

# ASSIGNMENT 1

## AIM:

Create a shell for ubuntu operating system which will mimic the behavior of bash shell.

## THEORY:

- Shell:

A shell is a special user program that provides an interface for the user to use operating system services. Shell accepts human-readable commands from users and converts them into something which the kernel can understand. It is a command language interpreter that executes commands read from input devices such as keyboards or from files.

Shell can be accessed by users using a command line interface. A special program called Terminal in Linux/macOS, or Command Prompt in Windows OS is provided to type in the human-readable commands such as “cat”, “ls” etc. and then it is being executed. The result is then displayed on the terminal to the user.

- Subprocess Module: The Python subprocess module is a tool that allows you to run other programs or commands from your Python code. It can be used to open new programs, send them data and get results back.
- Tkinter: In this experiment we use Tkinter to create the shell interface. Tkinter is the inbuilt python module that is used to create GUI applications.  
`from tkinter import *`  
`from tkinter.ttk import *`

## CODE:

```
import tkinter as tk
import subprocess
import os

def run_command(command):
    result = subprocess.run(command, shell=True, capture_output=True,
                             text=True)
    output_text.insert(tk.END, f"$ {command}\n")
    output_text.insert(tk.END, result.stdout)
    output_text.insert(tk.END, result.stderr)

def execute_command(event=None):
    result=[]
    command = command_entry.get("1.0", tk.END).strip()
    if command=='history':
        with open(os.path.join(os.path.expanduser('~'), '.bash_history'),
                  'r') as f:
            for line in f:
                result.append(line)

        output_text.insert(tk.END, f"$ {command}\n")
        output_text.insert(tk.END, result)

    if command.startswith("cd "):
        new_directory = command[3:]
        try:
            os.chdir(new_directory)
        except FileNotFoundError:
            output_text.insert(tk.END, f"Directory not
found:{new_directory}\n")
    if command.startswith("clear"):
        output_text.delete(1.0,tk.END)

    else:
        run_command(command)
        command_entry.delete("1.0", tk.END)

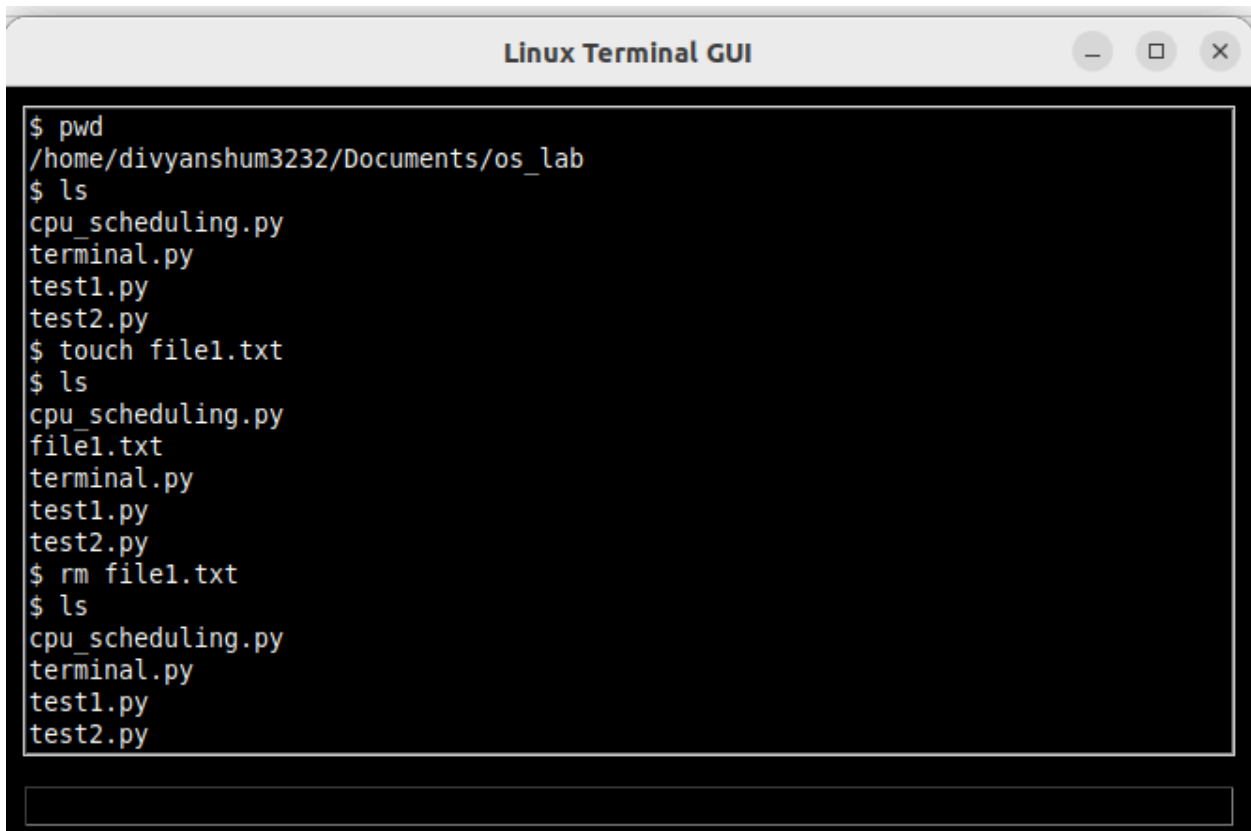
root = tk.Tk()

root.configure(background="black")
root.title("Linux Terminal GUI")
output_text = tk.Text(root, height=20, width=80,
```

```
background="black",fg="white")
output_text.pack(pady=10, padx=10)
output_text.insert(tk.END, "")
command_entry = tk.Text(root, height=1, width=80,background="black",
fg="white")
command_entry.pack(pady=5, padx=10)
command_entry.bind('<Return>', execute_command)
root.mainloop()
```

## OUTPUT:

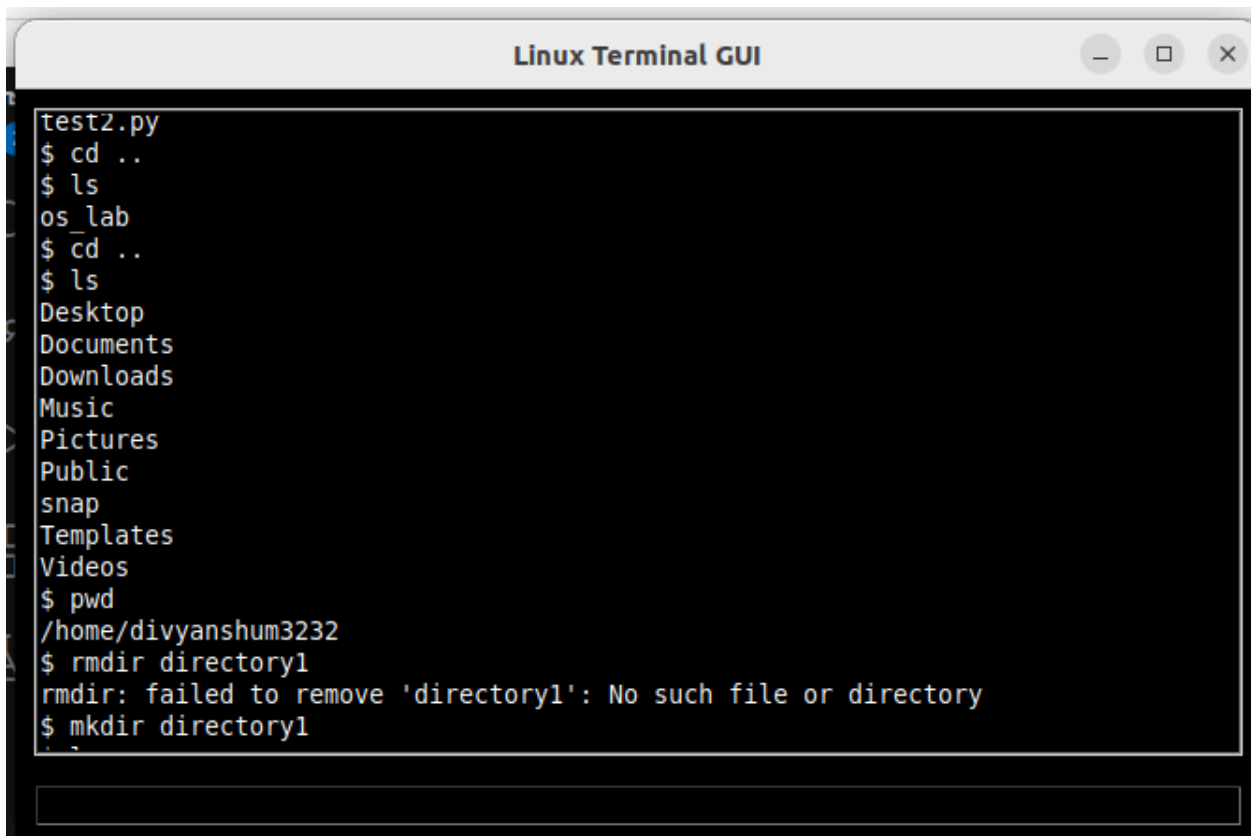
- ls,pwd,touch,rm commands:



```
Linux Terminal GUI

$ pwd
/home/divyanshum3232/Documents/os_lab
$ ls
cpu_scheduling.py
terminal.py
test1.py
test2.py
$ touch file1.txt
$ ls
cpu_scheduling.py
file1.txt
terminal.py
test1.py
test2.py
$ rm file1.txt
$ ls
cpu_scheduling.py
terminal.py
test1.py
test2.py
```

- cd,mkdir,rmdir commands:

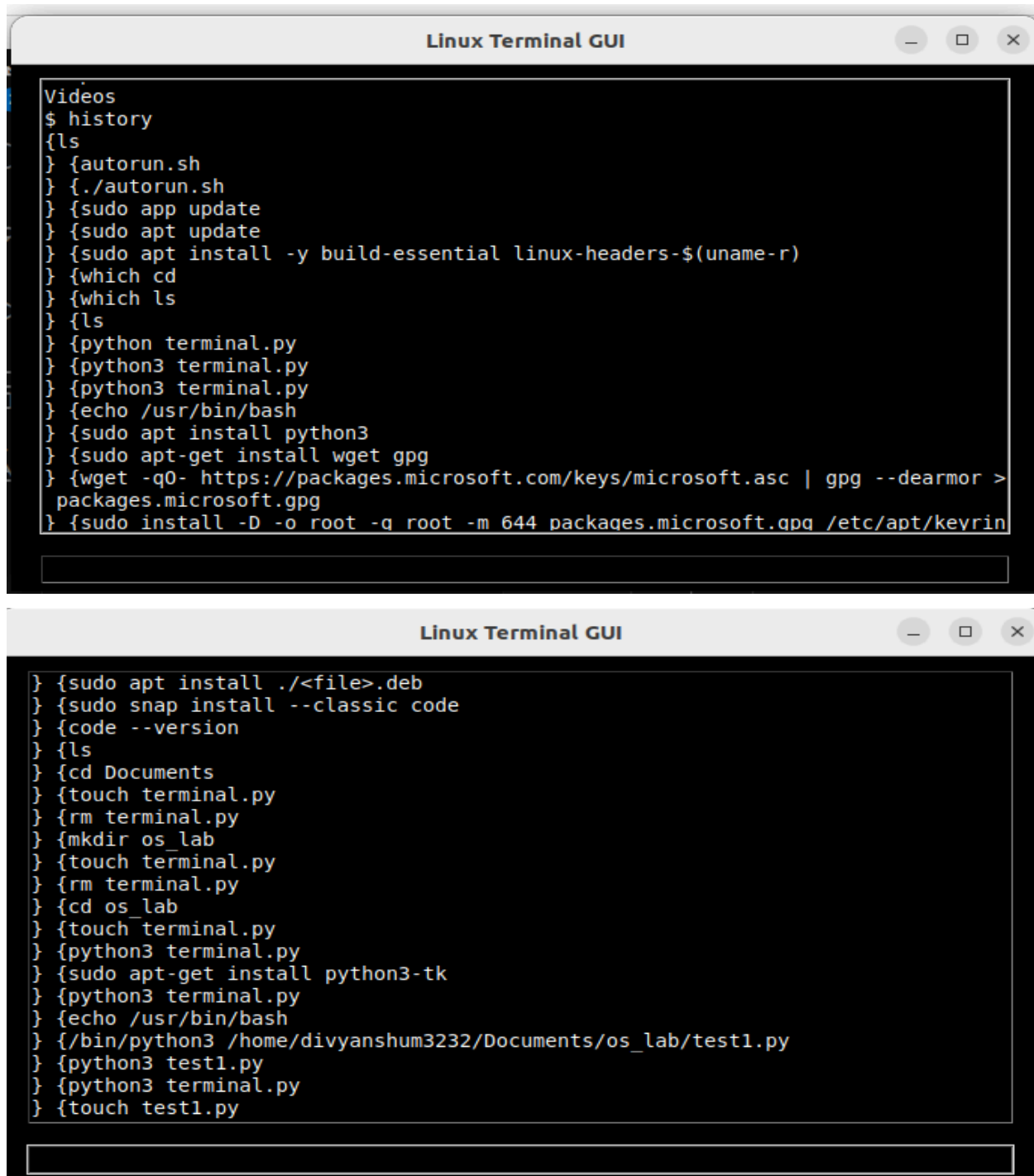


```
test2.py
$ cd ..
$ ls
os_lab
$ cd ..
$ ls
Desktop
Documents
Downloads
Music
Pictures
Public
snap
Templates
Videos
$ pwd
/home/divyanshum3232
$ rmdir directory1
rmdir: failed to remove 'directory1': No such file or directory
$ mkdir directory1
```



```
$ ls
Desktop
directory1
Documents
Downloads
Music
Pictures
Public
snap
Templates
Videos
$ rmdir directory1
$ ls
Desktop
Documents
Downloads
Music
Pictures
Public
snap
```

- history command:



The image shows two screenshots of a Linux Terminal GUI window. The window has a title bar that says "Linux Terminal GUI" and standard window controls (minimize, maximize, close). The terminal displays a list of commands entered, each preceded by a closing curly brace '}', indicating they are part of a history list. The first screenshot shows the first 20 commands of the history, starting with 'Videos' and '\$ history'. The second screenshot shows the continuation of the history, starting from the 21st command.

```
Videos
$ history
{ls
} {autorun.sh
} {./autorun.sh
} {sudo apt update
} {sudo apt update
} {sudo apt install -y build-essential linux-headers-${uname-r}
} {which cd
} {which ls
} {ls
} {python terminal.py
} {python3 terminal.py
} {python3 terminal.py
} {echo /usr/bin/bash
} {sudo apt install python3
} {sudo apt-get install wget gpg
} {wget -qO- https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor >
packages.microsoft.gpg
} {sudo install -D -o root -g root -m 644 packages.microsoft.gpg /etc/apt/keyrin

} {sudo apt install ./<file>.deb
} {sudo snap install --classic code
} {code --version
} {ls
} {cd Documents
} {touch terminal.py
} {rm terminal.py
} {mkdir os_lab
} {touch terminal.py
} {rm terminal.py
} {cd os_lab
} {touch terminal.py
} {python3 terminal.py
} {sudo apt-get install python3-tk
} {python3 terminal.py
} {echo /usr/bin/bash
} {/bin/python3 /home/divyanshum3232/Documents/os_lab/test1.py
} {python3 test1.py
} {python3 terminal.py
} {touch test1.py
```

## CONCLUSION:

Thus, we have successfully learnt to build our own shell terminal.