

PEDRO HENRIQUE RESENDE RIBEIRO  
12011BCC004

ATIVIDADE PRÁTICA 01

## **INTRODUÇÃO AOS SISTEMAS OPERACIONAIS**



UNIVERSIDADE FEDERAL DE UBERLÂNDIA  
FACULDADE DE COMPUTAÇÃO  
2021

## SUMÁRIO

1 INTRODUÇÃO .....	1
2 SOLUÇÃO DA ATIVIDADE PRÁTICA .....	1
Questão 01 .....	1
Questão 02 .....	3
Questão 03 .....	4
Questão 04 .....	7
Questão 05 .....	9
Questão 06 .....	9
Questão 07 .....	11
Questão 08 .....	12
Questão 09 .....	13
Questão 10 .....	16
REFERÊNCIAS BIBLIOGRÁFICAS .....	19

## **1 INTRODUÇÃO**

Este trabalho foi desenvolvido em um notebook Dell Inspiron 3241, processador Intel Core i5-3337U de 1.80 GHz, 8 GB de memória RAM DDR3, 1 TB de HD, 120 GB de SSD e placa de vídeo Nvidia Geforce GT 625M de 1 GB de memória dedicada. A máquina possui dois sistemas operacionais instalados: Windows 10 Home Single Language (versão 21H1) e o Ubuntu 20.04.2.0 LTS.

## **2 SOLUÇÃO DA ATIVIDADE PRÁTICA**

**01) Ao ligar o computador, busque identificar as etapas de execução do POST, Boot Manager, Boot Loader e inicialização do sistema operacional.**

De acordo com Coutinho (2010), existem três firmwares principais na memória ROM dos computadores: BIOS (Basic Input Output System), POST (Power On System Test) e o SETUP (configurações do computador).

O POST verifica se componentes como processador, memória, disco, dentre outros, estão funcionando adequadamente. Caso exista alguma falha no sistema computacional, uma mensagem de erro é mostrada ao usuário. Caso contrário, o usuário tem a opção de acessar o SETUP da máquina.

No SETUP, o usuário tem à sua disposição diversas informações e configurações da máquina como temperatura de funcionamento da CPU, rotação dos coolers, tensão do processador, opções de boot, dentre outras (GARRETT, 2015).

Em seguida, se houver mais de um sistema operacional presentes, o Boot Manager é responsável por mostrar para o usuário os sistemas disponíveis e dar a opção de escolha do sistema a ser iniciado. É importante ressaltar que, mesmo que exista mais de um SO na máquina, sem a presença do Boot Manager, um sistema iniciará por padrão ou pode ocorrer de nenhum sistema ser iniciado.

Após o Boot Manager, o Boot Loader é inicializado. Ele é responsável por encontrar a partição na qual o sistema operacional está presente no disco e, em seguida, carregar o SO na memória do computador (COUTINHO, 2010). Após essa etapa, o sistema operacional entra em funcionamento e assume o controle dos processos da máquina.

A máquina utilizada para a atividade prática não mostra explicitamente a etapa do POST. Ao ligar o notebook, é possível ver a logo do fabricante (no caso a Dell) e a mensagem indicando qual tecla pressionar para acessar o SETUP. A Figura 1 ilustra o que foi descrito.

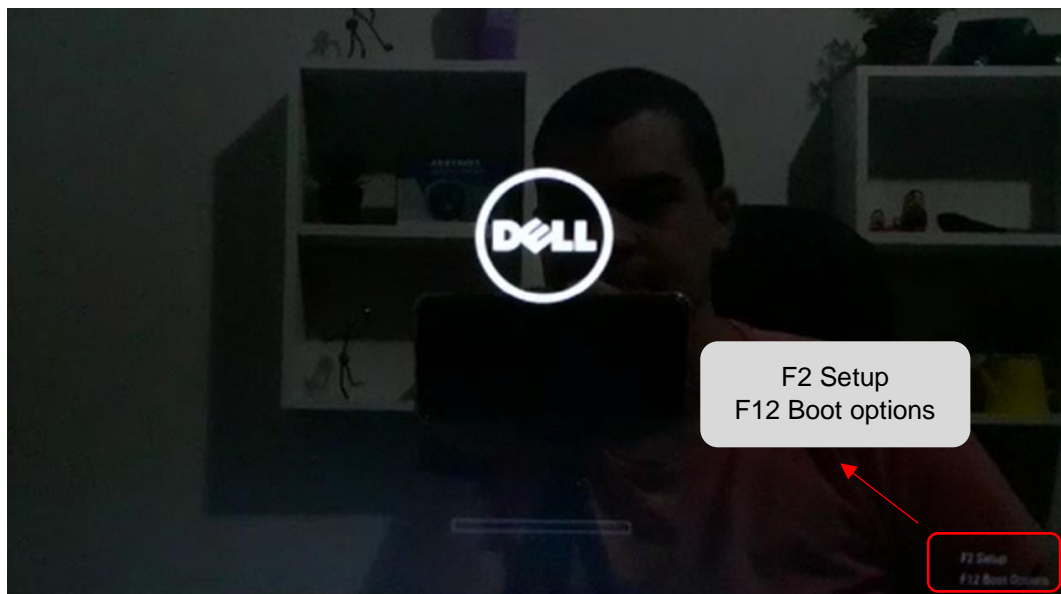


Figura 1 – Etapa do POST e teclas para acesso do SETUP

Em seguida, o Boot Manager é iniciado, conforme mostrado na Figura 2.

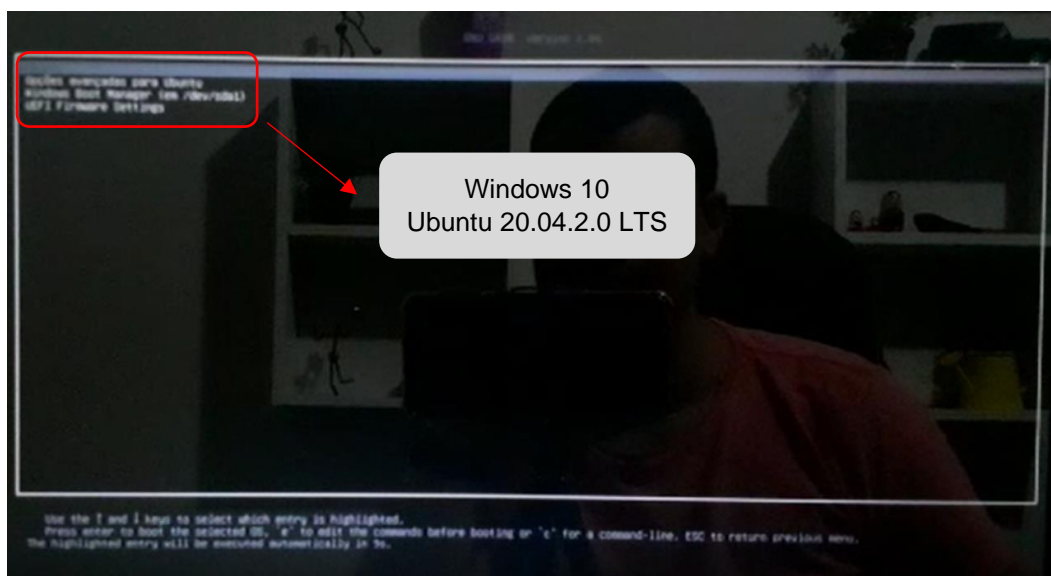


Figura 2 – Boot Manager

O Boot Manager utilizado é o GRUB 2.04. Para a máquina em questão, os sistemas operacionais instalados são o Windows 10 (versão 21H1) e Ubuntu 20.04.2.0 LTS.

Escolhendo a inicialização do Windows, aparece novamente uma tela semelhante a da Figura 1 e, em seguida, aparece a tela inicial do SO. A Figura 3 ilustra o que foi descrito.



Figura 3 – Tela inicial do Windows 10

**02) Ao ligar o computador, busque identificar o fabricante e a data do firmware do seu computador.**

A Figura 4 ilustra a tela inicial do SETUP do notebook.

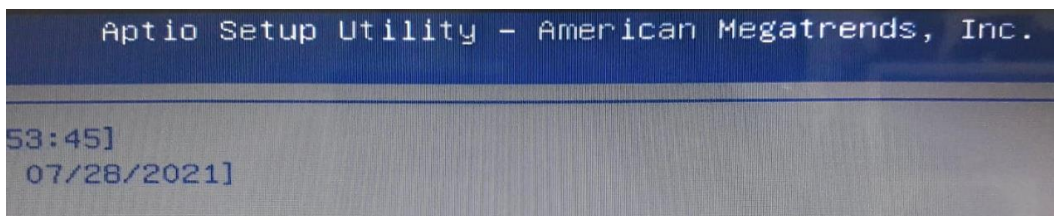


Figura 4 – Fabricante do firmware do notebook

De acordo com o site da Dell, a versão do firmware é 1.04, A13, cuja data da versão é 26/07/2011 (o notebook foi comprado em 2013).

**03) Ligue o computador e acesse o programa SETUP (firmware). Navegue nas opções disponíveis e se familiarize com elas. Faça isso em computadores diferentes para ver as diferenças de funcionalidades entre o firmware de diferentes fabricantes e versões.**

As Figura 5 a 9 mostram as opções presentes no SETUP do notebook Dell Inspiron 3421. Na página principal do SETUP (Figura 5) aparecem informações como hora e data, versão do BIOS, modelo do notebook, etiqueta de serviço (fornece a opção de download de drivers no site da Dell) e diversas outras relacionadas, por exemplo, à memória e a CPU.

Na aba “Advanced” (Figura 6) aparecem opções sobre teste de velocidade do processador, virtualização, sobre as portas USB e o modo de operação SATA. Na aba “Security” (Figura 7) aparecem opções sobre senhas do sistema. Na aba “Boot” (Figura 8) aparecem opções sobre a ordem do boot da máquina, boot em modo seguro, escolha do modo Legacy ou UFI. Por fim, a aba “Exit” (Figura 9) apresenta opções para salvar as configurações realizadas, descartar mudanças ou restaurar as opções padrão de fábrica.

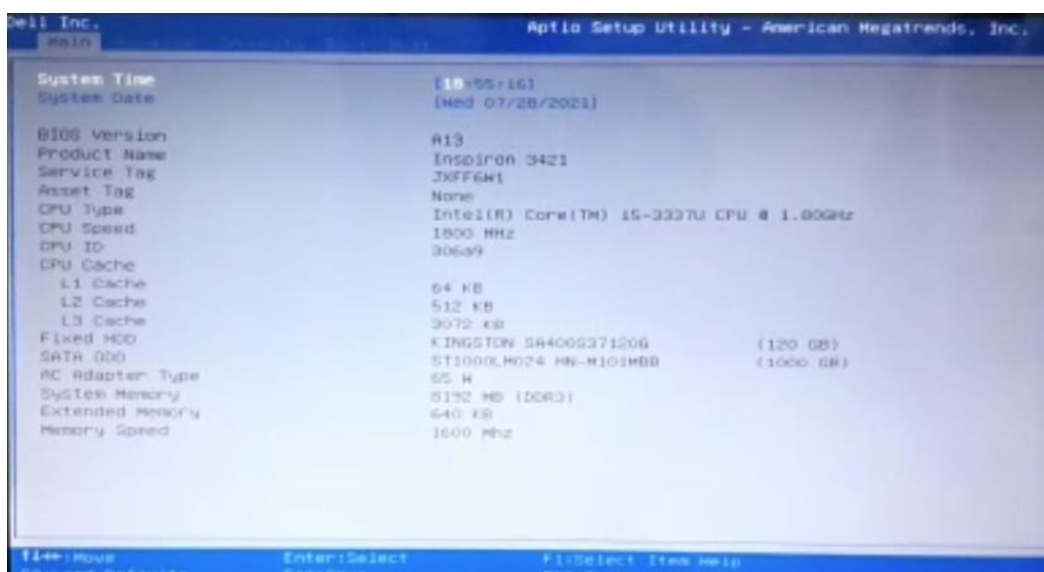


Figura 5 – Página principal do SETUP

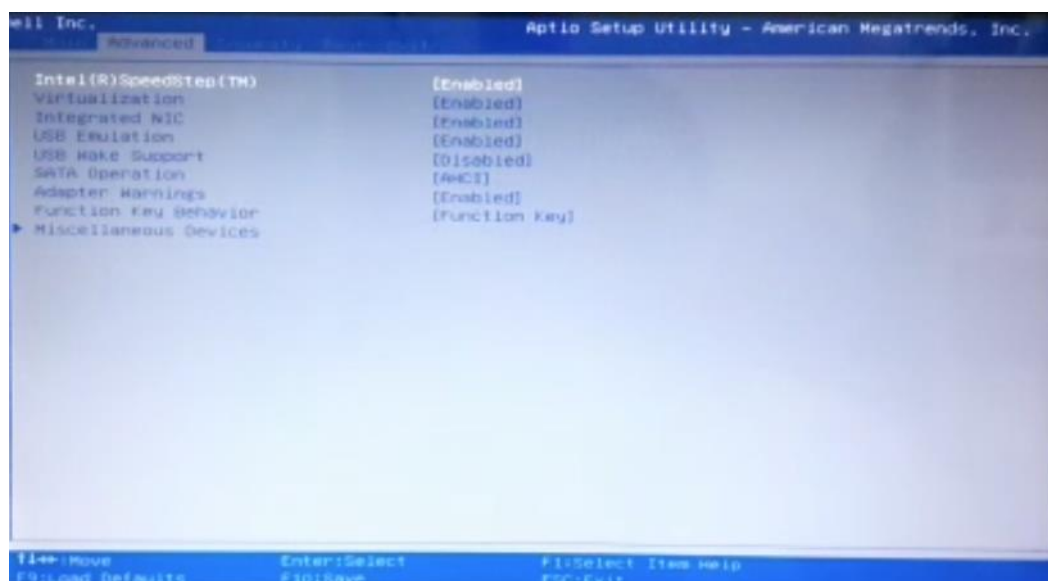


Figura 6 – Aba de configurações avançadas do SETUP

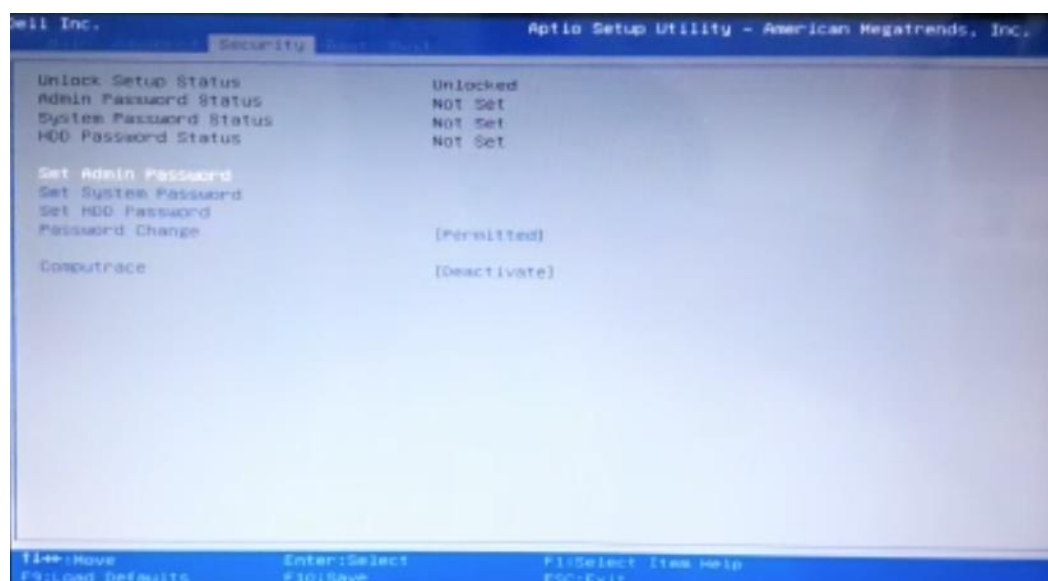


Figura 7 – Aba de configurações de segurança do SETUP

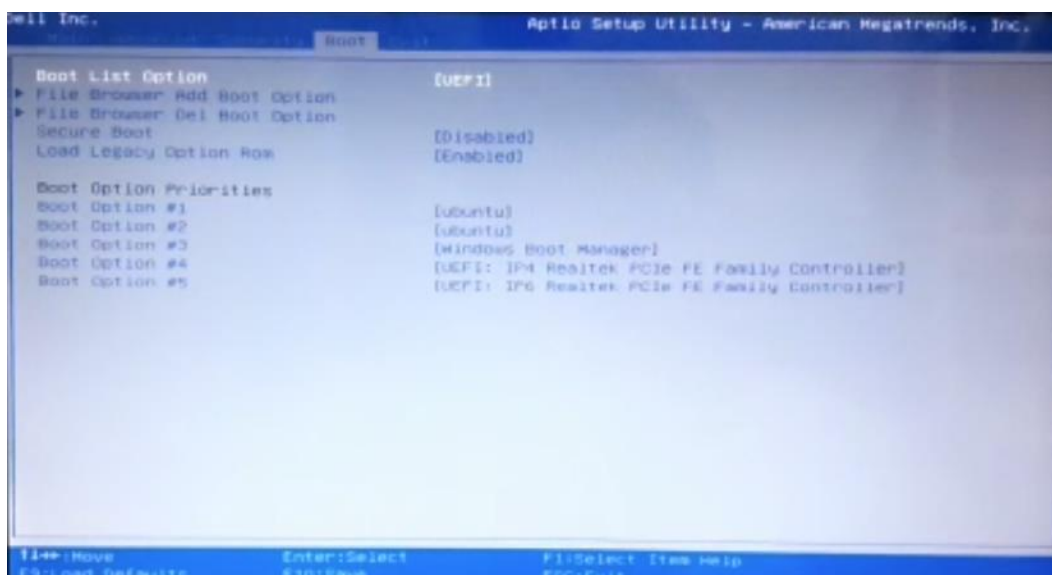


Figura 8 – Aba de configurações de boot do SETUP

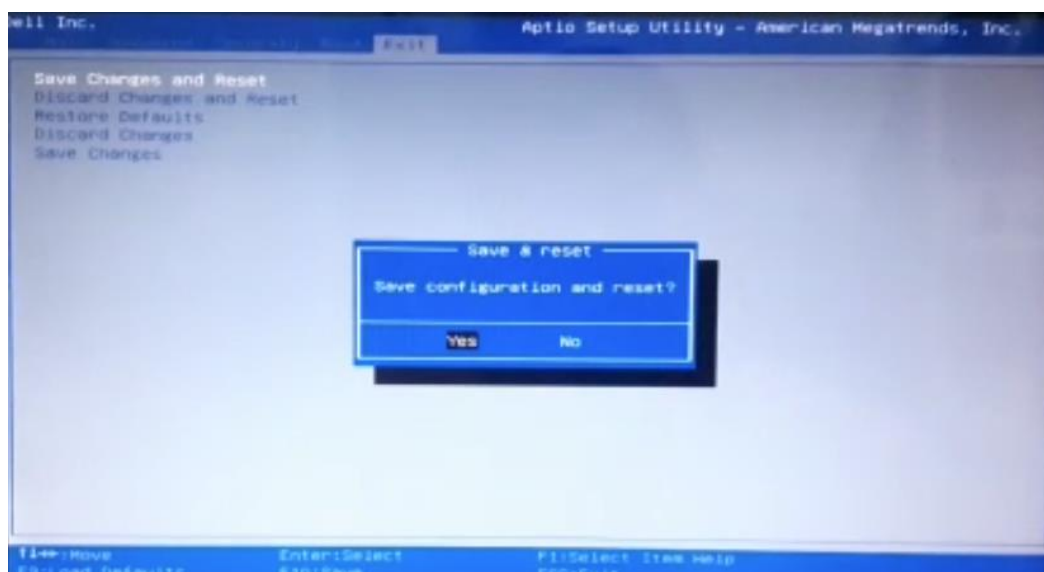


Figura 9 – Aba “Exit” do SETUP

O segundo computador utilizado para analisar o SETUP foi um Dell Inspiron Gaming Edition i15-7567. A máquina possui um processador Intel Core i5-7300HQ, 8 GB de memória RAM DDR4, 1 TB de HD e uma placa de vídeo Nvidia Geforce GTX 1050 com 4 GB de memória GDDR5. A Figura 10 ilustra o SETUP.



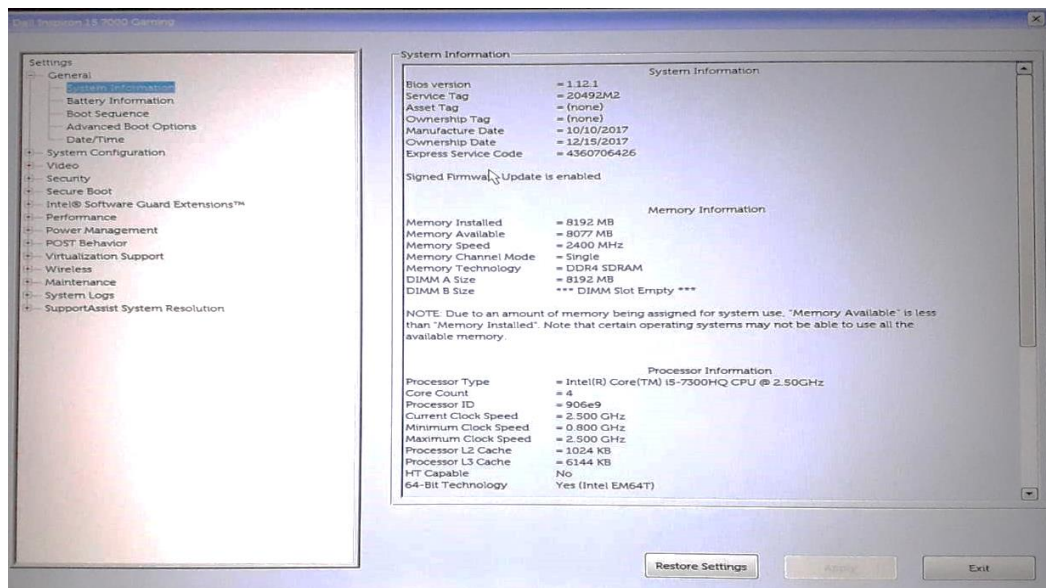


Figura 10 – Outro exemplo de SETUP

Como é possível notar, a interface do SETUP é bem mais atualizada (fabricação em 2017) e conta com o recurso de seleção através do mouse. Esse segundo modelo contém as mesmas configurações citadas para o SETUP do Dell Inspiron 3421, porém possui uma série de configurações extras.

**04) Ligue o computador e identifique a configuração (marca, modelo, capacidade, quantidade instalada) de processadores, memória principal e memória secundária. Para isso, acesse as informações do firmware SETUP e também as ferramentas que acompanham o sistema operacional.**

De acordo com o descrito na introdução e na descrição da Figura 5 (página principal do SETUP), as configurações do computador são as seguintes:

- Processador Intel Core i5-3337U de 1.80 GHz, 2 núcleos e 4 threads
- HD Samsung ST1000LM024 HN-M101MBB de 1 TB
- SSD Kingston SA400S37120G de 120 GB
- Memória RAM Dell (2 x 4 GB) DDR3 1600 MHz
- Placa gráfica Nvidia Geforce GT 625M com 1 GB de memória GDDR3

As Figuras 11 e 12 ilustram ferramentas do sistema operacional que permitem adquirir informações a respeito dos componentes citados acima.

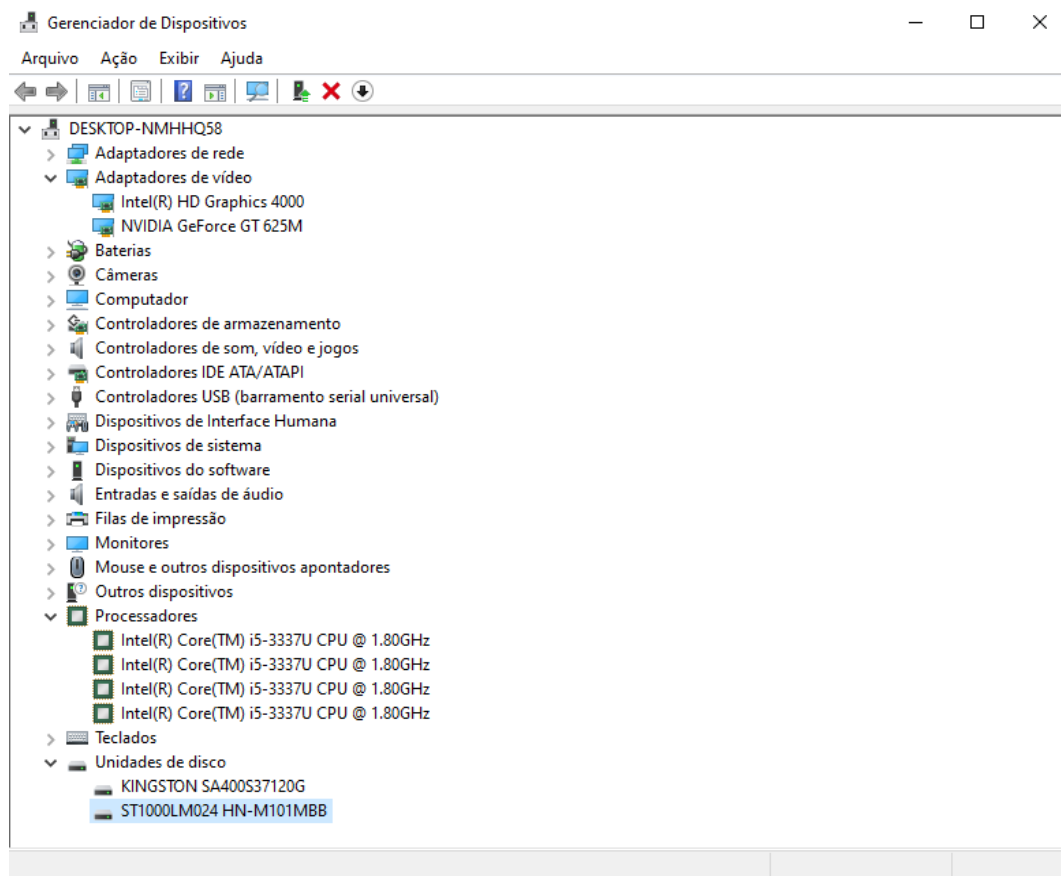


Figura 11 – Gerenciador de dispositivos do Windows 10

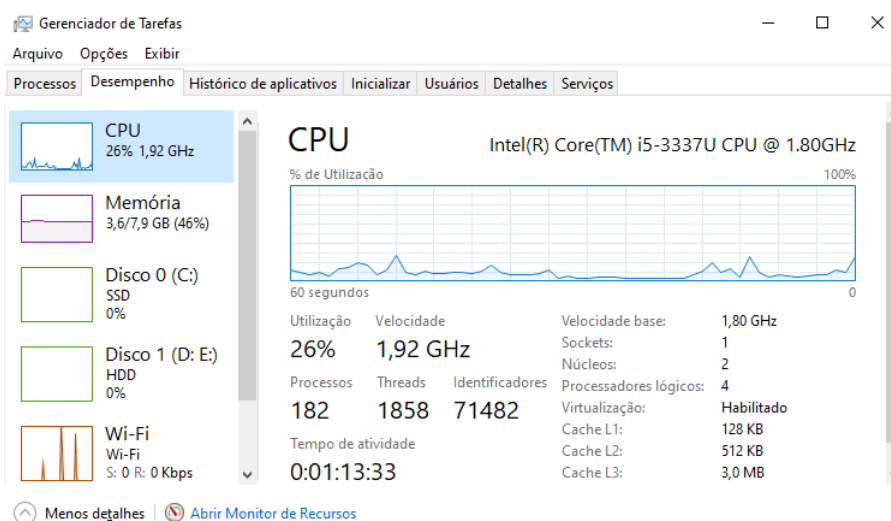


Figura 12 – Aba “Desempenho” do gerenciador de tarefas do Windows 10

**05) Verifique se o sistema operacional do seu computador é multiusuário. Em caso positivo, crie dois novos usuários (user1 e user2). Acesse esse sistema, simultaneamente, com esses dois usuários. Para isso, utilize o login pelo console e o login remoto.**

O Windows 10 possui o suporte para multiusuários em algumas versões do SO. A que está instalada no notebook (Windows 10 Home Single Language – versão 21H1) não possui o recurso de multiusuários. De acordo com o site da Microsoft, apenas as versões Enterprise ou Pro possuem o recurso. A Figura 13 mostra a mensagem do SO informando que não é possível usar o recurso.

## Área de Trabalho Remota

A Área de Trabalho Remota permite conectar-se a e controlar este computador a partir de um dispositivo remoto usando um cliente da Área de Trabalho Remota (disponível para Windows, Android, iOS e macOS). Você poderá trabalhar em outro dispositivo como se estivesse trabalhando diretamente neste computador.

Sua edição Home do Windows 10 não tem suporte para a Área de Trabalho Remota.



Saiba como fazer upgrade da sua edição do Windows 10

Figura 13 – Informação sobre o recurso multiusuário

**06) Utilizando o programa shell, em linha de comando, execute diferentes programas no seu computador (exemplo: notepad.exe, gedit, etc.). Faça isso tanto no Linux quanto no Windows.**

Para abrir o notepad utilizando o prompt de comandos do Windows, basta digitar “notepad” sem aspas na linha de comando e pressionar enter. A Figura 14 mostra o processo descrito.

No Linux, para abrir o gedit, basta abrir o terminal do Linux e digitar o comando “gedit” sem aspas na linha de comando e pressionar enter. A Figura 15 mostra o processo descrito.

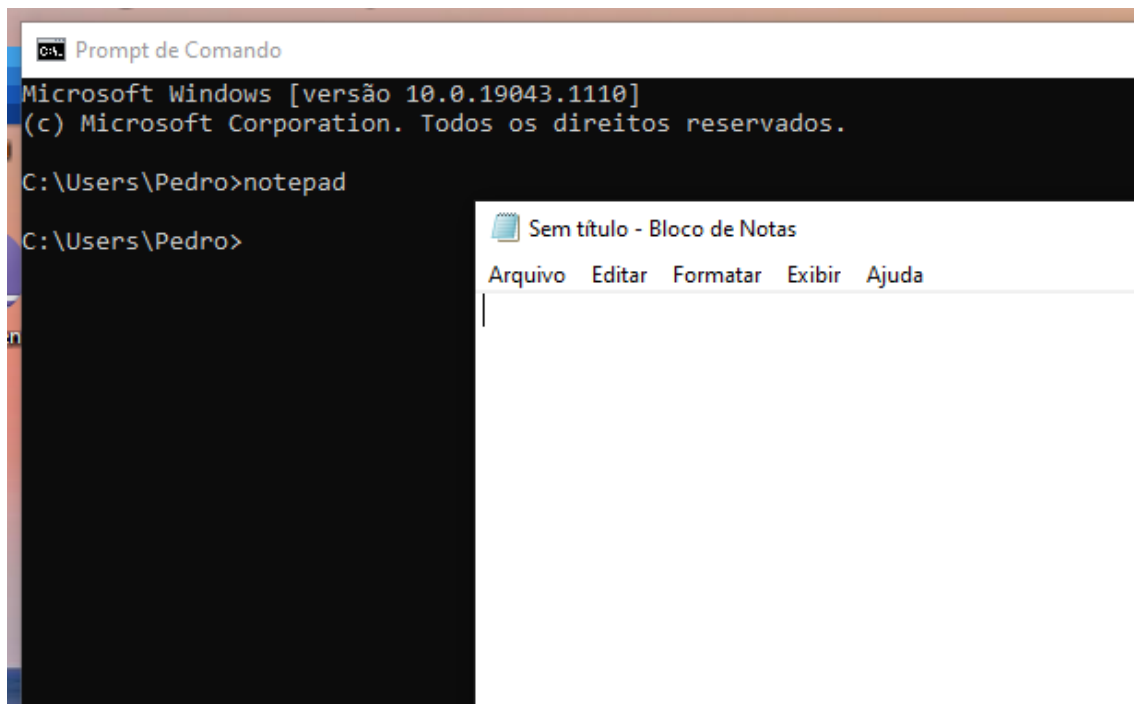


Figura 14 – Comando no shell para abrir o notepad

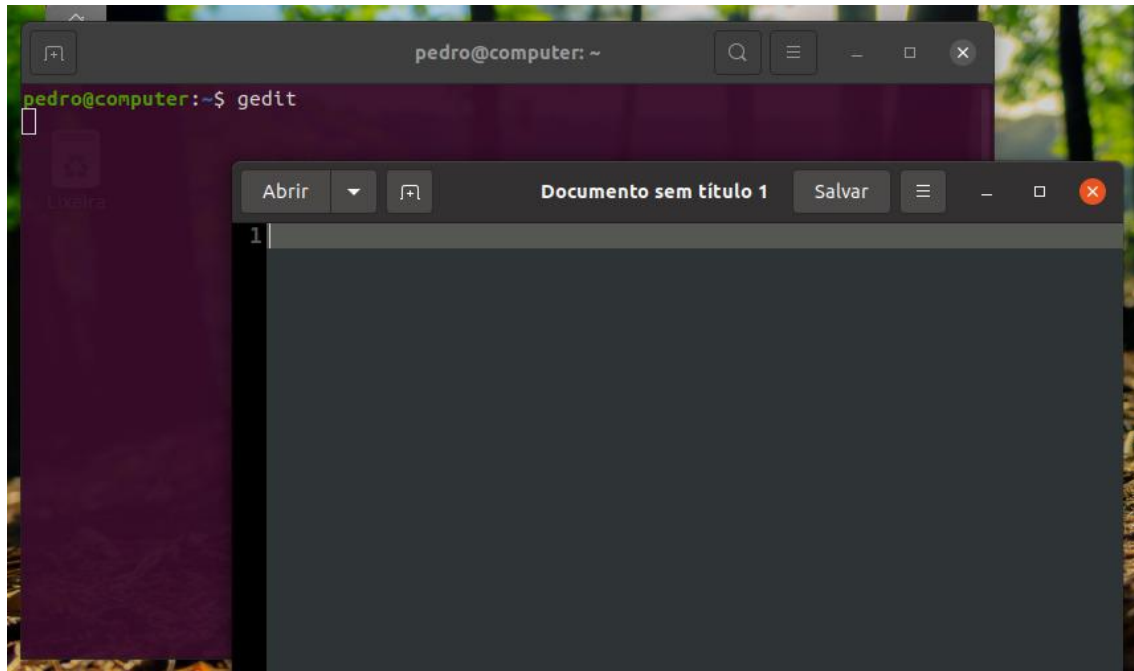


Figura 15 – Comando no bash para abrir o gedit

07) Utilizando o programa shell, em linha de comando, liste todos os processos existentes no seu computador. Faça isso tanto no Linux quanto no Windows.

```

Prompt de Comando
Microsoft Windows [versão 10.0.19043.1110]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Pedro>tasklist

Nome da imagem                Identifi Nome da sessão    Sessão#  Uso de memór
=====
System Idle Process           0 Services              0         8 K
System                        4 Services              0        120 K
Registry                     100 Services            0       77.716 K
smss.exe                      348 Services            0        1.192 K
csrss.exe                     568 Services            0        5.428 K
wininit.exe                   808 Services            0        7.040 K
csrss.exe                      816 Console             1         5.960 K
services.exe                  912 Services            0       10.088 K
lsass.exe                     956 Services            0       21.832 K
winlogon.exe                  972 Console             1       11.756 K
svchost.exe                   660 Services            0       30.816 K
fontdrvhost.exe              696 Services            0         3.492 K
fontdrvhost.exe              684 Console             1         7.824 K
svchost.exe                   496 Services            0       13.640 K
svchost.exe                   420 Services            0         8.536 K
dwm.exe                       1032 Console            1       77.812 K
svchost.exe                   1104 Services           0         7.896 K
svchost.exe                   1152 Services           0       10.448 K
svchost.exe                   1196 Services           0       11.920 K
svchost.exe                   1240 Services           0         9.292 K
svchost.exe                   1248 Services           0       15.436 K
svchost.exe                   1296 Services           0         6.464 K
svchost.exe                   1344 Services           0       13.824 K

```

Figura 16 – Lista de processos em execução obtida através do prompt de comandos do Windows

```

pedro@computer: ~
MB swap: 1953,0 total, 1953,0 livre, 0,0 usados, 6101,7 mem dispon.

  PID USUARIO  PR  NI   VIRT   RES   SHR S  %CPU  %MEM  TEMPO+  COMANDO
  1783 pedro    20   0 1154060  85036 48568 R   9,7   1,1   0:09.77 Xorg
  1931 pedro    20   0 4148364 239404 86604 S   6,5   3,0   0:12.32 gnome-shell
  2411 pedro    20   0   13,6g 295404 104352 S   6,5   3,7   0:29.75 teams
  3205 pedro    20   0  824624  52208 39660 S   6,5   0,6   0:00.76 gnome-terminal-

top - 19:17:59 up 11 min, 1 user, load average: 0,85, 0,48, 0,25
Tarefas: 242 total, 1 em exec., 241 dormindo, 0 parado, 0 zumbi
%CPU(s): 12,3 us, 4,0 sis, 0,0 ni, 83,6 oc, 0,0 ag, 0,0 ih, 0,0 is, 0,0 tr
MB mem : 7857,2 total, 4921,2 livre, 1295,1 usados, 1640,9 buff/cache
MB swap: 1953,0 total, 1953,0 livre, 0,0 usados, 6005,5 mem dispon.

  PID USUARIO  PR  NI   VIRT   RES   SHR S  %CPU  %MEM  TEMPO+  COMANDO
  1783 pedro    20   0 1158484  86400 49308 S  20,9   1,1   0:14.96 Xorg
  1931 pedro    20   0 4149616 242296 87064 S  14,6   3,0   0:18.37 gnome-shell
  3376 pedro    20   0  745072  46304 35376 S   9,9   0,6   0:00.79 gnome-screensho
  2411 pedro    20   0   13,6g 295448 104352 S   3,6   3,7   0:31.93 teams
  3205 pedro    20   0  824768  52436 39796 S   3,0   0,7   0:01.94 gnome-terminal-
  2357 pedro    20   0 1016868  79368 50372 S   2,6   1,0   0:10.34 teams
  1662 root      20   0         0         0      0 I   0,7   0,0   0:00.67 kworker/1:0-events

```

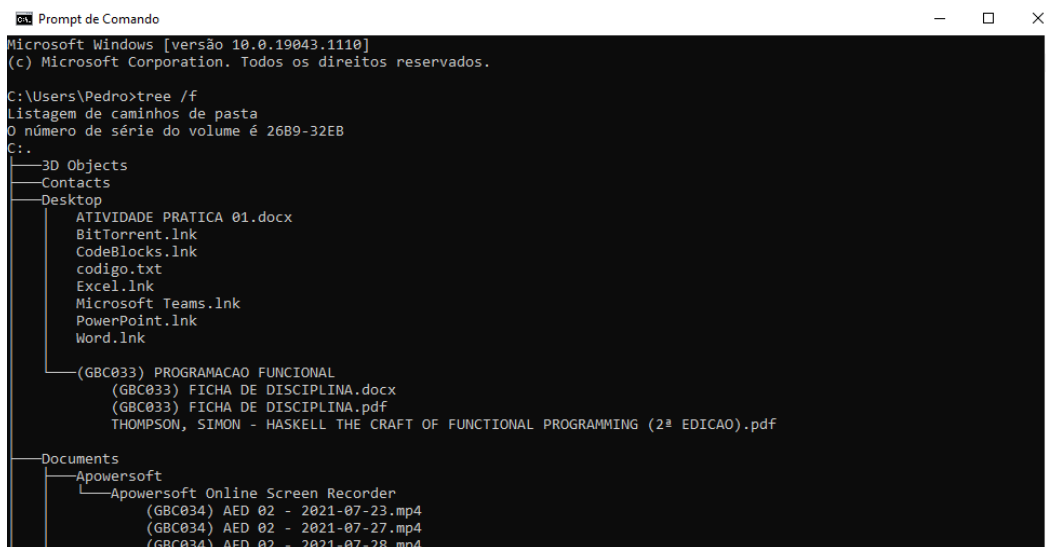
Figura 17 – Lista de processos em execução obtidas no terminal do Linux

De acordo com o site do Vanderley (2010), para listar os processos em execução utilizando o prompt de comandos do Windows, basta digitar o comando tasklist. A Figura 16 ilustra o que foi descrito.

A Figura 17 mostra como listar os processos em execução no terminal do Linux. Para fazer isso, basta digitar o comando “top” no terminal e apertar enter.

## **08) Utilizando o programa shell, em linha de comando, liste todos os arquivos existentes no seu computador. Faça isso tanto no Linux quanto no Windows.**

De acordo com o site da Microsoft (2017), o comando “tree” exibe a estrutura do disco graficamente. Associado com o comando “tree” é possível adicionar “/f” para exibir o nome dos arquivos em cada diretório. A Figura 18 ilustra o que foi descrito.



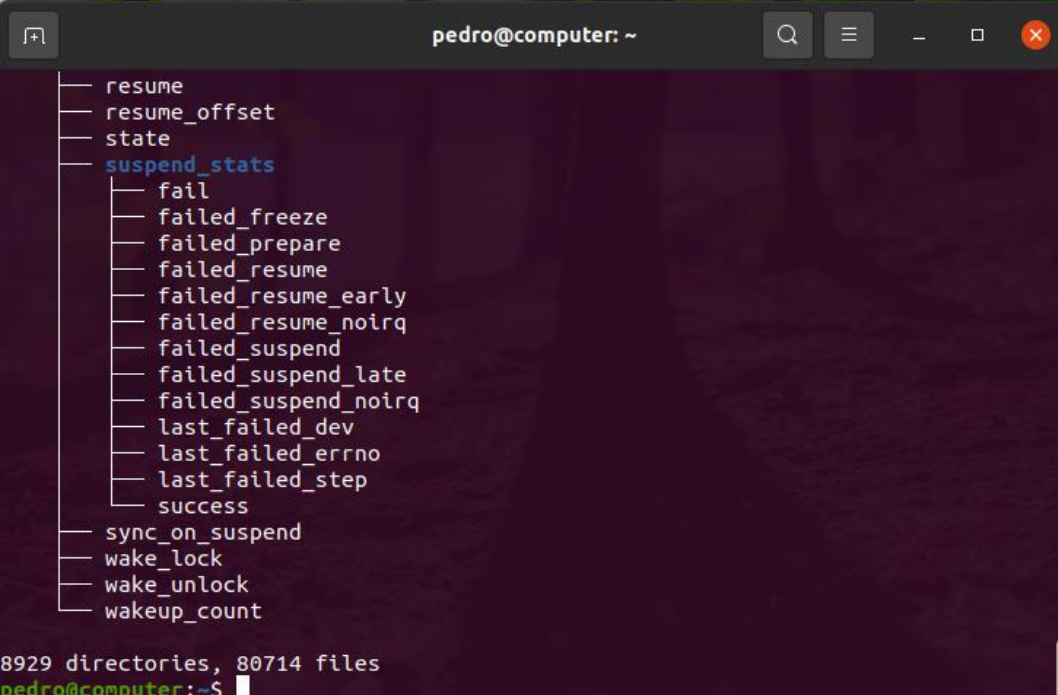
```
Prompt de Comando
Microsoft Windows [versão 10.0.19043.1110]
(c) Microsoft Corporation. Todos os direitos reservados.

C:\Users\Pedro>tree /f
Listagem de caminhos de pasta
O número de série do volume é 26B9-32EB
C:.
|-- 3D Objects
|-- Contacts
|-- Desktop
|   |-- ATIVIDADE PRATICA 01.docx
|   |-- BitTorrent.lnk
|   |-- CodeBlocks.lnk
|   |-- codigo.txt
|   |-- Excel.lnk
|   |-- Microsoft Teams.lnk
|   |-- PowerPoint.lnk
|   |-- Word.lnk
|   |-- (GBC033) PROGRAMACAO FUNCIONAL
|       |-- (GBC033) FICHA DE DISCIPLINA.docx
|       |-- (GBC033) FICHA DE DISCIPLINA.pdf
|       |-- THOMPSON, SIMON - HASKELL THE CRAFT OF FUNCTIONAL PROGRAMMING (2ª EDICAO).pdf
|-- Documents
|   |-- Apowersoft
|       |-- Apowersoft Online Screen Recorder
|           |-- (GBC034) AED 02 - 2021-07-23.mp4
|           |-- (GBC034) AED 02 - 2021-07-27.mp4
|           |-- (GBC034) AED 02 - 2021-07-28.mp4
```

Figura 18 – Lista dos arquivos existentes no computador utilizando o prompt de comandos do Windows

Para listar os arquivos presentes no computador utilizando o terminal do Linux, basta digitar o comando “tree /sys” e o nome de todos os arquivos serão mostrados no formato de árvore, semelhante ao que foi mostrado na Figura 18. A Figura 19 mostra a árvore com os nomes dos arquivos.





```
pedro@computer: ~  
— resume  
— resume_offset  
— state  
— suspend_stats  
  — fail  
  — failed_freeze  
  — failed_prepare  
  — failed_resume  
  — failed_resume_early  
  — failed_resume_noirq  
  — failed_suspend  
  — failed_suspend_late  
  — failed_suspend_noirq  
  — last_failed_dev  
  — last_failed_errno  
  — last_failed_step  
  — success  
— sync_on_suspend  
— wake_lock  
— wake_unlock  
— wakeup_count  
  
8929 directories, 80714 files  
pedro@computer:~$
```

Figura 19 - Lista dos arquivos existentes no computador através do terminal de comandos do Linux

**09) Utilizando ferramentas do sistema operacional, verifique o espaço livre em memória principal e secundária do seu computador. Faça isso tanto no Linux quanto no Windows.**

Para analisar o espaço livre da memória principal no Windows, pode-se utilizar o gerenciador de tarefas. Ele mostra a quantidade de memória em uso e a quantidade disponível. A Figura 20 ilustra a quantidade de memória RAM disponível que, para o computador utilizado, é de 4,4 GB.

Para analisar o espaço livre em disco (memória secundária) no Windows, pode-se utilizar o explorador de arquivos. Ao abrir o explorador de arquivos, o SO informa a quantidade de espaço livre dos discos presentes. No caso, o SSD de 120 GB foi dividido em 2 partições (uma para o Windows e outra para o Linux) e o HD também foi particionado, uma partição para instalação de programas de baixo uso e outra para armazenar arquivos. O restante da unidade é utilizada para armazenar arquivos do Linux. A Figura 21 ilustra o que foi descrito.

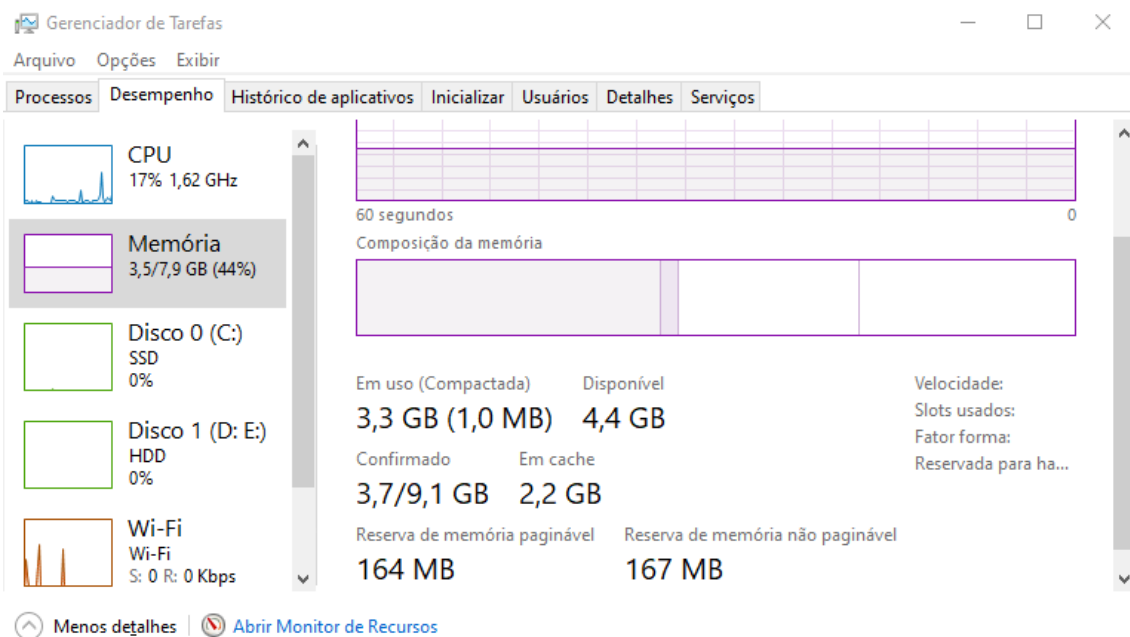


Figura 20 – Quantidade de memória RAM disponível vista pelo gerenciado de tarefas do Windows

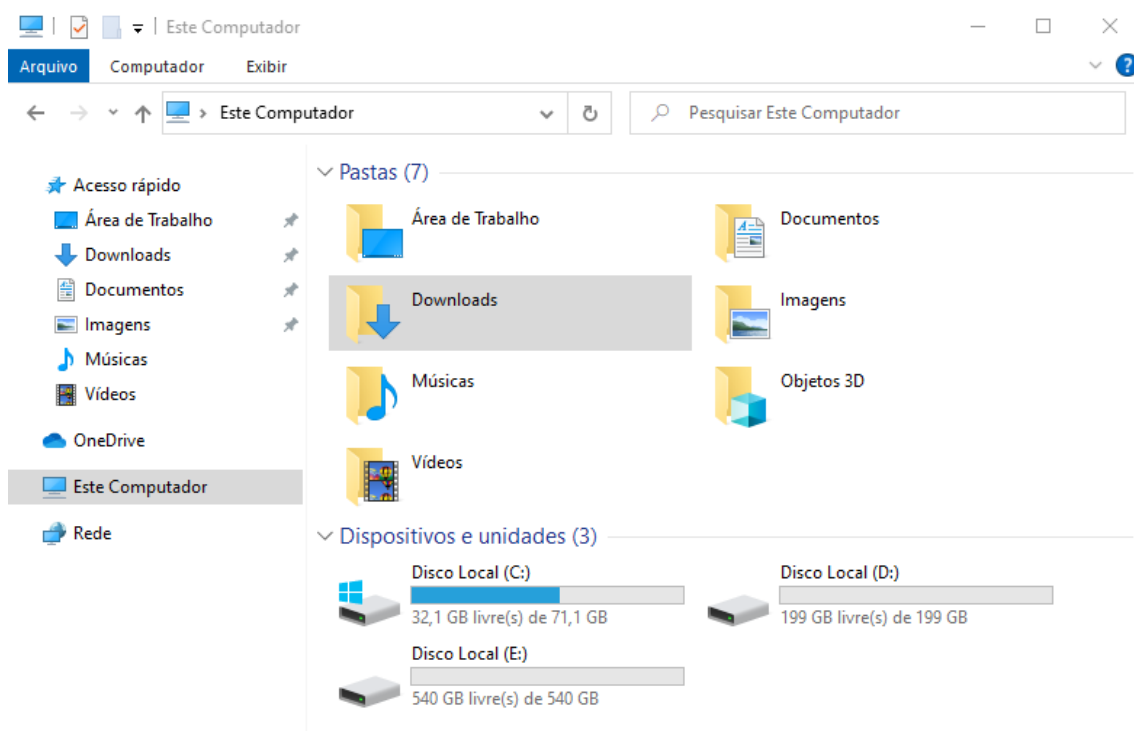


Figura 21 – Espaço livre e utilizado dos discos visto pelo explorador de arquivos do Windows



No Ubuntu, para analisar o espaço livre da memória RAM, pode-se digitar o comando “free -m” no terminal. A Figura 22 mostra as informações obtidas com o comando. Para analisar o espaço livre dos discos, basta utilizar o gerenciador de arquivos do Ubuntu. Ele fornece a informação de quantos gigas foram ocupados e quantos estão livres para uso. A Figura 23 mostra os valores para o computador utilizado.



```

pedro@computer: ~
pedro@computer:~$ free
              total        usada        livre      compart.  buff/cache  disponível
Mem.:          8045820      1259988      4339252        305524      2446580      6187692
Swap:          1999868           0      1999868
pedro@computer:~$ free -m
              total        usada        livre      compart.  buff/cache  disponível
Mem.:           7857         1230         4239           296         2387         6044
Swap:           1952           0         1952
pedro@computer:~$

```

Figura 22 – Espaço livre de memória RAM visto pelo terminal do Linux

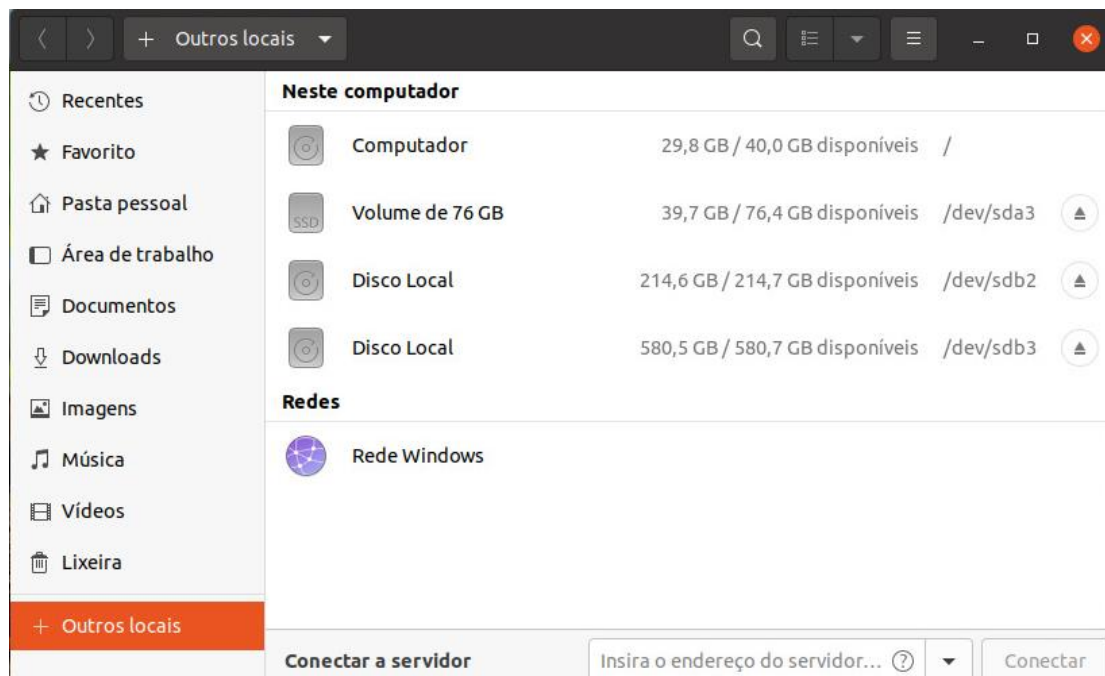


Figura 23 - Espaço livre em disco visto pelo gerenciador de arquivos do Linux

**10) Escreva um programa shell que tenha interface com o usuário por meio da linha de comando para o sistema operacional Linux. O programa shell deve suportar os seguintes comandos:**

- **CLS:** apagar a tela
- **DIR:** listar os arquivos
- **DATE:** mostrar a data corrente do sistema no formato DD/MM/AAAA
- **TIME:** mostrar a hora corrente do sistema no formato HH:MM:SS

O código abaixo foi criado para executar no Linux e contempla todos os comandos citados acima. Um comando adicional “exit” foi incluído para encerrar a execução do terminal. O código está comentado para facilitar o entendimento.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#include <dirent.h>
#include <unistd.h>
#include <time.h>

void limpar_tela( ){
    // Impressão para limpar a tela do prompt
    printf("\033[H\033[J");
}

void limpar_buffer( ){
    // Limpa o buffer do teclado
    setbuf(stdin, NULL);
}

void cabecalho( ){
    // Cabeçalho para quando limpa a tela
    printf("Terminal do Windows no Ubuntu\n");
    printf("Copyright (c) 2021 Pedro Ribeiro\n");

    printf("\n");
}

void main( ){
    char comando[20]; // Armazena o comando digitado
    char *diretorio_atual; // Ponteiro para o diretório atual
    int dia, mes, ano; // Variáveis aux. para a data
    int horas, minutos, segundos; // Variáveis aux. para o horário
```

```

printf("\n");

cabecalho( );

while(strcmp(comando, "exit") != 0){ // Encerra com "exit"

    diretorio_atual = getenv("PWD"); // Pega o diretório atual

    printf("user%s$ ", diretorio_atual);
    scanf("%s", &(*comando)); // Leitura do comando
    limpar_buffer( );

    // Comparação do comando digitado
    // Só serão aceitos comandos em minúsculo

    if(strcmp(comando, "cls") == 0){

        limpar_tela( );
        cabecalho( );

    }else if(strcmp(comando, "dir") == 0){

        // Ponteiro para o nome do arquivo
        struct dirent *nome_arquivo;
        // Ponteiro para o diretório de execução
        DIR *diretorio = opendir(diretorio_atual);

        if(diretorio == NULL){
            // Caso ocorra falha de acesso
            printf("\nErro ao acessar o diretorio!\n");
            break;
        }else{

            printf("\nArquivos no diretorio:\n\n");
            // Laço para a impressão dos arquivos do diretório
            // NULL indica que não mais nenhum arquivo
            while((nome_arquivo = readdir(diretorio)) != NULL){
                // Acessa o ponteiro e pega o nome do arquivo
                printf("%s\n", nome_arquivo->d_name);
            }

            printf("\n");

            closedir(diretorio);
        }

    }else if(strcmp(comando, "date") == 0){
        // Pega a data e armazena no ponteiro data
        time_t t = time(NULL);
        struct tm data = *localtime(&t);
        // Variáveis auxiliares para simplificar a impressão
        dia = data.tm_mday;
        mes = data.tm_mon + 1;
        ano = data.tm_year + 1900;
        // %02d formata a impressão para permitir zero a esquerda
        printf("\n%d/%02d/%02d\n\n", dia, mes, ano);
    }else if(strcmp(comando, "time") == 0){

```

```

// Pega o horário e armazena no ponteiro horario
time_t t = time(NULL);
struct tm horario = *localtime(&t);
// Variáveis auxiliares para simplificar a impressão
horas = horario.tm_hour;
minutos = horario.tm_min;
segundos = horario.tm_sec;
// %02d formata a impressão para permitir zeros a esquerda
printf("\n%02d:%02d:%02d\n\n", horas, minutos, segundos);

// Comando exit encerra a execução do terminal
}else if(strcmp(comando, "exit") == 0){

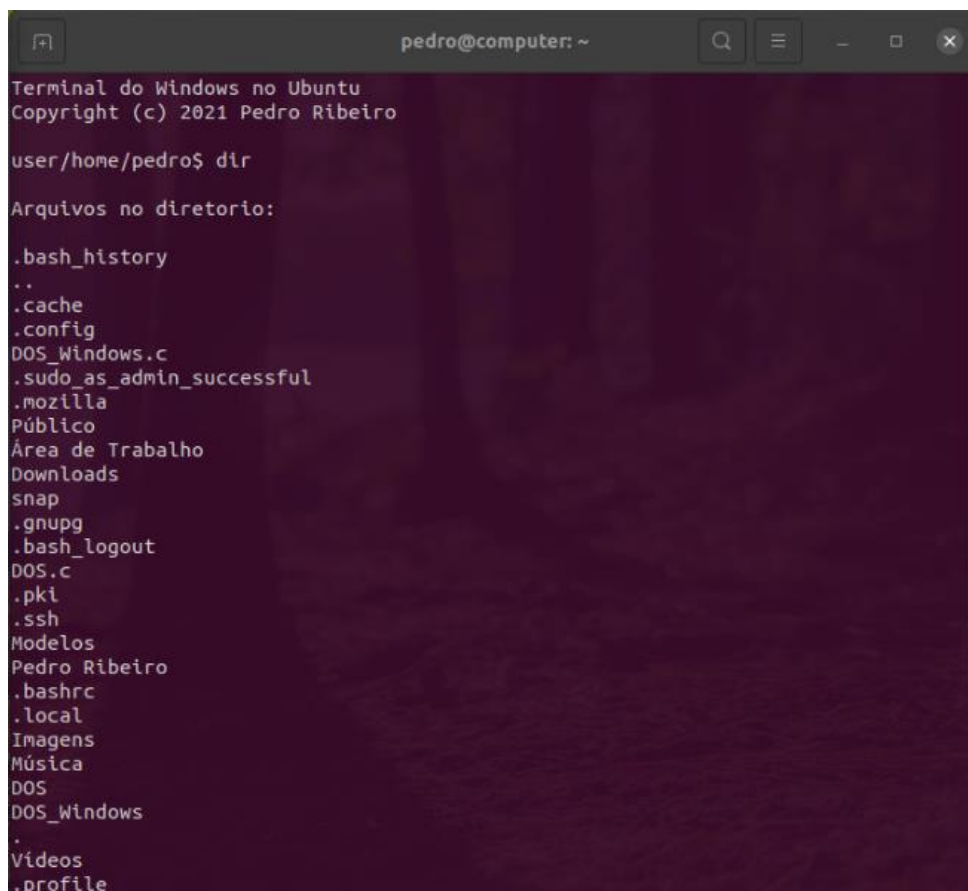
    printf("\nTerminal encerrado!\n");
    break;

}else{
    // Caso o usuário digite um comando errado será mostrada
    // esta mensagem de erro
    printf("\nComando invalido!\n\n");
}

}

printf("\n");
// A função é void main, não é necessário o return 0
}

```



```

pedro@computer: ~
Terminal do Windows no Ubuntu
Copyright (c) 2021 Pedro Ribeiro

user/home/pedro$ dir

Arquivos no diretorio:

.bash_history
..
.cache
.config
DOS_Windows.c
.sudo_as_admin_successful
.mozilla
Público
Área de Trabalho
Downloads
snap
.gnupg
.bash_logout
DOS.c
.pki
.ssh
Modelos
Pedro Ribeiro
.bashrc
.local
Imagens
Música
DOS
DOS_Windows
.
Vídeos
.profile

```

Figura 24 – Funcionamento do terminal da questão 10

## REFERÊNCIAS BIBLIOGRÁFICAS

COUTINHO, Bruno C. Sistemas Operacionais – Curso Técnico em Informática. Disponível em: <<https://www.ufsm.br/app/uploads/sites/342/2020/04/SISTEMAS-OPERACIONAIS.pdf>>. Acessado em: 26/07/2021.

GARRETT, Filipe. O que é setup? Tire suas dúvidas sobre como o sistema funciona. Disponível em: <<https://www.techtudo.com.br/noticias/noticia/2015/02/o-que-e-setup.html>>. Acessado em: 26/07/2021.

VANDERLEY, Paulo. Como fechar um programa no Windows sem usar o gerenciador de tarefas. Disponível em: <<https://www.techtudo.com.br/dicas-e-tutoriais/noticia/2010/12/como-fechar-um-programa-no-windows-sem-usar-o-gerenciador-de-tarefas.html>>. Acessado em: 26/07/2021.

MICROSOFT. Função tree. Disponível em: <<https://docs.microsoft.com/pt-br/windows-server/administration/windows-commands/tree>>. Acessado em: 26/07/2021.

Manual online Linux. Syscalls. Disponível em: <<http://man.he.net/man2/syscalls>>. Acessado em: 24/07/2021.