são encontrados softwares adicionais e proprietários |
| B | Local onde as configurações do sistema ficam disponíveis de forma global |

### **3 - Escolha 3 (três) diretórios Linux a partir da raiz (/) e explique sua respectiva função.**

Regras: Não vale repetir os diretórios da questão anterior ☺

1. /media
   1. Diretório onde podem ser encontradas os dispositivos de armazenamento externo como pendrive e HD externo (e também unidades de rede);
2. /boot
   1. Diretório onde estão localizadas as configurações de boot do sistema, o bootloader e o grub;
3. /home
   1. Diretório onde podem ser encontrados os arquivos pessoais de cada usuário criado no SO.

### **4 - Qual a finalidade de existirem grupos em sistemas operacionais baseados no Linux?**

Grupos permitem que definir um conjunto de permissões a um usuário de uma só vez, sem precisar definí-las uma a uma para cada usuário.

### **5 - Descreva cada permissão:**

Exemplo:

.rw-rw-rw- -> . significa xxx, yyy tem permissões de nnn e mmm, etc…

|  |  |
| --- | --- |
| permissão | descrição |
| .rw-rw-rw- | Dono, Grupo e Outros: Ler e Escrever |
| lr-x---rwx | Dono: Ler e Executar; Outros: Ler, Escrever e Executar |
| drwxr-xr-x | Dono: Ler, Escrever e Executar; Grupo e Outros: Ler e Executar |
| .rw-r--r-- | Dono: Ler e Escrever; Grupo e Outros: Ler |
| drwxrwxrwx | Dono, Grupo e Outros: Ler, Escrever e Executar |

#### Conteúdo

R -> Read

W -> Write

X -> Execute

Os 3 primeiros representando as permissões do usuário que criou o arquivo

Os 3 seguintes são as permissões do grupo o qual o dono do arquivo pertence

Os 3 últimos mostram as permissões dos demais usuários

### **6 - Defina STDIN, STDOUT e STDERR**

STDIN é o tipo de entrada padrão. Geralmente dispõe um espaço da tela para que alguma informação seja inserida através do teclado.

STDOUT é o meio de imprimir o output gerado por comandos shell. Por padrão pode retornar 0 caso o comando tenha sido bem sucedido, também pode retornar 1 caso tenha ocorrido alguma falha.

STDERR é uma mensagem de erro padrão utilizada para imprimir o output em um terminal. Também é usada para guardar as mensagens de erros geradas por softwares e comandos.

### **7 - Diferencie uma CLI de uma GUI, apontando suas vantagens e desvantagens**

A CLI (Command Line Interface) tem algumas de suas vantagens ser mais direta e objetiva pois dispõe de diversos comandos específicos para realizar boa parte das coisas que seriam no GUI, porém sem precisar passar por várias telas, consequentemente poupando parte do processamento que seria despendido, sendo então mais eficiente. Entretanto, tem como algumas desvantagens a pouca atratividade aos usuários gerais, podendo muitas vezes passar despercebida e também a limitação quanto ao seu uso para várias tarefas, visto que um comando precisa ser concluído para que outro seja executado logo em seguida.

Já o GUI (Graphic User Interface) se mostra muito mais atrativo aos usuários comuns, tão comum que se tornou o padrão utilizado mundialmente, tanto em computadores pessoais quanto em smartphones tendo como desvantagem a redução na eficiência e não ter todas as funcionalidades presentes na CLI.

(OBS: Não sei o motivo de ter colocado GUI no masculino e CLI no feminino)

## Lista de exercícios 3 – Shell

### **1 – O que é Shell?**

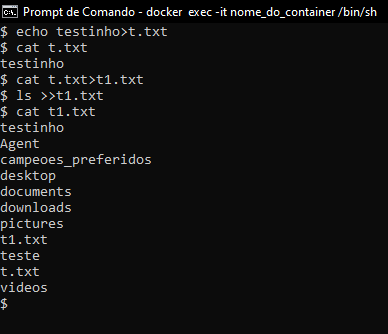
Com suas palavras, explique o que é um shell:

R:

É um software que lê e executa comandos que são inseridos nele (geralmente através do teclado para que sejam realizadas operações no computador.

### **2 – Sequência de comandos**

Quero executar esta sequência de comandos: cat xx.txt, ls, sort e guardar num arquivo .txt. Cite ao menos uma forma de fazer isso, de maneira que o arquivo final tenha o conteúdo do ls e do cat, não importando a ordem dos comandos.



### **3 – Shebang**

Qual a utilidade de usar a técnica de shebang?

R:

É basicamente a ideia de modularização de código em programação. Com isso você pode compactar uma série de comandos em um único arquivo que poderá posteriormente ser chamado pelo seu nome para executar os comandos internos desse arquivo.

### **4 – Sequência de Comandos 2**

Abaixo, serão listadas várias ações. Execute uma por vez no terminal e escreva o comando que usou logo em seguida.

Estrutura de diretórios sugerida

./ccuenf

./exercicios

|  |  |
| --- | --- |
| ação | comando |
| Verifique em qual diretório você está | pwd |
| Crie a estrutura de diretórios sugerida logo acima na home do usuário | mkdir ccuenf exercicios |
| Crie um arquivo “numeros.txt” com o cat, não importa a quantidade de números | cat > numeros.txt |
| Duplique “numeros.txt” para “numeros1.txt” e “numeros2.txt” | cat numeros.txt>numeros1.txt && cat numeros.txt>numeros2.txt |
| Copie todos os arquivos com extensão .txt para a pasta “exercicios” | cp /home/ccuenf/teste/\*.txt exercicios |
| Exiba os arquivos da pasta “exercicios” com seus detalhes (permissões, tamanho e etc) sem sair da pasta atual | ls -lh |
| Apague a pasta “ccuenf” e “exercicios” | ls -lh |

### **5 – Listar arquivos da pasta /bin**

Liste todos os arquivos da pasta /bin, inverta a ordem (de Z para A) e depois salve num arquivo. Quais foram os comandos utilizados? Não é obrigatório executar a sequência em apenas uma linha.

ls /bin -r > coisasbin.txt

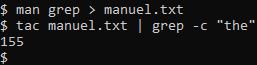
### **6 – Inverter ordem das Linhas**

Crie um arquivo .txt com o centeúdo da execução de man <comando> - onde “comando” é qualquer comando linux conhecido e que possua um manual - e inverta a ordem das linhas, procure pela palavra “the” e conte o número de ocorrências. Coloque aqui seu resultado e qual comando usou para pegar o manual.

**man grep > manuel.txt**

**tac manuel.txt | grep -c "the"**

O resultado foi: 155



### **7 – Instalar um programa**

Escolha um ou mais programas da lista e tente **instalá-lo**, **execute-o** e descreva o resultado.

Programas:

* sl
* cmatrix
* rig
* nyancat

Após terminar o exercício, remova o programa

**Instalando**: sudo apt install nyancat

**Executando**: nyancat



Meu deus... como eu saio?!



Ui

Consegui sair usando o Ctrl+C

**Descrição:** Um contador em segundos de quanto tempo o nyancat ficou “nyanzando”

**Desinstalando:** sudo apt remove nyancat

### **8 – Resultado das expressões**

Qual o resultado das seguintes expressões?

Já que as colunas da tabela são formadas por |, não posso escrever o pipe dentro da tabela. Tem como escapar com o \vert, porém ele só é renderizado no org-mode do emacs :/

\vert == |

|  |  |
| --- | --- |
| expressão | resultado |
| true && echo foo || echo bar | foo |
| false && echo foo || echo bar | bar |
| true || echo foo && echo bar | bar |
| false || echo foo && echo bar | foo  bar |

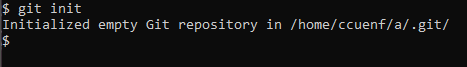
## Lista de exercícios 4 – Git e GitHub

### **1 - Qual a diferença entre git e GitHub? Como os dois estão relacionados?**

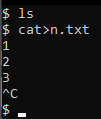
O Git é uma ferramenta para criar e gerenciar diferentes versões de um mesmo projeto. Já o GitHub é uma plataforma online que disponibiliza o armazenamento de códigos e o uso do Git para gerenciá-los.

### **2 - Execute o seguinte git-flow:**

#### Crie um repositório local



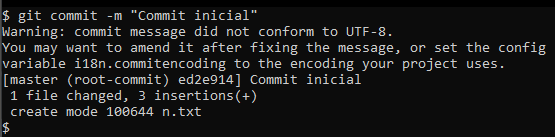
#### Crie um ou mais arquivos com conteúdo



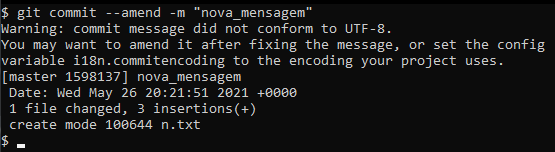
#### Adicione todo o diretório ao Staging

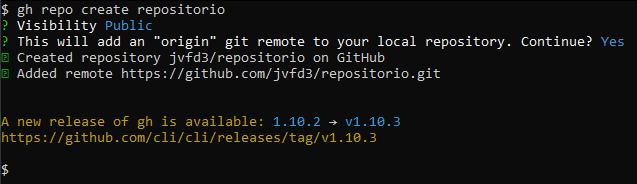


#### Crie um commit

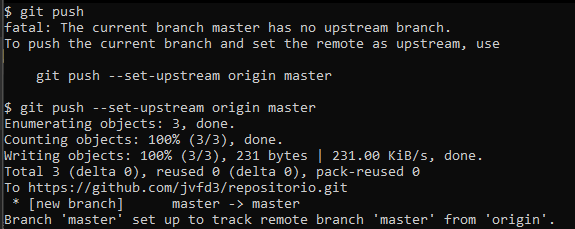


#### Edite a mensagem do último commit

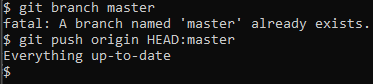


1. Crie um repositório remoto com a cli do github ou no site deles 

#### Adicione a url do repositório remoto no seu repositório local



#### Envie os commits locais para o repositório remoto



#### Coloque aqui o link para seu repositório remoto!

https://github.com/jvfd3/repositorio