



Universidade Federal de Viçosa  
Campus Florestal  
CCF 451 - Sistemas Operacionais

## Trabalho Prático 1

Nome: Gabriel Vitor da Fonseca Miranda

Matrícula: 3857

### Sumário:

1- Atividade de Preparação do Ambiente.....	2
2- Execução do AutoRuns.....	8
3- Execução do Process Explorer.....	9
4- Execução do BgInfo.....	12
5- Shell script.....	13
6- Comando ps.....	15
7- 5 Comandos para o Linux.....	18
8- Conceito da Disciplina.....	22
9- Conclusão.....	22

## 1. Atividade de Preparação do Ambiente

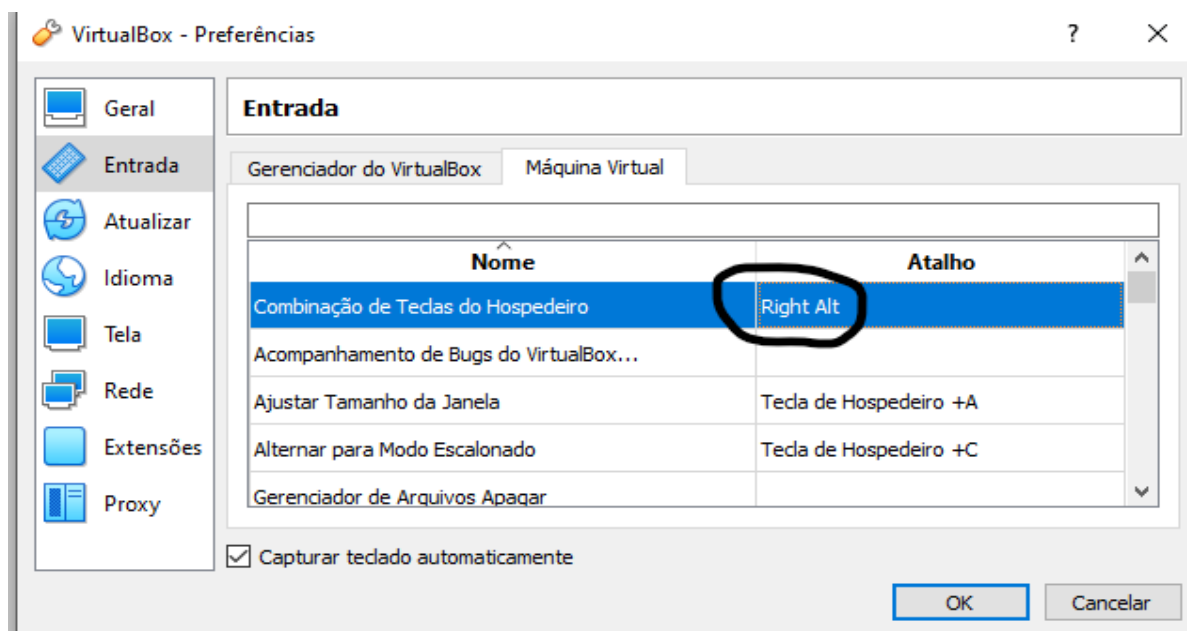
Para este trabalho, inicialmente, será feita a preparação do ambiente para serem desenvolvidas as atividades. Como o Sistema Operacional da máquina que será usada para os teste é o windows, faz-se necessário então, instalar uma maquina virtual para ter acesso ao ambiente Linux.

Sendo assim, primeiramente será instalado o VirtualBox, logo após será criado uma VM Linux Ubuntu, depois será instalado o Ubuntu nesta VM. Um dos pré-requisitos para esta instalação é possuir um computador Windows com no mínimo 8GB de memória ram.

### Passos para a instalação do VirtualBox:

#### Checklist:

- Google: Pesquise no google por VirtualBox. Vá em Downloads, e procure por Windows hosts.
- Instalar com as configurações padrão.
- Configurando a host: File(Arquivo) -> Preferences(preferências)-> Input(Entrada) -> Virtual Machine(Máquina Virtual) -> Hostkey Combination (Combinação de Teclas de Hospedeiro).

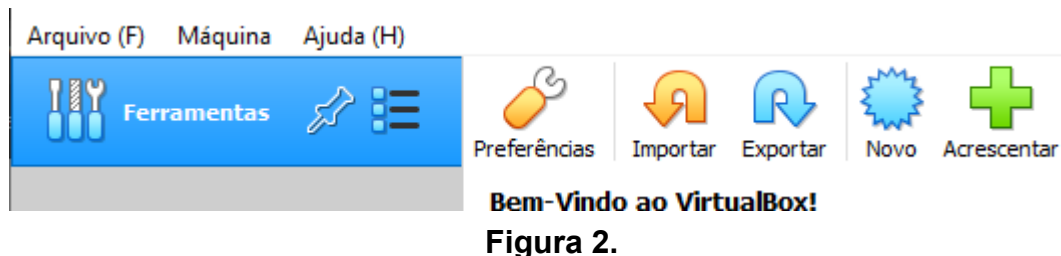


**Figura 1.**

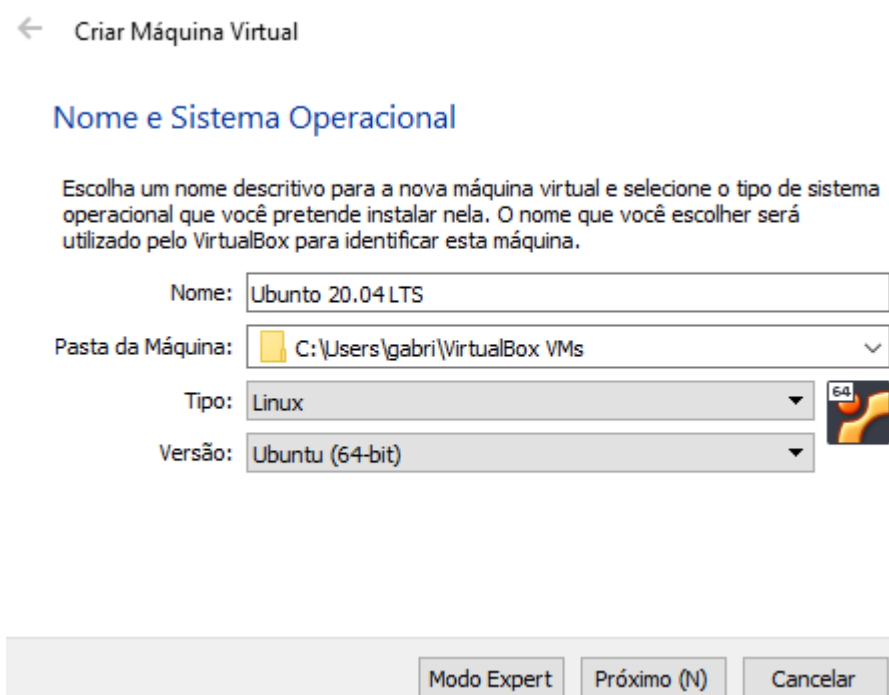
Então, apague o nome que está nessa opção usando uma borracha ao lado, e em seguida aperte o Alt Direito. Dessa forma, a palavra que aparece na **Figura 1** que está circulada irá aparecer, então clique no botão OK.

## Preparando o Ubuntu

Com seu VirtualBox aberto, aperte em New(Novo).



Então irá abrir uma pequena janela onde serão adicionados o nome da máquina, a pasta onde a máquina será armazenada, o tipo, que é o Sistema Operacional Linux, e sua versão. O que foi preenchido na **Figura 3** replique em sua janela, e prossiga apertando em próximo.



Nessa parte, será selecionado a quantidade de memória (RAM) em megabytes que será alocado para a máquina virtual. É recomendado que sejam pelo menos 4GB de memória ram alocadas, no caso 4096 MB. Então siga o exemplo da **Figura 4** abaixo e clique em próximo.

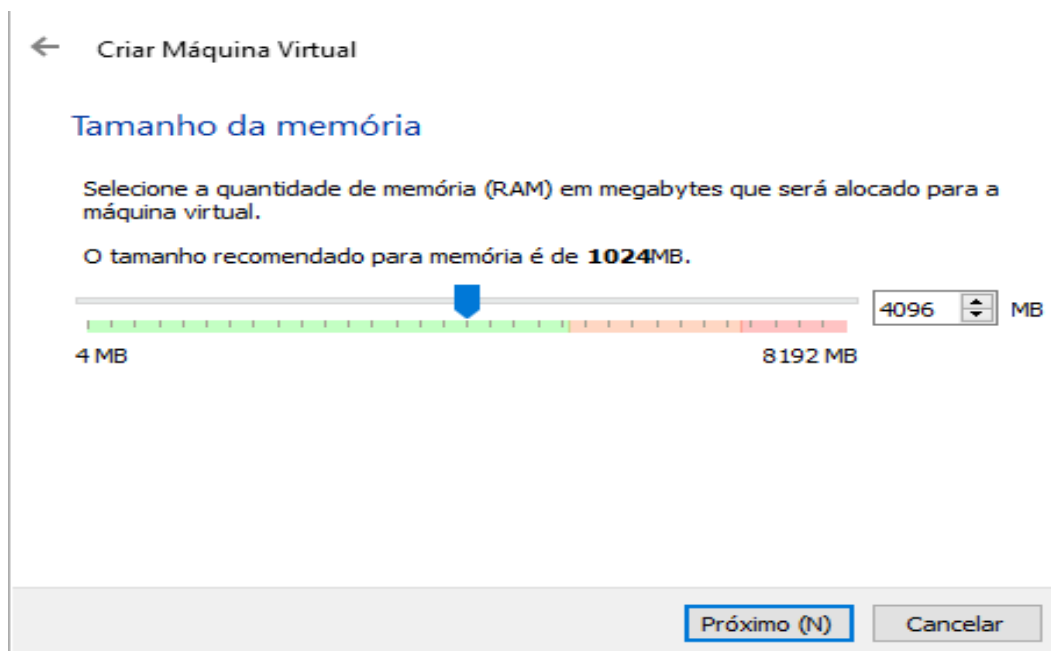


Figura 4.

Na parte de disco rígido, você tem que apenas que criar um novo disco rígido. **Figura 5.**

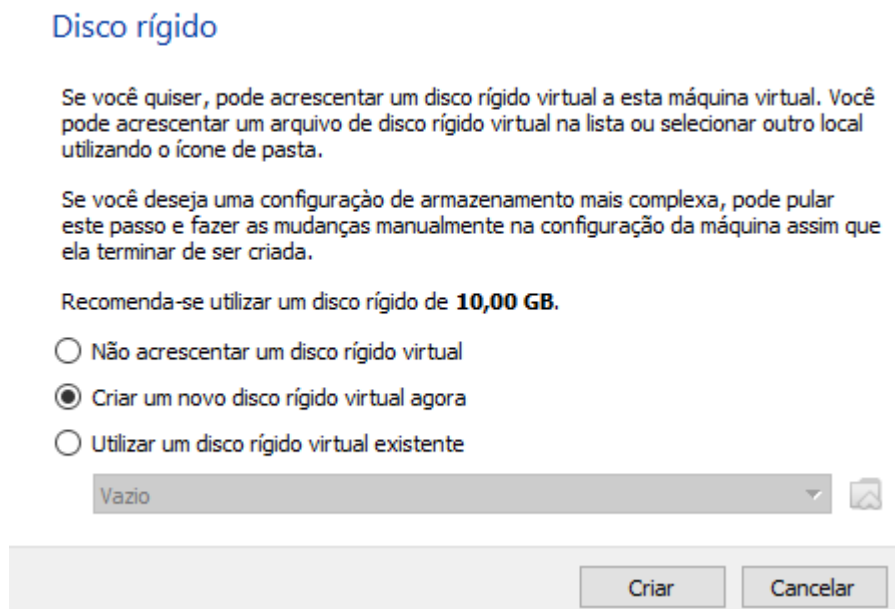


Figura 5.

Tipo de arquivo de disco rígido:

**Tipo de arquivo de disco rígido**

Escolha o tipo de arquivo que você gostaria de utilizar para o novo disco rígido virtual. Caso não necessite utilizá-lo com outros softwares de virtualização, pode deixar esta opção como está.

☒ VDI (VirtualBox Disk Image)  
☐ VHD (Virtual Hard Disk)  
☐ VMDK (Virtual Machine Disk)

Modo Expert   **Próximo (N)**   Cancelar

**Figura 6.**

Na próxima janela, é perguntado se você deseja que a memória utilizada seja alocada dinamicamente, é bom colocar esta opção já que a memória vai ir crescendo a medida que for precisando.

Na janela Localização e tamanho do arquivo, recomenda-se colocar no mínimo 20 GB para a maquina virtual, no caso esse seria o máximo de memória que ela poderia armazenar, para o trabalho coloquei 40GB. Depois é so clicar em criar, e a máquina virtual é criada.

## Instalando o Ubuntu

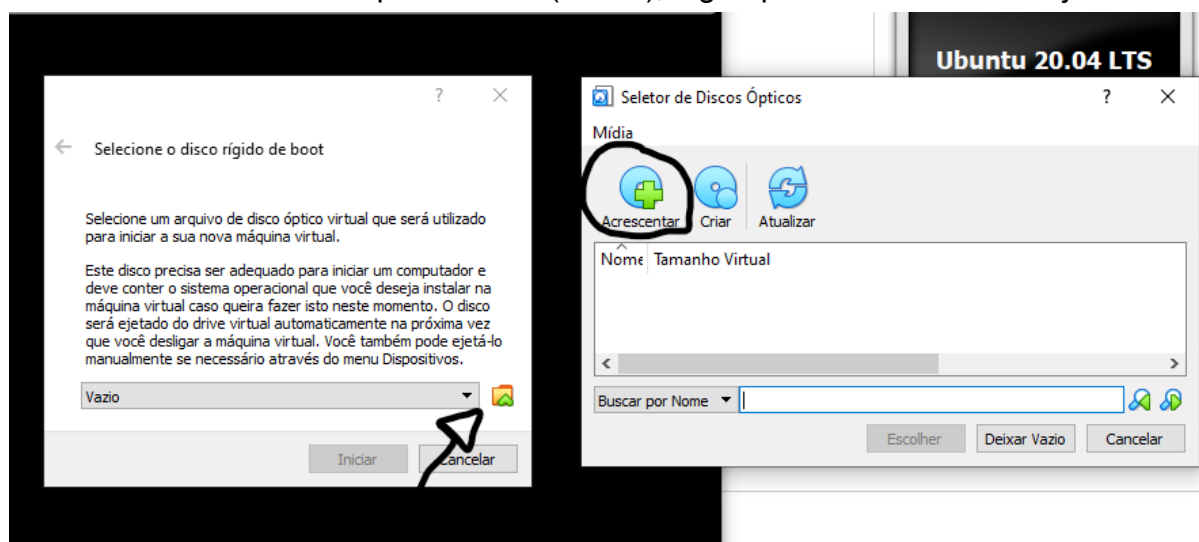
Nesta parte será feito o dowload do Ubuntu, para isto entre no Google e pesquise Ubuntu para desktop, clique no primeiro link do Google, entre no site do google e baixe o **Ubuntu Desktop, versão: 20.04 LTS**.

### Checklist:

- Google: Ubuntu Dowload.
- Baixar ISO
- VirtualBox -> Start
  - Select start-up disk
  - Add -> ISO -> Start
  - English -> Install Ubuntu
  - Keyboard layout -> Portuguese (Brazil)
    - Testar / Tab->Tab->ENTER

- Updates and other software
  - Normal Installation.
  - Download updates while installing Ubuntu.
  - Install third-party software.
- Erase disk and install third-party software.
- Instalar com as configurações padrão.

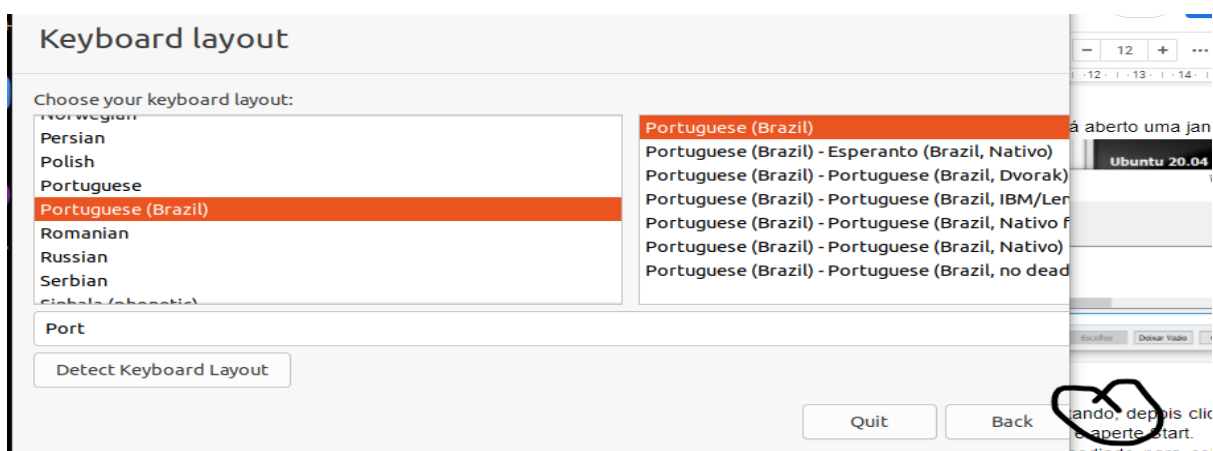
Neste momento clique em Start(Iniciar), logo após será aberto uma janela:



**Figura 7.**

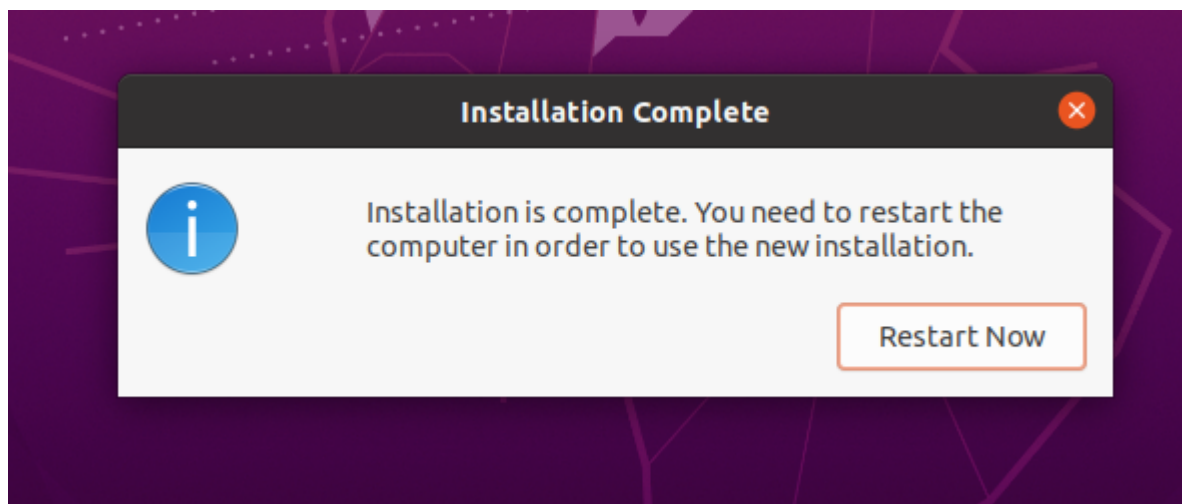
Nesta janela clique na pasta onde a seta está apontando, depois clique em acrescentar. Então adicione a versão Linux baixada, escolha e aperte Start.

Então espere até abrir uma janela no VirtualBox, pedindo para colocar o idioma da sua máquina, escolha o idioma e aperte em instalar Ubuntu. Aparecerá também a escolha do tipo de teclado, fica a sua escolha. Nesse passo não aparecerá o botão continue, então aperte TAB até a opção do botão continue e pressione:



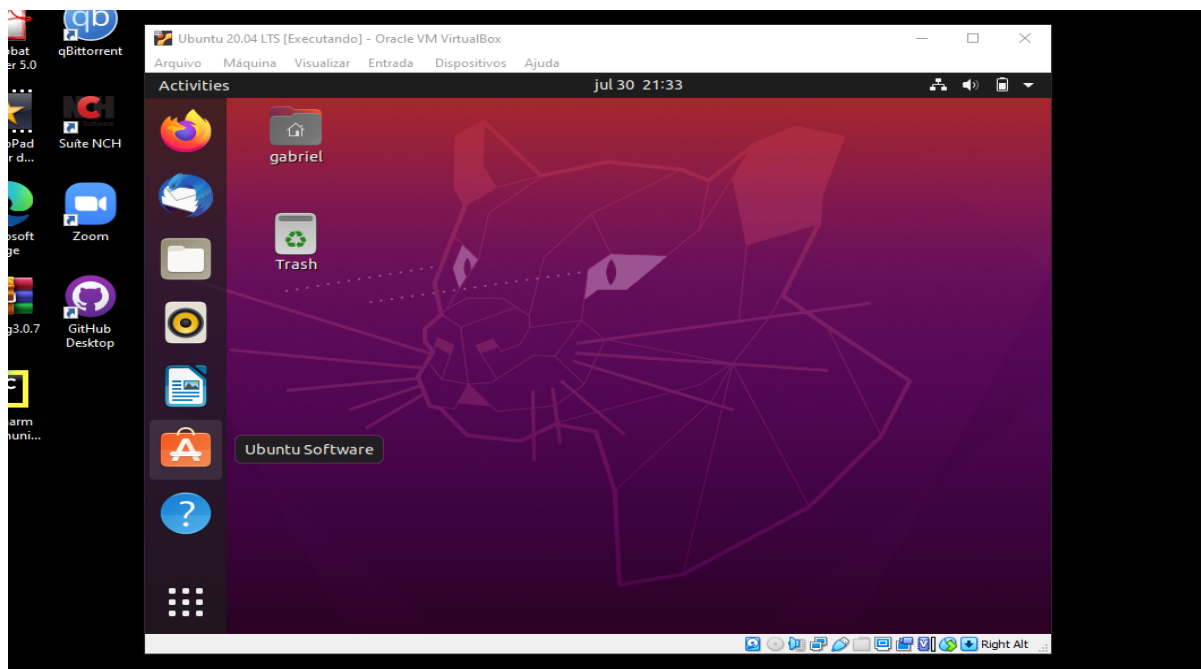
**Figura 8.**

O continue estará nessa parte circulada, porém não dá para o usuário visualizar. Em seguida sempre aperte em continuar, irá aparecer uma aba de login e lá preencha com suas informações, será feita uma instalação que irá demorar alguns minutos.



**Figura 9.**

Logo em seguida continue os passos, sempre dando next, então sua máquina estará pronta para uso.



**Figura 10.**

## 2. Execução do AutoRuns

Nesta parte do trabalho será apresentado o utilitário Autoruns, que foi pedido na atividade, o Autoruns é um utilitário capaz de listar os programas que estão configurados para serem executados durante o login ou inicialização do sistema. O aplicativo exibe a ordem dos processos carregados e pode ser configurado para exibir diversos itens.

Programas que estão configurados para iniciar durante a inicialização do sistema: **Figura 11.**

Autorun Entry	Description	Publisher	Image Path	Timestamp
HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\AlternateShell				07/12/2019 06:15
<input checked="" type="checkbox"/> cmd.exe	Processador de comandos do ... (Verified)		c:\windows\system32\cmd.exe	10/12/1953 23:58
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run				31/07/2021 15:20
<input checked="" type="checkbox"/> RtkAudUService	Realtek HD Audio Universal S... (Verified)	Realtek Semiconduct...	c:\windows\system32\rtkaudi...	23/12/2019 04:25
HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run				17/07/2021 16:57
<input checked="" type="checkbox"/> Dropbox	Dropbox	(Verified) Dropbox, Inc	c:\program files (x86)\dropbox\...	10/10/2016 21:33
<input checked="" type="checkbox"/> SunJavaUpdateS...	Java Update Scheduler	(Verified) Oracle America, Inc.	c:\program files (x86)\common ...	23/09/2016 00:38
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run				17/07/2021 17:06
<input checked="" type="checkbox"/> GoogleChromeAut...	Google Chrome	(Verified) Google LLC	c:\program files\google\chrom...	17/07/2021 21:07
<input checked="" type="checkbox"/> Web Companion	Web Companion	(Verified) LAVASOFT SOFTW...	c:\program files (x86)\lavasoft\...	03/12/2020 12:18
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run				21/07/2021 22:30
<input checked="" type="checkbox"/> es32	Processo de host do Windows ... (Verified)		c:\windows\system32\rundll32...	01/07/1967 13:25
<input checked="" type="checkbox"/> es64	Processo de host do Windows ... (Verified)		c:\windows\syswow64\rundll3...	26/08/2026 13:58
HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components				21/07/2021 22:21
<input checked="" type="checkbox"/> Google Chrome	Google Chrome Installer	(Verified) Google LLC	c:\program files\google\chrom...	17/07/2021 21:07
<input checked="" type="checkbox"/> n/a	Processo de host do Windows ... (Verified)		c:\windows\system32\rundll32...	01/07/1967 13:25
HKLM\SOFTWARE\Wow6432Node\Microsoft\Active Setup\Installed Components				21/07/2021 22:21
<input checked="" type="checkbox"/> n/a	Processo de host do Windows ... (Verified)		c:\windows\syswow64\rundll3...	26/08/2026 13:58

**Figura 11.**

Destes programas os programas não fornecidos pela microsoft são:

- Google Chrome
- Dropbox
- web companion
- Sun Java update

Estes são alguns programas que inicializam junto com o windows.

O que mais chama a atenção no Autoruns é que o programa permite que você altere as configurações de apenas alguns dos aplicativos menos importantes, mas bloqueia programas no sistema, como antivírus e rede, como medida de defesa protetora. O Autoruns, por outro lado se mostra uma ferramenta que permite controlar todos os cantos do processo de inicialização do windows, e isto é muito útil para deixar o sistema operacional mais rápido.



### 3. Execução do Process Explorer

Para este teste será usado o Process Explorer, este mostra-lhe todos os processos e tarefas em execução no seu sistema. Este programa apresenta informações detalhadas sobre todos os elementos, mostrando as bibliotecas e recursos em uso. Graças ao Process Explorer será possível saber o que está acontecendo no computador a qualquer momento, todavia também com ele é possível controlar o arranque no windows e terminar processos desnecessários ou mudar sua propriedades.

#### Processos que estão sendo executados no momento no sistema:

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
ACCStd.exe	0.19	70 K	10 K	12264		
ACCSvc.exe		2 K	5 K	3472	ACCSvc	Acer Incorporated
AdminService.exe	< 0.01	2 K	6 K	3428	Windows Setup API	Windows (R) Win 7 DDK p...
AppMonitorPlugIn.exe		23 K	25 K	2976	AppMonitorPlugIn	Acer Incorporated
audiodg.exe		8 K	15 K	8736		
Bridge_Service.exe		22 K	20 K	3564	Bridge_Service	GOTrustID Inc.
chrome.exe	< 0.01	164 K	210 K	3492	Google Chrome	Google LLC
chrome.exe		2 K	8 K	9444	Google Chrome	Google LLC
chrome.exe	< 0.01	331 K	242 K	10376	Google Chrome	Google LLC
chrome.exe	< 0.01	25 K	45 K	10388	Google Chrome	Google LLC
chrome.exe		8 K	18 K	10400	Google Chrome	Google LLC
chrome.exe		16 K	40 K	10692	Google Chrome	Google LLC
chrome.exe		87 K	104 K	10720	Google Chrome	Google LLC
chrome.exe		16 K	15 K	10768	Google Chrome	Google LLC
chrome.exe		29 K	46 K	10804	Google Chrome	Google LLC
chrome.exe		33 K	48 K	10812	Google Chrome	Google LLC
chrome.exe		40 K	36 K	11060	Google Chrome	Google LLC
chrome.exe		7 K	21 K	10640	Google Chrome	Google LLC
chrome.exe		217 K	154 K	10300	Google Chrome	Google LLC
chrome.exe		247 K	176 K	7744	Google Chrome	Google LLC
chrome.exe		200 K	171 K	8196	Google Chrome	Google LLC
chrome.exe		227 K	160 K	2472	Google Chrome	Google LLC
chrome.exe		77 K	64 K	8816	Google Chrome	Google LLC
chrome.exe		35 K	9 K	10016	Google Chrome	Google LLC
chrome.exe	< 0.01	379 K	21 K	8820	Google Chrome	Google LLC
chrome.exe		48 K	62 K	8936	Google Chrome	Google LLC
chrome.exe		63 K	101 K	11516	Google Chrome	Google LLC
chrome.exe		222 K	265 K	4532	Google Chrome	Google LLC
chrome.exe		24 K	56 K	12620	Google Chrome	Google LLC

CPU Usage: 4.19%   Commit Charge: 76.95%   Processes: 199   Physical Usage: 66.05%

**Figura 12.**

Para se descobri as DLL's foi tirado as informações do processo: explorer.exe. Essas foram as DLL's: **Figura 13.**

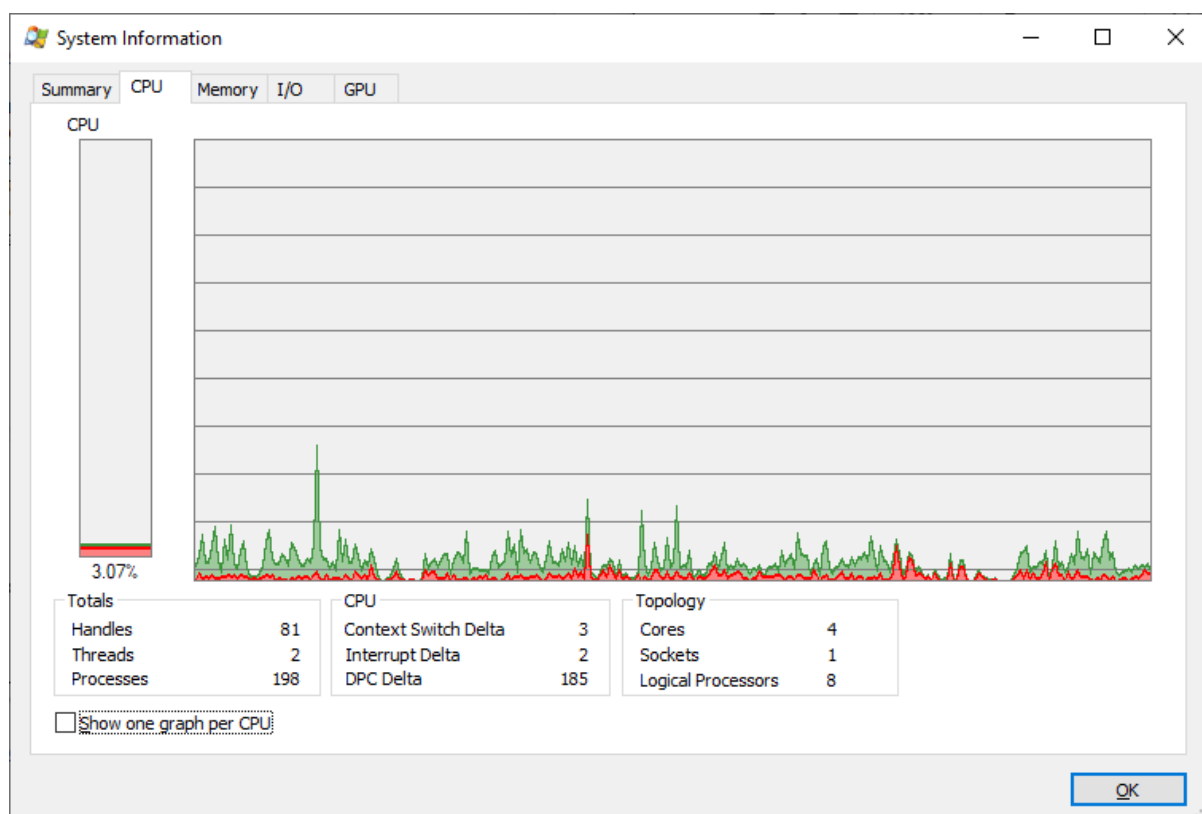


Figura 13.

Agora para casos de testes será usado o programa **Microsoft Excel**:

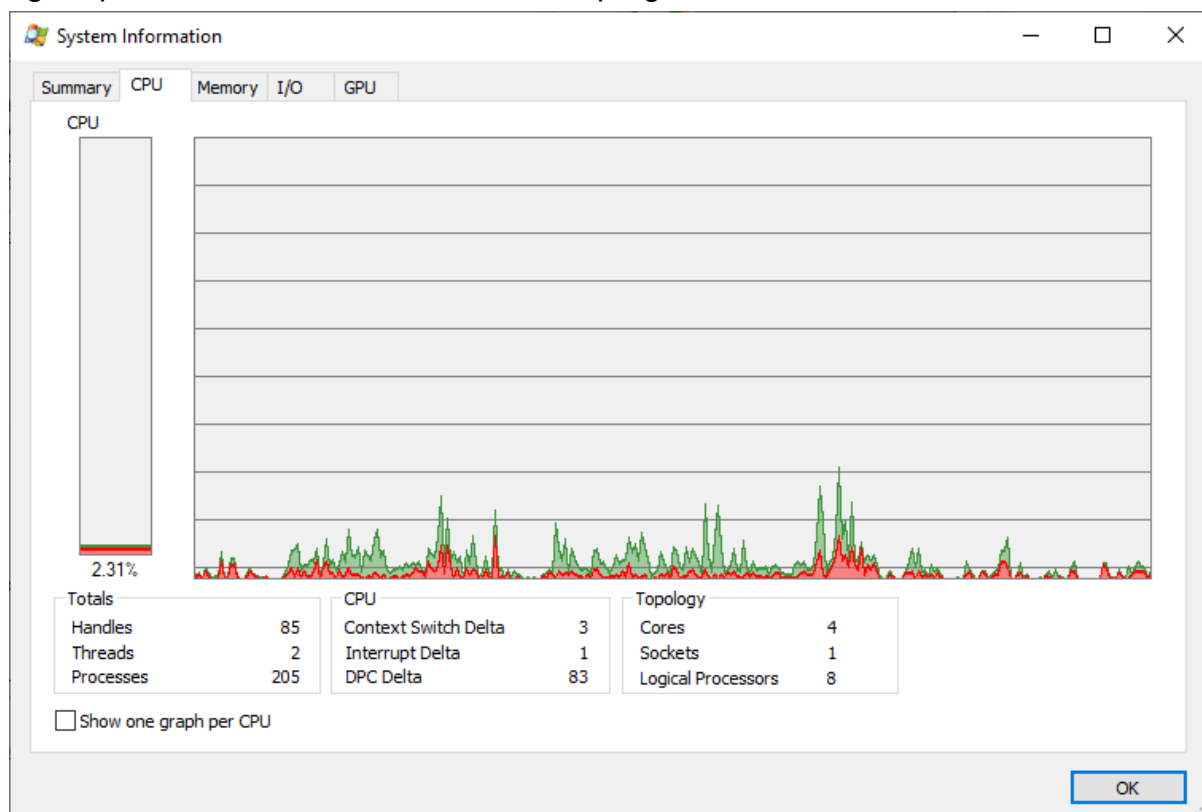
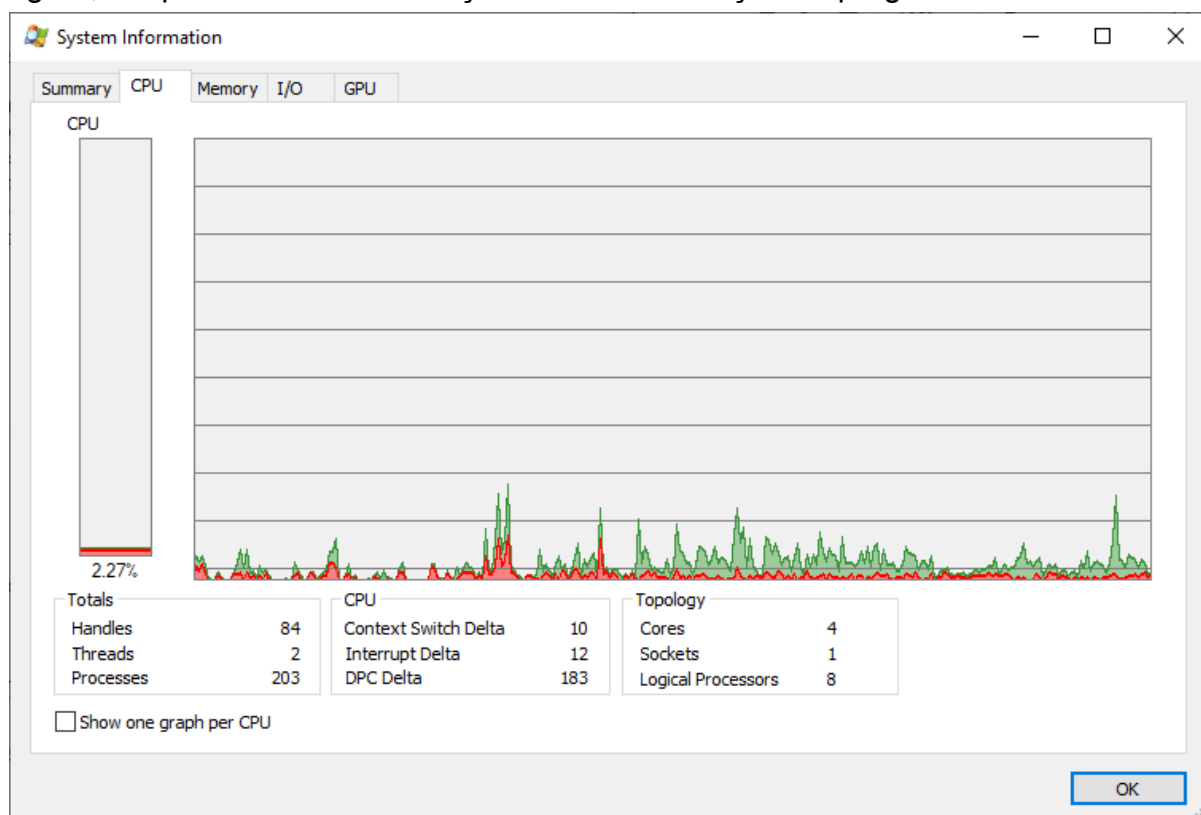


Figura 14.

Esses são todos os dados fornecidos pelo software Process Explorer um minuto após a inicialização do Excel.**Figura 14.**

Agora, um print com as informações com a finalização do programa:



**Figura 15.**

O que chamou a atenção no processo de inicialização e finalização do excel é como os elementos Context Switch Delta, Interrupt Delta e DPC Delta variarão de acordo com o início e o fim do programa, verifique na tabela abaixo o aumento dos mesmos:

	Início	Fim
Context Switch Delta	3	10
Interrupt Delta	1	12
DPC Delta	83	183

É perceptível que a CPU logo após o tempo passado teve um aumento em cada um dos seus elementos, ou seja, o processamento para o Excel aumentou consideravelmente de acordo com o tempo passado.

O mais interessante na ferramenta é o fato dela demonstrar todas as atividades em aberto no computador, e sua visão detalhada sobre cada programa

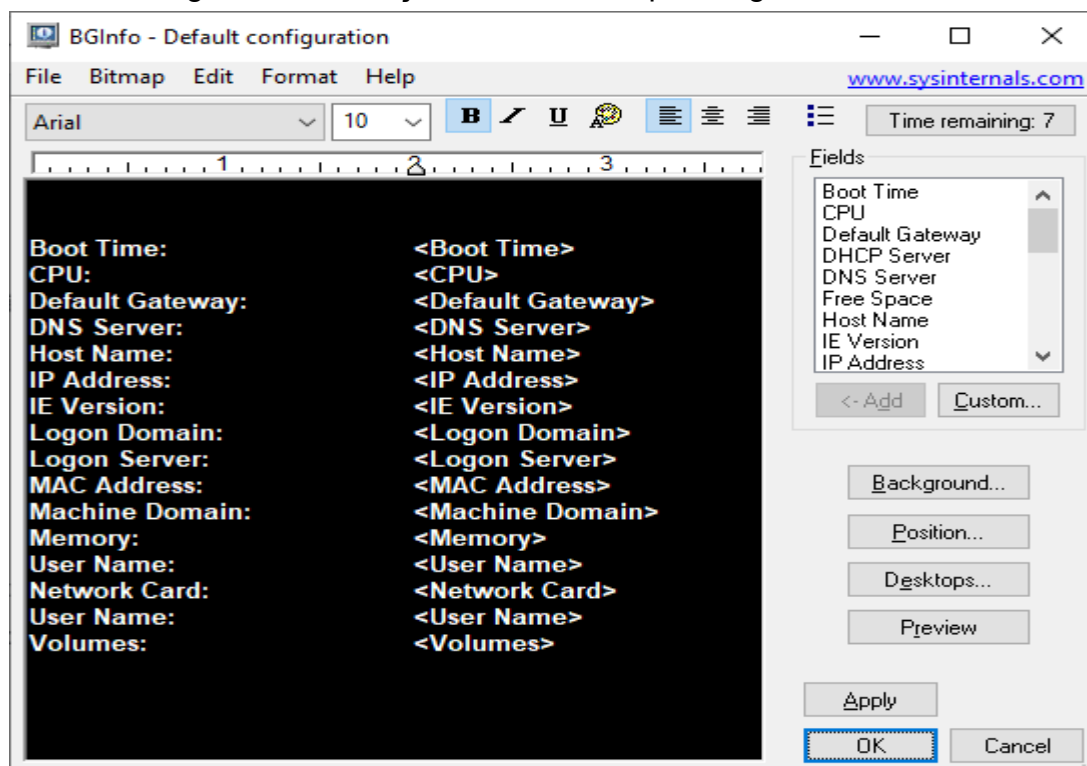
executado na máquina. Outrossim, um ponto bem importante é que os usuários conseguem monitorar e visualizar todos os processos que estão ativos, inativos e ocultos no sistema, além de poder reparar recursos que estão com problemas, verificar as DLLs que estão sendo utilizadas e acompanhar como está o consumo de memórias, CPU e GPU.

#### 4. Execução do Bginfo

O software escolhido foi o Bginfo, ele exibe automaticamente informações relevantes sobre um computador Windows no plano de fundo da área de trabalho, como o nome do computador, endereço IP, versão do service pack e muito mais. Além disso você pode editar qualquer campo, bem como as cores da fonte e do plano de fundo, e pode colocá-lo em sua pasta de inicialização para que seja executado a cada inicialização, ou mesmo configurá-lo para ser exibida como plano de fundo para a tela logon.

Outrossim, será mostrado como usar o Bginfo e automatizar as informações da área de trabalho, isto é bem simples basta baixar a ferramenta, e nela serão mostradas algumas informações que você vai poder deixá-las fixadas na tela do seu computador.

Essas são algumas informações das muitas que o Bginfo demonstra na tela:



**Figura 16.**

Como podem ver na **Figura 16**, o Bginfo permite que o usuário veja muitas informações do computador e as deixa na tela, como no exemplo da **Figura 17**.

```

Boot Time:          31/07/2021 15:20
CPU:                Quad 1.60 GHz Intel Core i5-10210U (Hyper-Threaded)
Default Gateway:    192.168.0.1
DNS Server:         (none)
                   (none)
                   (none)
                   (none)
                   192.168.0.1
Host Name:          LAPTOP-6PCHOA54
IP Address:         192.168.56.1
                   (none)
                   (none)
                   (none)
                   192.168.0.106
IE Version:         11.789.19041.0
Logon Domain:       MicrosoftAccount
Logon Server:
MAC Address:        0A-00-27-00-00-05
                   A6-63-A1-52-29-D2
                   B6-63-A1-52-29-D2
                   70-69-79-AB-B7-76
                   A4-63-A1-52-29-D2
Machine Domain:     WORKGROUP
Memory:             8014 MB
User Name:          gabri
Network Card:       VirtualBox Host-Only Ethernet Adapter
                   Microsoft Wi-Fi Direct Virtual Adapter #3
                   Microsoft Wi-Fi Direct Virtual Adapter
                   Realtek PCIe GbE Family Controller
                   Qualcomm Atheros QCA9377 Wireless Network Adapter

User Name:          gabri
Volumes:            C:\ 475,82 GB NTFS

```

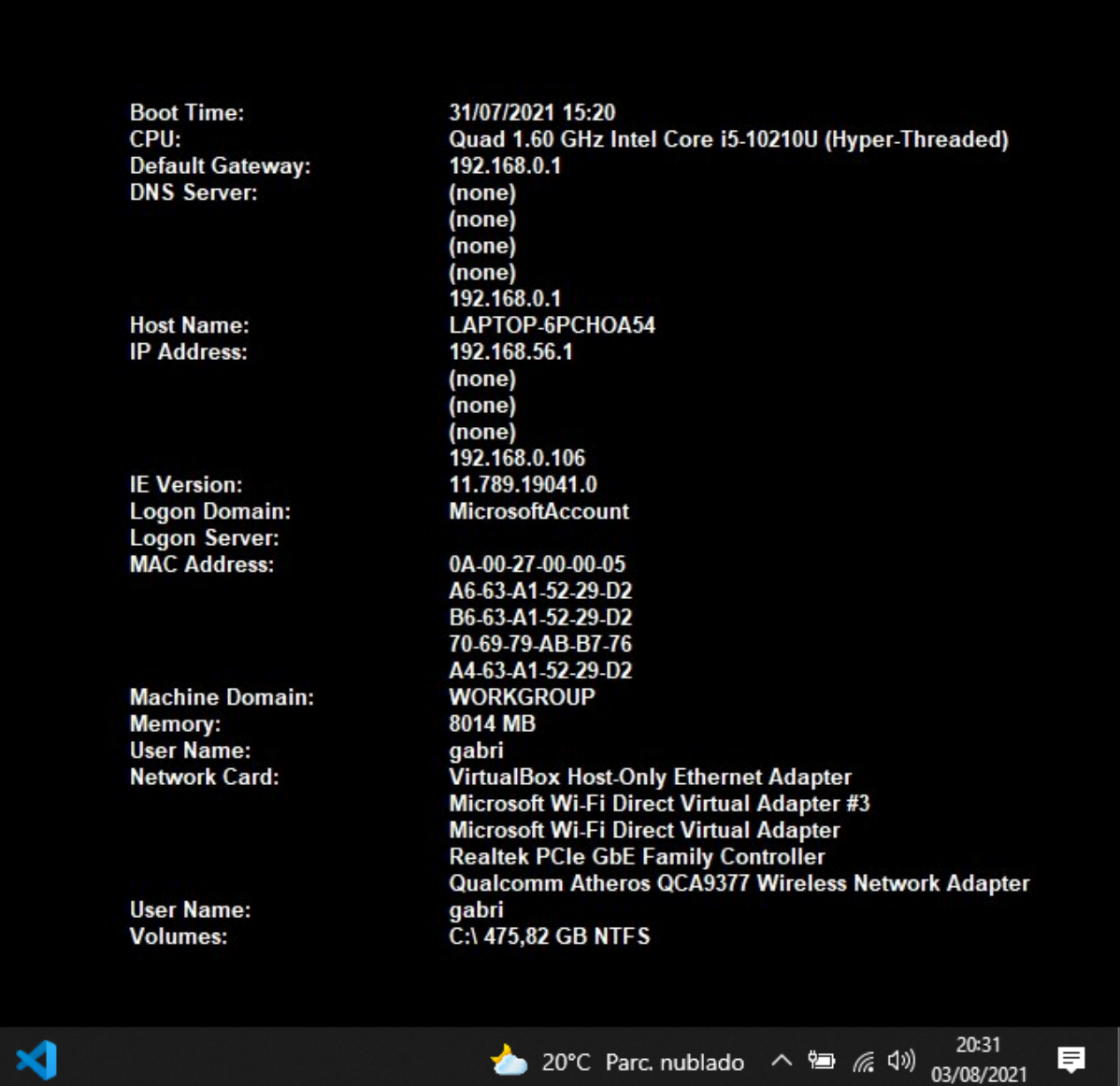


Figura 17.

## 5.Shell script

Foi solicitado pela atividade que fosse pesquisado sobre o Shell Script. Com a pesquisa foi feito o seguinte código usando bash:

```
#!/bin/bash
printf "who:\n\n"
who

printf "\nwhoami\n\n"
whoami

printf "\nuptime\n\n"
uptime #Quanto tempo o sistema está executando

printf "\nw\n\n"
w #O usuário que está logado no sistema

printf "\ndate\n\n"
date #Data atual

printf "\nhostname\n\n"
hostname #Nome do Host

printf "\nfree\n\n"
free #Mostra a quantidade de espaço livre
```

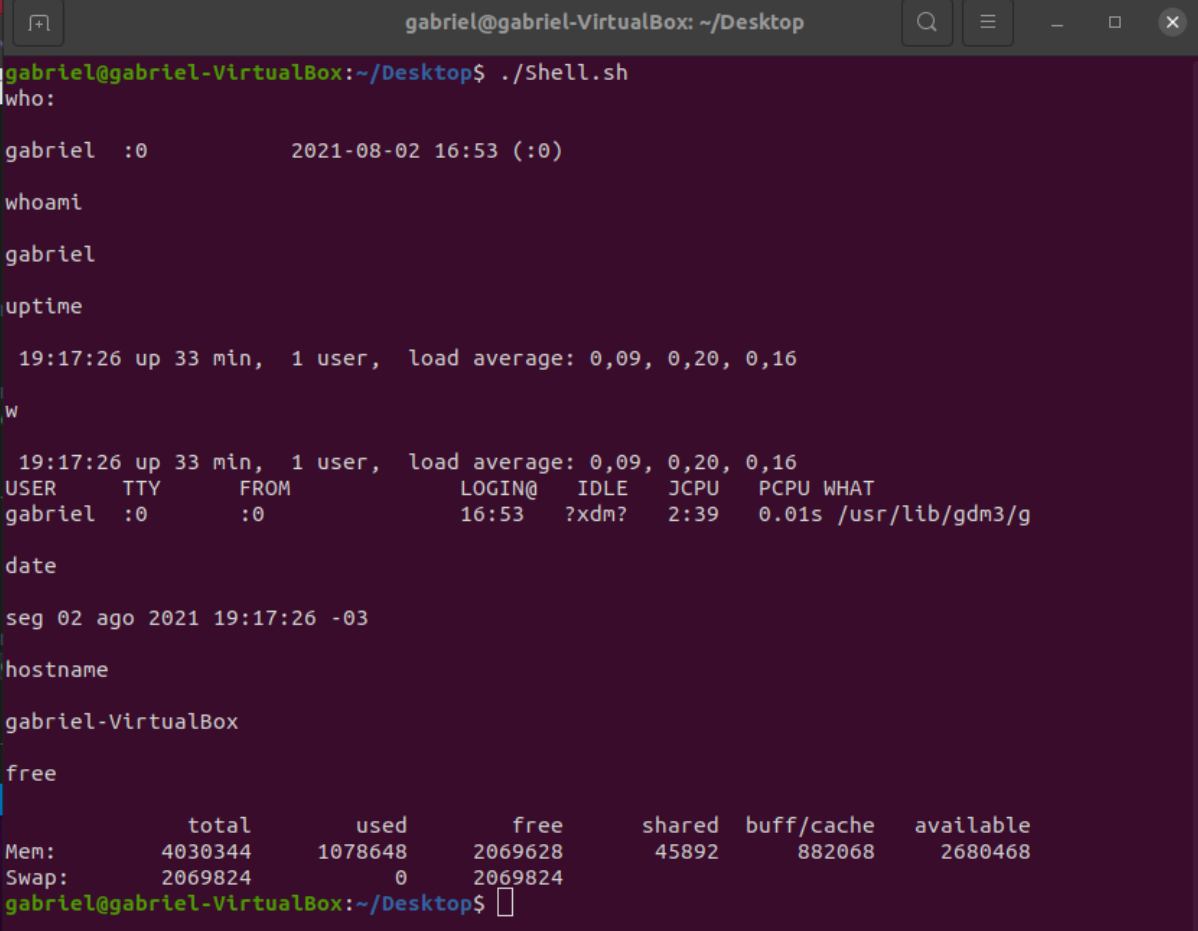
Foram pedidos os seguintes comandos:

- Who
- Whoami
- Date
- hostname

Além destes, foram adicionados os seguintes comandos:

- Uptime
- Free

E foi executado o comando **./Shell.sh**, sendo que Shell.sh é o nome do arquivo, e o seguinte resultado foi obtido, como mostrado na **Figura 18**.



```

gabriel@gabriel-VirtualBox: ~/Desktop
gabriel@gabriel-VirtualBox:~/Desktop$ ./Shell.sh
who:

gabriel  :0          2021-08-02 16:53 (:0)

whoami

gabriel

uptime

 19:17:26 up 33 min,  1 user,  load average: 0,09, 0,20, 0,16

w

 19:17:26 up 33 min,  1 user,  load average: 0,09, 0,20, 0,16
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU WHAT
gabriel   :0        :0            16:53    ?xdm?  2:39   0.01s /usr/lib/gdm3/g

date

seg 02 ago 2021 19:17:26 -03

hostname

gabriel-VirtualBox

free

          total        used        free      shared  buff/cache   available
Mem:      4030344       1078648       2069628         45892        882068       2680468
Swap:      2069824           0       2069824
gabriel@gabriel-VirtualBox:~/Desktop$

```

Figura 18.

## 6. Comando ps

Nesta parte do trabalho foi pedido que fosse usado o comando ps, o comando ps serve para listar os processo em execução. Então o comando ps é um comando extremamente robusto para te mostrar quem é o dono desse processo(user), o identificador do processo(PID) e qual é o processo em execução, e uma linha de comando onde esse processo foi executado. Então será executado o comando “ps -ef | more”, e “ps -aux|more” podemos ver abaixo o resultado da execução dos mesmos. Para fins de análise, iremos particionar a lista. Para analisarmos quais programas iniciam junto com o Linux, basta olharmos a hora que o programa foi iniciado e a hora do sistema. Observe na **Figura 19**. a discrepância dos horários, ou seja, processos como “3242”(PID) são executados quando o sistema já está carregado, e processos como “1” é executado na inicialização do sistema.

```

gabriel@gabriel-VirtualBox: ~
gabriel@gabriel-VirtualBox:~$ ps -ef | more
UID          PID    PPID  C STIME TTY          TIME CMD
root           1        0  0 19:36 ?        00:00:01 /sbin/init splash
root           2        0  0 19:36 ?        00:00:00 [kthreadd]
root           3        2  0 19:36 ?        00:00:00 [rcu_gp]
root           4        2  0 19:36 ?        00:00:00 [rcu_par_gp]
root           6        2  0 19:36 ?        00:00:00 [kworker/0:0H-events_highpri
]
root           9        2  0 19:36 ?        00:00:00 [mm_percpu_wq]
root          10        2  0 19:36 ?        00:00:00 [rcu_tasks_rude_]
root          11        2  0 19:36 ?        00:00:00 [rcu_tasks_trace]
root          12        2  0 19:36 ?        00:00:00 [ksoftirqd/0]
root          13        2  0 19:36 ?        00:00:00 [rcu_sched]
root          14        2  0 19:36 ?        00:00:00 [migration/0]
root          15        2  0 19:36 ?        00:00:00 [idle_inject/0]
root          16        2  0 19:36 ?        00:00:00 [cpuhp/0]
root          17        2  0 19:36 ?        00:00:00 [kdevtmpfs]
root          18        2  0 19:36 ?        00:00:00 [netns]
root          19        2  0 19:36 ?        00:00:00 [inet_frag_wq]
root          20        2  0 19:36 ?        00:00:00 [kauditd]
root          21        2  0 19:36 ?        00:00:00 [khungtaskd]
root          22        2  0 19:36 ?        00:00:00 [oom_reaper]
root          23        2  0 19:36 ?        00:00:00 [writeback]
root          24        2  0 19:36 ?        00:00:00 [kcompactd0]
root          25        2  0 19:36 ?        00:00:00 [ksmd]
root          26        2  0 19:36 ?        00:00:00 [khugepaged]
root          72        2  0 19:36 ?        00:00:00 [kintegrityd]
root          73        2  0 19:36 ?        00:00:00 [kblockd]
root          74        2  0 19:36 ?        00:00:00 [blkcg_punt_bio]
root          75        2  0 19:36 ?        00:00:00 [tpm_dev_wq]
root          76        2  0 19:36 ?        00:00:00 [ata_sff]
root          77        2  0 19:36 ?        00:00:00 [md]
root          78        2  0 19:36 ?        00:00:00 [edac-poller]
root          79        2  0 19:36 ?        00:00:00 [devfreq_wq]
root          80        2  0 19:36 ?        00:00:00 [watchdogd]
root          82        2  0 19:36 ?        00:00:00 [kworker/0:1H-kblockd]
root          84        2  0 19:36 ?        00:00:00 [kswapd0]
root          85        2  0 19:36 ?        00:00:00 [ecryptfs-kthrea]
root          87        2  0 19:36 ?        00:00:00 [kthreadd]

```

Figura 19. Execução do comando “ps - ef | more”



```

gabriel@gabriel-VirtualBox: ~
root      300      2  0 19:36 ?        00:00:00 [loop9]
root      319      2  0 19:36 ?        00:00:00 [cryptd]
systemd+  499      1  0 19:36 ?        00:00:00 /lib/systemd/systemd-resolved
systemd+  500      1  0 19:36 ?        00:00:00 /lib/systemd/systemd-timesyncd
root      535      1  0 19:36 ?        00:00:00 /usr/lib/accounts-service/accounts-daemon
root      536      1  0 19:36 ?        00:00:00 /usr/sbin/acpid
avahi     538      1  0 19:36 ?        00:00:00 avahi-daemon: running [gabriel-VirtualBox.local]
root      540      1  0 19:36 ?        00:00:00 /usr/sbin/cron -f
root      544      1  0 19:36 ?        00:00:00 /usr/sbin/cupsd -l
message+  545      1  0 19:36 ?        00:00:00 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile
n --syslog-only
root      546      1  0 19:36 ?        00:00:04 /usr/sbin/NetworkManager --no-daemon
root      562      1  0 19:36 ?        00:00:00 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-startup-triggers
root      564      1  0 19:36 ?        00:00:00 /usr/lib/policykit-1/polkitd --no-debug
syslog    572      1  0 19:36 ?        00:00:00 /usr/sbin/rsyslogd -n -iNONE
root      575      1  0 19:36 ?        00:00:01 /usr/lib/udev/udevadm settle
root      576      1  0 19:36 ?        00:00:00 /usr/libexec/switcheroo-control
root      577      1  0 19:36 ?        00:00:00 /lib/systemd/systemd-logind
root      583      1  0 19:37 ?        00:00:00 /usr/lib/udisks2/udisksd
root      584      1  0 19:37 ?        00:00:00 /sbin/wpa_supplicant -u -s -O /run/wpa_supplicant
avahi     590     538  0 19:37 ?        00:00:00 avahi-daemon: chroot helper
lp        613     544  0 19:37 ?        00:00:00 /usr/lib/cups/notifier/dbus dbus://
root      620      1  0 19:37 ?        00:00:00 /usr/sbin/cups-browsed
root      636      1  0 19:37 ?        00:00:00 /usr/sbin/ModemManager --filter-policy=strict
root      648      1  0 19:37 ?        00:00:00 /usr/sbin/gdm3
root      666      1  0 19:37 ?        00:00:00 /usr/bin/python3 /usr/share/unattended-upgrades/unattended-upgrade-st
gnal
whoopsie  704      1  0 19:37 ?        00:00:00 /usr/bin/whoopsie -f
kernoops  708      1  0 19:37 ?        00:00:00 /usr/sbin/kerneloops --test
kernoops  713      1  0 19:37 ?        00:00:00 /usr/sbin/kerneloops
rtkit     790      1  0 19:37 ?        00:00:00 /usr/libexec/rtkit-daemon
root      863      1  0 19:37 ?        00:00:00 /usr/lib/upower/upowerd
colord    1122     1  0 19:37 ?        00:00:00 /usr/libexec/colord
root      1154     2  0 19:37 ?        00:00:00 [kworker/0:7-events]
root      1234     2  0 20:17 ?        00:00:00 [kworker/u2:0-events_unbound]
root      1237     2  0 20:25 ?        00:00:00 [kworker/u2:2-events_power_efficient]
root      1247     2  0 20:40 ?        00:00:00 [kworker/u2:1-events_unbound]
root      1253     648  0 20:44 ?        00:00:00 gdm-session-worker [pam/gdm-password]

```

Figura 20. Execução do ‘ps -ef | more’, mais exemplos

```

gabriel@gabriel-VirtualBox: ~
gabriel@gabriel-VirtualBox:~$ ps -aux | more
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.2 102192 11604 ?        Ss   19:36   0:01 /sbin/init splash
root         2  0.0  0.0      0     0 ?        S    19:36   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        I<   19:36   0:00 [rcu_gp]
root         4  0.0  0.0      0     0 ?        I<   19:36   0:00 [rcu_par_gp]
root         6  0.0  0.0      0     0 ?        I<   19:36   0:00 [kworker/0:0H-events_highpri]
root         9  0.0  0.0      0     0 ?        I<   19:36   0:00 [mm_percpu_wq]
root        10  0.0  0.0      0     0 ?        S    19:36   0:00 [rcu_tasks_rude_]
root        11  0.0  0.0      0     0 ?        S    19:36   0:00 [rcu_tasks_trace]
root        12  0.0  0.0      0     0 ?        S    19:36   0:00 [ksoftirqd/0]
root        13  0.0  0.0      0     0 ?        I    19:36   0:00 [rcu_sched]
root        14  0.0  0.0      0     0 ?        S    19:36   0:00 [migration/0]
root        15  0.0  0.0      0     0 ?        S    19:36   0:00 [idle_inject/0]
root        16  0.0  0.0      0     0 ?        S    19:36   0:00 [cpuhp/0]
root        17  0.0  0.0      0     0 ?        S    19:36   0:00 [kdevtmpfs]
root        18  0.0  0.0      0     0 ?        I<   19:36   0:00 [netns]
root        19  0.0  0.0      0     0 ?        I<   19:36   0:00 [inet_frag_wq]
root        20  0.0  0.0      0     0 ?        S    19:36   0:00 [kauditd]
root        21  0.0  0.0      0     0 ?        S    19:36   0:00 [khungtaskd]
root        22  0.0  0.0      0     0 ?        S    19:36   0:00 [oom_reaper]
root        23  0.0  0.0      0     0 ?        I<   19:36   0:00 [writeback]
root        24  0.0  0.0      0     0 ?        S    19:36   0:00 [kcompactd0]
root        25  0.0  0.0      0     0 ?        SN   19:36   0:00 [ksmd]
root        26  0.0  0.0      0     0 ?        SN   19:36   0:00 [khugepaged]
root        72  0.0  0.0      0     0 ?        I<   19:36   0:00 [kintegrityd]
root        73  0.0  0.0      0     0 ?        I<   19:36   0:00 [kblockd]
root        74  0.0  0.0      0     0 ?        I<   19:36   0:00 [blkcg_punt_bio]
root        75  0.0  0.0      0     0 ?        I<   19:36   0:00 [tpm_dev_wq]
root        76  0.0  0.0      0     0 ?        I<   19:36   0:00 [ata_sff]
root        77  0.0  0.0      0     0 ?        I<   19:36   0:00 [md]
root        78  0.0  0.0      0     0 ?        I<   19:36   0:00 [edac-poller]
root        79  0.0  0.0      0     0 ?        I<   19:36   0:00 [devfreq_wq]
root        80  0.0  0.0      0     0 ?        S    19:36   0:00 [watchdogd]
root        82  0.0  0.0      0     0 ?        I<   19:36   0:00 [kworker/0:1H-kblockd]
root        84  0.0  0.0      0     0 ?        S    19:36   0:00 [kswapd0]
root        85  0.0  0.0      0     0 ?        S    19:36   0:00 [ecryptfs-kthrea]
root        87  0.0  0.0      0     0 ?        I<   19:36   0:00 [kthrotld]

```

Figura 21. Execução do comando “ps -aux|more”

Para a segunda parte dessa atividade, foi pedido para gerar quatro arquivos, root-processos-1, root-processos-2, meus-processos-1 e meus-processos-2 e compara-los. Comparando os dois processos root, podemos notar que alguns processos como [kworker/0:1-events], que só estão presentes no arquivo 1, são processos que originam de dependências de algum programa que foi aberto, ou seja, não são executadas na inicialização do sistema. Processo como [kworker/0:1-cgroup\_destroy] são executados na inicialização do sistema, e são fechados após algum tempo de uso. Já na análise de “meus-processos”, não houve um processo que exista na parte 2 que não existia na parte 1, porém, a recíproca não é verdadeira, isso nos mostra os processos que usuário abriu durante a utilização, intencional ou não.

Link para os 4 arquivos: [Link](#)

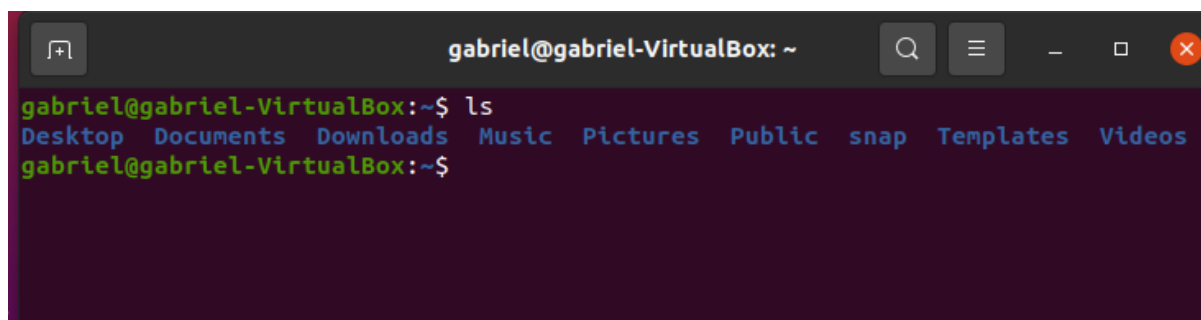
## 7. 5 Comandos para o Linux

Nesta parte do trabalho prático serão mostrados 5 comandos muito úteis para o sistema operacional Linux.

### 1) Comando ls:

Para listar os arquivos existentes em algum diretório, basta usar o comando “ls”. Se este comando for executado sem parâmetros, ele listará o conteúdo do diretório em que você se encontra. Mas você pode indicar uma caminho para ele, como ls/usr/bin, por exemplo.

Comando ls:

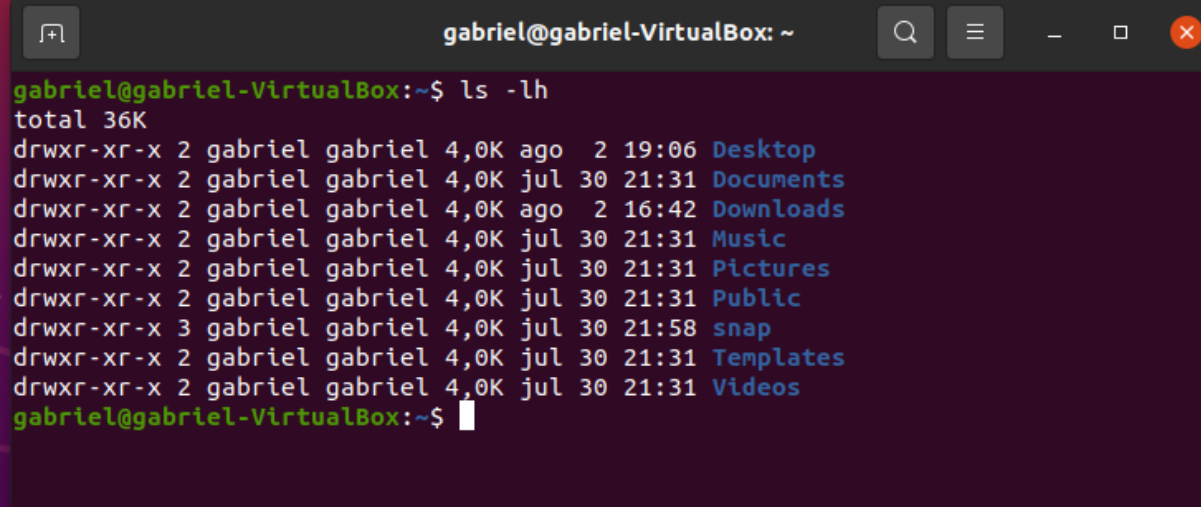
A screenshot of a terminal window titled 'gabriel@gabriel-VirtualBox: ~'. The prompt is 'gabriel@gabriel-VirtualBox:~\$'. The user has entered the command 'ls'. The output of the command is displayed in a colorized format: 'Desktop Documents Downloads Music Pictures Public snap Templates Videos'. The prompt is now 'gabriel@gabriel-VirtualBox:~\$'.

**Figura 22.**

Também é possível usar o ls para conferir o tamanho e a data da criação de cada arquivo ou pasta. Para isso, use o parâmetro -lh, como no exemplo: ls -lh. E se

você também quiser listar os arquivos ocultos, que começam com um ponto, use a opção `-a`(`ls -lha`).

Comando `ls -lh`:

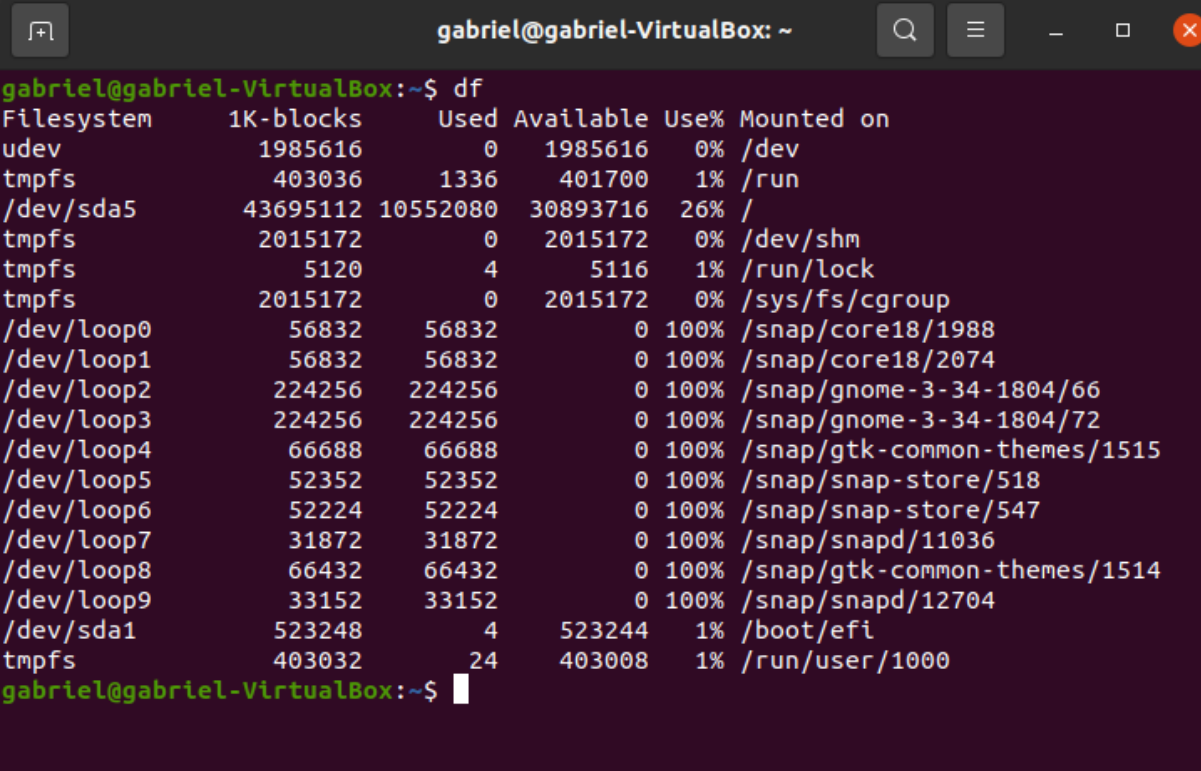


```
gabriel@gabriel-VirtualBox: ~
total 36K
drwxr-xr-x 2 gabriel gabriel 4,0K ago  2 19:06 Desktop
drwxr-xr-x 2 gabriel gabriel 4,0K jul 30 21:31 Documents
drwxr-xr-x 2 gabriel gabriel 4,0K ago  2 16:42 Downloads
drwxr-xr-x 2 gabriel gabriel 4,0K jul 30 21:31 Music
drwxr-xr-x 2 gabriel gabriel 4,0K jul 30 21:31 Pictures
drwxr-xr-x 2 gabriel gabriel 4,0K jul 30 21:31 Public
drwxr-xr-x 3 gabriel gabriel 4,0K jul 30 21:58 snap
drwxr-xr-x 2 gabriel gabriel 4,0K jul 30 21:31 Templates
drwxr-xr-x 2 gabriel gabriel 4,0K jul 30 21:31 Videos
gabriel@gabriel-VirtualBox:~$
```

Figura 23.

## 2) Comando `df`:

O comando `df`(display filesystem) exibem informações sobre o uso do espaço em disco de todos os sistema de arquivos montados.

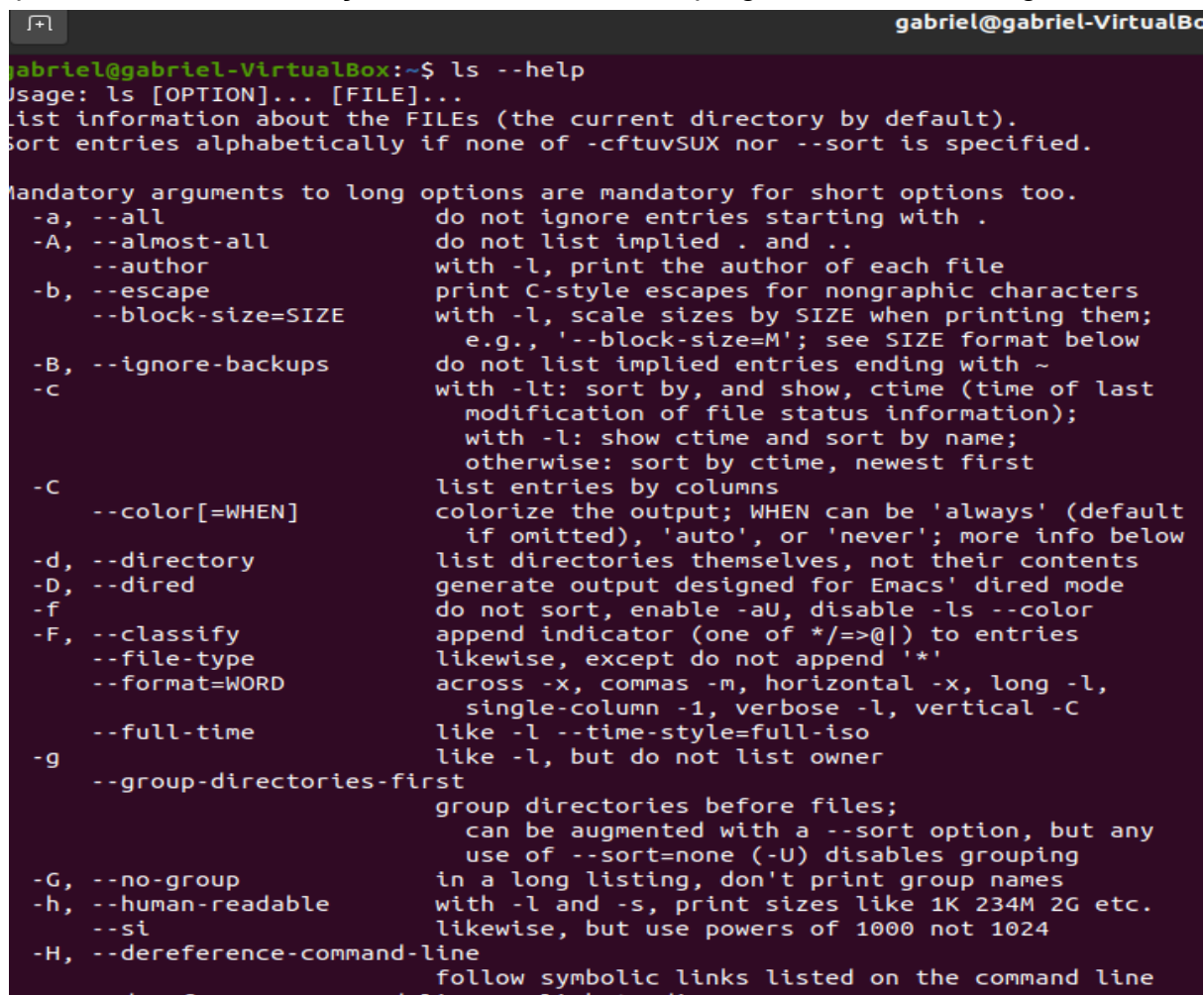


```
gabriel@gabriel-VirtualBox:~$ df
Filesystem      1K-blocks    Used Available Use% Mounted on
udev            1985616      0    1985616   0% /dev
tmpfs           403036      1336    401700   1% /run
/dev/sda5       43695112 10552080 30893716  26% /
tmpfs           2015172      0    2015172   0% /dev/shm
tmpfs           5120         4        5116   1% /run/lock
tmpfs           2015172      0    2015172   0% /sys/fs/cgroup
/dev/loop0      56832      56832      0 100% /snap/core18/1988
/dev/loop1      56832      56832      0 100% /snap/core18/2074
/dev/loop2      224256     224256      0 100% /snap/gnome-3-34-1804/66
/dev/loop3      224256     224256      0 100% /snap/gnome-3-34-1804/72
/dev/loop4      66688      66688      0 100% /snap/gtk-common-themes/1515
/dev/loop5      52352      52352      0 100% /snap/snap-store/518
/dev/loop6      52224      52224      0 100% /snap/snap-store/547
/dev/loop7      31872      31872      0 100% /snap/snapd/11036
/dev/loop8      66432      66432      0 100% /snap/gtk-common-themes/1514
/dev/loop9      33152      33152      0 100% /snap/snapd/12704
/dev/sda1       523248      4    523244   1% /boot/efi
tmpfs           403032      24    403008   1% /run/user/1000
gabriel@gabriel-VirtualBox:~$
```

Figura 24.

### 3) Comando help:

O comando `help` é utilizado para trazer informações de um determinado programa. No caso se você tiver dificuldade em algum comando é só usar o `--help` que ele trará as informações desse determinado programa como na imagem abaixo:



```

gabriel@gabriel-VirtualBox:~$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

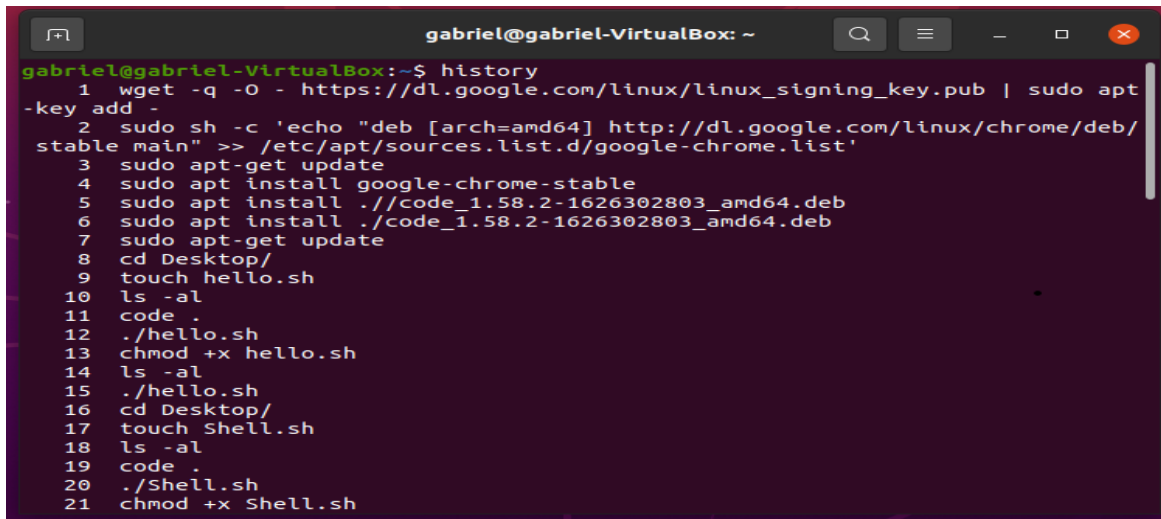
Mandatory arguments to long options are mandatory for short options too.
  -a, --all                do not ignore entries starting with .
  -A, --almost-all        do not list implied . and ..
  --author                 with -l, print the author of each file
  -b, --escape             print C-style escapes for nongraphic characters
  --block-size=SIZE        with -l, scale sizes by SIZE when printing them;
                           e.g., '--block-size=M'; see SIZE format below
  -B, --ignore-backups     do not list implied entries ending with ~
  -c                       with -lt: sort by, and show, ctime (time of last
                           modification of file status information);
                           with -l: show ctime and sort by name;
                           otherwise: sort by ctime, newest first
  -C                       list entries by columns
  --color[=WHEN]           colorize the output; WHEN can be 'always' (default
                           if omitted), 'auto', or 'never'; more info below
  -d, --directory          list directories themselves, not their contents
  -D, --dired              generate output designed for Emacs' dired mode
  -f                       do not sort, enable -aU, disable -ls --color
  -F, --classify           append indicator (one of */=>@|) to entries
  --file-type              likewise, except do not append '*'
  --format=WORD            across -x, commas -m, horizontal -x, long -l,
                           single-column -1, verbose -l, vertical -C
  --full-time              like -l --time-style=full-iso
  -g                       like -l, but do not list owner
  --group-directories-first
                           group directories before files;
                           can be augmented with a --sort option, but any
                           use of --sort=none (-U) disables grouping
  -G, --no-group           in a long listing, don't print group names
  -h, --human-readable     with -l and -s, print sizes like 1K 234M 2G etc.
  --si                    likewise, but use powers of 1000 not 1024
  -H, --dereference-command-line
                           follow symbolic links listed on the command line
  --dereference-command-line-symlink-to-dir
                           follow symbolic links to directories listed on the command line

```

Figura 25.

### 4) Comando history:

Quando você já estiver familiarizado com o Linux, vai perceber que você pode executar centenas de comando todos os dias. Por exemplo, o comando `history` (histórico) é particularmente útil se você quer rever quais comandos você usou antes. Exemplo:



```

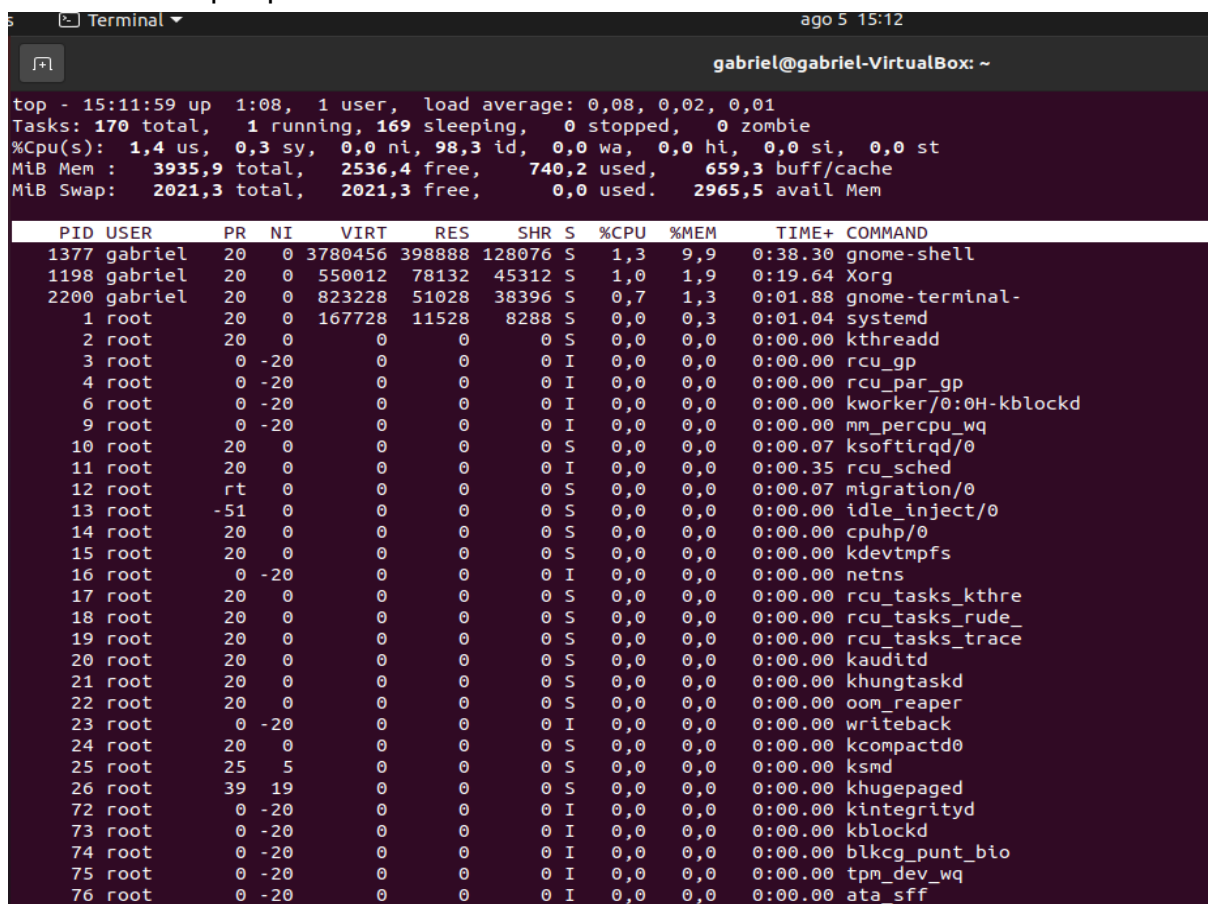
gabriel@gabriel-VirtualBox: ~
gabriel@gabriel-VirtualBox:~$ history
1  wget -q -O - https://dl.google.com/linux/linux_signing_key.pub | sudo apt
-key add -
2  sudo sh -c 'echo "deb [arch=amd64] http://dl.google.com/linux/chrome/deb/
stable main" >> /etc/apt/sources.list.d/google-chrome.list'
3  sudo apt-get update
4  sudo apt install google-chrome-stable
5  sudo apt install ./code_1.58.2-1626302803_amd64.deb
6  sudo apt install ./code_1.58.2-1626302803_amd64.deb
7  sudo apt-get update
8  cd Desktop/
9  touch hello.sh
10 ls -al
11 code .
12 ./hello.sh
13 chmod +x hello.sh
14 ls -al
15 ./hello.sh
16 cd Desktop/
17 touch Shell.sh
18 ls -al
19 code .
20 ./Shell.sh
21 chmod +x Shell.sh

```

Figura 26.

### 5) Comando top:

Equivalente ao gerenciador de tarefas do windows, o comando top vai mostrar uma lista de processos que estão em execução e o quanto de CPU cada processo usa de espaço no sistema, especialmente para saber qual processo deve ser encerrado porque ele consome muitos recursos.



```

top - 15:11:59 up 1:08, 1 user, load average: 0,08, 0,02, 0,01
Tasks: 170 total, 1 running, 169 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1,4 us, 0,3 sy, 0,0 ni, 98,3 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
MiB Mem : 3935,9 total, 2536,4 free, 740,2 used, 659,3 buff/cache
MiB Swap: 2021,3 total, 2021,3 free, 0,0 used. 2965,5 avail Mem

  PID USER      PR  NI   VIRT   RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 1377 gabriel   20   0 3780456 398888 128076 S   1,3   9,9   0:38.30 gnome-shell
 1198 gabriel   20   0 550012  78132 45312 S   1,0   1,9   0:19.64 Xorg
 2200 gabriel   20   0 823228  51028 38396 S   0,7   1,3   0:01.88 gnome-terminal-
    1 root      20   0 167728  11528  8288 S   0,0   0,3   0:01.04 systemd
    2 root      20   0          0          0 0 S   0,0   0,0   0:00.00 kthreadd
    3 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 rcu_gp
    4 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 rcu_par_gp
    6 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 kworker/0:0H-kblockd
    9 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 mm_percpu_wq
   10 root      20   0          0          0 0 S   0,0   0,0   0:00.07 ksoftirqd/0
   11 root      20   0          0          0 0 I   0,0   0,0   0:00.35 rcu_sched
   12 root      rt    0          0          0 0 S   0,0   0,0   0:00.07 migration/0
   13 root     -51   0          0          0 0 S   0,0   0,0   0:00.00 idle_inject/0
   14 root      20   0          0          0 0 S   0,0   0,0   0:00.00 cpuhp/0
   15 root      20   0          0          0 0 S   0,0   0,0   0:00.00 kdevtmpfs
   16 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 netns
   17 root      20   0          0          0 0 S   0,0   0,0   0:00.00 rcu_tasks_kthre
   18 root      20   0          0          0 0 S   0,0   0,0   0:00.00 rcu_tasks_rude_
   19 root      20   0          0          0 0 S   0,0   0,0   0:00.00 rcu_tasks_trace
   20 root      20   0          0          0 0 S   0,0   0,0   0:00.00 kauditd
   21 root      20   0          0          0 0 S   0,0   0,0   0:00.00 khungtaskd
   22 root      20   0          0          0 0 S   0,0   0,0   0:00.00 oom_reaper
   23 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 writeback
   24 root      20   0          0          0 0 S   0,0   0,0   0:00.00 kcompactd0
   25 root      25   5          0          0 0 S   0,0   0,0   0:00.00 ksmd
   26 root      39  19          0          0 0 S   0,0   0,0   0:00.00 khugepaged
   72 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 kintegrityd
   73 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 kblockd
   74 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 blkcg_punt_bio
   75 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 tpm_dev_wq
   76 root      0 -20          0          0 0 I   0,0   0,0   0:00.00 ata_sff

```

Figura 27.



## 8. Conceito da Disciplina

Nesta parte do trabalho será utilizado o conceito de chamadas de sistemas, mas especificamente na chamada “kill”, que envia um sinal para um processo ser encerrado. Veja na **Figura 28**, o processo com o PID “14044”, apos usarmos o comando abaixo, vemos que o processo foi corretamente encerrado.

```
gabriel 12506 0.0 0.8 504912 35568 ? Sl 16:35 0:00 update-notifier
gabriel 14014 1.2 8.5 4003840 343660 ? Ssl 16:47 0:11 /usr/bin/gnome-shell
gabriel 14032 0.0 0.2 393820 8612 ? Sl 16:47 0:00 ibus-daemon --panel disable --xim
gabriel 14036 0.0 0.1 245704 7552 ? Sl 16:47 0:00 /usr/libexec/ibus-dconf
gabriel 14037 0.0 0.7 282168 30672 ? Sl 16:47 0:00 /usr/libexec/ibus-extension-gtk3
gabriel 14040 0.0 0.6 206720 26964 ? Sl 16:47 0:00 /usr/libexec/ibus-x11 --kill-daemon
gabriel 14043 0.0 0.1 245648 7700 ? Sl 16:47 0:00 /usr/libexec/ibus-portal
gabriel 14074 0.0 0.6 2607708 26616 ? Sl 16:47 0:00 /usr/bin/gjs /usr/share/gnome-shell/org.gnome.Shell.Notifications
gabriel 14088 0.0 0.1 171868 7268 ? Sl 16:47 0:00 /usr/libexec/ibus-engine-simple
root 14173 0.0 0.0 0 0 ? I 16:48 0:00 [kworker/0:2-events]
```

**Figura 28.**

```
kill 14083
```

```
gabriel 12506 0.0 0.8 504912 35568 ? Sl 16:35 0:00 update-notifier
gabriel 14014 1.9 8.7 4003020 352684 ? Ssl 16:47 0:22 /usr/bin/gnome-shell
gabriel 14032 0.0 0.2 393820 8612 ? Sl 16:47 0:00 ibus-daemon --panel disable --xim
gabriel 14036 0.0 0.1 245704 7552 ? Sl 16:47 0:00 /usr/libexec/ibus-dconf
gabriel 14037 0.0 0.7 282168 30672 ? Sl 16:47 0:00 /usr/libexec/ibus-extension-gtk3
gabriel 14040 0.0 0.6 206720 26964 ? Sl 16:47 0:00 /usr/libexec/ibus-x11 --kill-daemon
gabriel 14074 0.0 0.6 2607708 26892 ? Sl 16:47 0:00 /usr/bin/gjs /usr/share/gnome-shell/org.gnome.Shell.Notifications
gabriel 14088 0.0 0.1 171868 7268 ? Sl 16:47 0:00 /usr/libexec/ibus-engine-simple
root 14173 0.0 0.0 0 0 ? I 16:48 0:00 [kworker/0:2-events]
root 14183 0.0 0.0 0 0 ? I 16:57 0:00 [kworker/u2:0-events_unbound]
```

**Figura 29.** Demonstração do processo que foi eliminado.

## 9. Conclusão

Neste trabalho, foram usado algumas ferramentas de gerenciamento em dois sistemas operacionais que ajudam em muito como identificar processos no sistema operacional. Ferramentas estas que ajudam a identificar processos que estão dando certa lentidão ao sistema e processos que inicializam com o sistema sem nenhuma necessidade.

Ademais, o uso de máquinas virtuais para a utilização de outros sistemas operacionais se mostrou muito útil, já que com alguns cliques se tinha acesso a dois sistemas operacionais, no caso do windows e linux. Todavia também, os comandos que foram pesquisados e demonstrados no linux, ajudaram a compreender os processos Linux.