**OH 10.7**

**Trees**

Linux processes -> kill pids

Creating a virtual ramdisk

-Timing of a virtual disk

-Memory checker commands

-python > find in files

FTP server services

**[ Trees ]**

If not built in: to include the tree command in Fedora terminal, you have a few options:

1. Open a terminal and run:

**sudo apt install tree**

This is the simplest and most straightforward method for Ubuntu users.

1. Or install using Snap:  
   If you prefer using Snap packages, you can enable snaps on Fedora and install the tree command:First, enable snaps:

**sudo apt install snapd**

**sudo ln -s /var/lib/snapd/snap /snap**

Then, install tree:

sudo snap install tree

Note that you may need to log out and back in for the snap installation to take effect

Some coms

Some useful options for the tree command include:

* -a: Show all files, including hidden ones
* -d: List directories only
* -L [number]: Limit the depth of the tree to the specified number of levels
* -f: Print the full path prefix for each file

Dirs only

**tree -d -L 2**

Exclude dirs

**tree -I "node\_modules"**

[ **Linux processes -> kill pids** ]

-Process IDs

Use the ping command to start a process. Ctrl+C terminates the process, but Ctrl+Z makes the process run in the background.

By pressing Ctrl+Z, we are told the process is stopped. Stopped doesn’t mean terminated. It’s like a car at a stop sign. We haven’t scrapped it and thrown it away. It’s still on the road, stationary, waiting to go. The process is now a background *job*.

The jobs command [will list the jobs](http://man7.org/linux/man-pages/man1/jobs.1p.html) that have been started in the current terminal session. And because jobs are (inevitably) processes, we can also use the ps command to see them. Let’s use both commands and compare their outputs. We’ll use the T option (terminal) option to only list the processes that are running in this terminal window. Note that there is no need to use a hyphen - with the T option.

Type the following commands and observe the outputs.

**jobs**

**ps T**

**https://man7.org/linux/man-pages/index.html**

To kill a process running a ping command in Fedora Linux, you can follow these steps:

1. First, find the process ID (PID) of the ping command. You can do this by using the pidof command:

**pidof ping**

This will return the PID of the ping process.

1. Once you have the PID, you can use the kill command to terminate the process:

**kill PID**

*Replace PID* with the actual process ID number you obtained in step 1.

1. If the process doesn't terminate with the default SIGTERM signal, you can use a stronger signal like SIGKILL:

**kill -9 PID**

The -9 option sends the SIGKILL signal, which forces the process to stop immediately without giving it a chance to clean up

**killall ping**

[ **Vitrual RAMDISK** ]

Virtual RAM disk

\*GUI apps to consider working a virtual ram disk

AMD Radeon RAMDisk nor StarWind RAM Disk

Exercise:

Create directory

**sudo mkdir -p /mnt/ramdisk**

Mount disk (as temporary)

**sudo mount -t tmpfs -o size=1G tmpfs /mnt/ramdisk**

Verify disk mount

**df -h**

Look for an entry with tmpfs in the output, which should show the size and usage of your RAM disk

Script to do same as above

#!/bin/bash

# Check if the RAM disk is already mounted

if [ "$(mount | grep -o "/mnt/ramdisk")" != "/mnt/ramdisk" ]; then

# Create the mount point directory

sudo mkdir -p /mnt/ramdisk

# Mount the RAM disk

sudo mount -t tmpfs -o size=1G tmpfs /mnt/ramdisk

# Change ownership to the current user

sudo chown -R $(whoami):$(whoami) /mnt/ramdisk

fi

# Verify the mount

mount | grep ramdisk

unmount disk

#!/bin/bash

# Unmount the RAM disk

sudo umount /mnt/ramdisk

Run application on ram disk / test ram disk for speed, etc.

**sudo apt install python3**

**sudo apt update *# Install pip for Python 3***

**sudo apt install python3-pip**

Working with curl commands

Create tmp directory and travel (cd) to it

For text file

**curl -O** [**https://www.gutenberg.org/files/1342/1342-0.txt**](https://www.gutenberg.org/files/1342/1342-0.txt)

Observe content of txt file with cat command

**cat 1342-0.txt**

Do a word count

**wc -l 1342-0.txt**

For html file (download with different file name)

**curl -o myfile.html https://www.example.com/index.html**

For image file

**curl -O https://upload.wikimedia.org/wikipedia/commons/thumb/3/35/Tux.svg/1200px-Tux.svg.png**

Open image file via command file above via ImageMagick!

**display 1200px-Tux.svg.png**

\*if necessary install imageMagick

**sudo apt install ImageMagick**

Create helloword.py file and execute it as test

**nano helloworld.py**

Enter in following code – save when complete

for i in range(1, 1000001):

if i % 1000 == 0:

print("Hello World")

Run program

**python3 helloworld.py**

Split terminals by including new tabs and run executions

Copy files from tmp directory -> ramdisk

Ready file for test run executions

Ex.

**cp file /mnt/ramdisk**

Time app runs

**time python3 helloworld.py**

Observe results

Memory test

**free -h**

or

**top**

Full test will follow

--

Options

Truncate existing files (not to eliminate file name for recreation)

**truncate -s 0 filename.txt**

--

Testing time of app runs

**time python3 your\_script.py**

Other Options:

**pip install flask**

create file app.py

Add contents to file from any text editor

**from flask import Flask**

**app = Flask(\_\_name\_\_)**

**@app.route('/')**

**def hello\_world():**

**return 'Hello, World!'**

**if \_\_name\_\_ == '\_\_main\_\_':**

**app.run(host='0.0.0.0', port=5000)**

Test file with curl command

Start app

**python3 app.py**

Test file output with curl command

**curl http://localhost:5000**

app.py content (adjust num\_lines for argument call in main if nec.)

import re

import time

def read\_file(filename, num\_lines=1000):

lines = []

with open(filename, 'r', encoding='utf-8') as file:

for \_ in range(num\_lines):

line = file.readline()

if not line:

break

lines.append(line)

return ''.join(lines)

def split\_into\_sentences(text):

sentences = re.split(r'(?<=[.!?])\s+', text)

return [sentence.strip() for sentence in sentences if sentence.strip()]

def display\_sentences(sentences):

for i, sentence in enumerate(sentences, 1):

print(f"Sentence {i}: {sentence}")

def main():

filename = "1342-0.txt"

start\_time = time.time()

# Read the first 1000 lines of the file

content = read\_file(filename, 1000)

# Split into sentences

sentences = split\_into\_sentences(content)

# Display sentences

display\_sentences(sentences)

end\_time = time.time()

print(f"\nProcessing time: {end\_time - start\_time:.2f} seconds")

print(f"Total sentences: {len(sentences)}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

Testing time of app runs from both ramdisk and tmp directory

**time python3 your\_script.py**

[ **FTP** ]

Set up server (vsftpd)

Install package

**sudo apt install vsftpd**

Open config file for entering

**sudo nano /etc/vsftpd/vsftpd.conf**

**Edit config file**

listen=YES

write\_enable=YES

xferlog\_enable=YES

connect\_from\_port\_20=YES

anonymous\_enable=NO

listen\_ipv6=NO

local\_enable=YES

Note- if you don’t see any of the lines above just add them in at the bottom of the file.

Save file and exit.

Add ftp service to firewall

**sudo firewall-cmd --add-service=ftp --permanent**

**sudo firewall-cmd –reload**

Enable and start service

**sudo systemctl enable vsftpd**

**sudo systemctl start vsftpd**

\*option – policy enforcement

**sudo setenforce 0**

check ports

**sudo ss -tulpn | grep :21**

FTP a file to a directory

From your directory type the following

**touch me**

**mkdir transfers**

**ip addr show** (note your address)

start ftp

**ftp your\_ip\_address**

Enter your credentials for username / password

If prompted to install ftp, do it!

**Then at ftp prompt type**

**ftp>cd transfers**

Initiate file (me) upload

**ftp>put me**

Bingo!

\*Using filezilla client

<https://filezilla-project.org/>