DIVYANSHU MAKHARIA
221070036

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 *

DIVYANSHU MAKHARIA
221070036

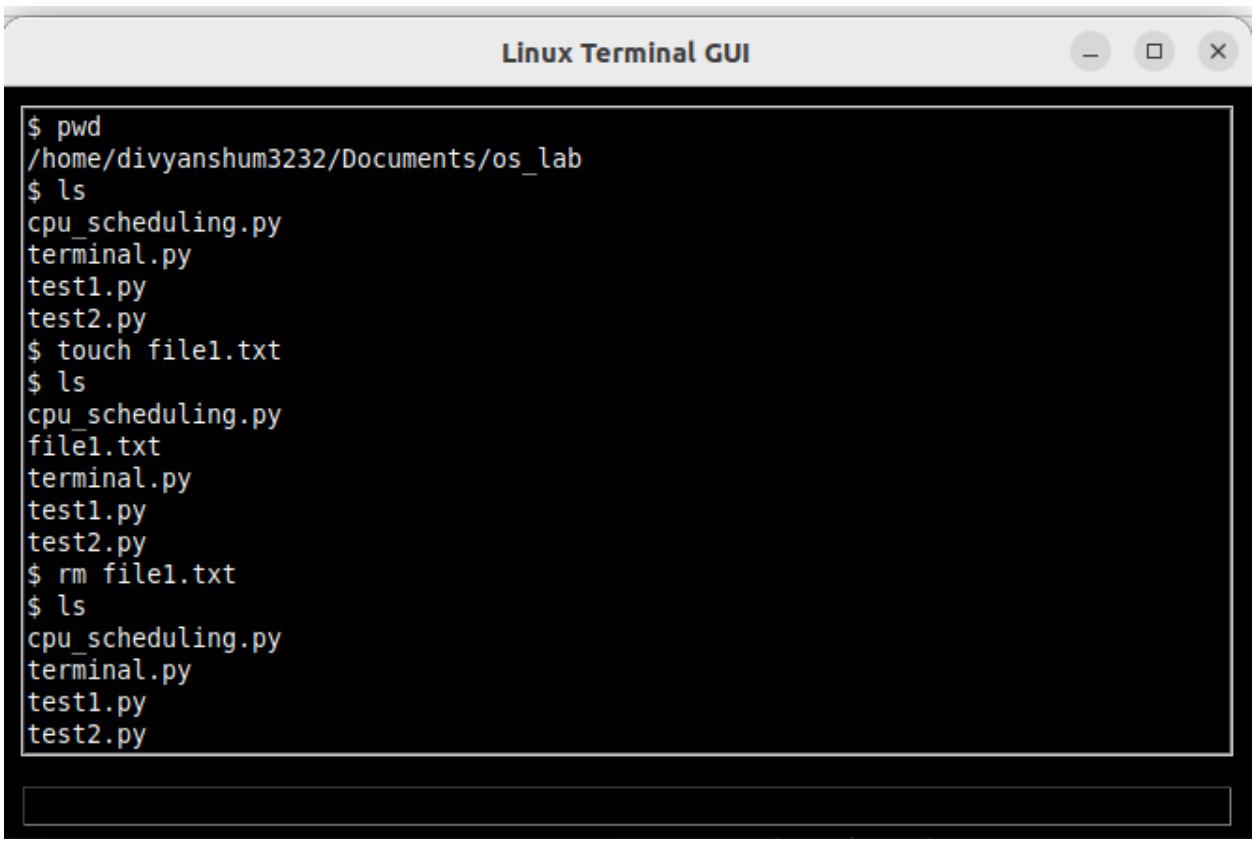
CODE:

```
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,
```

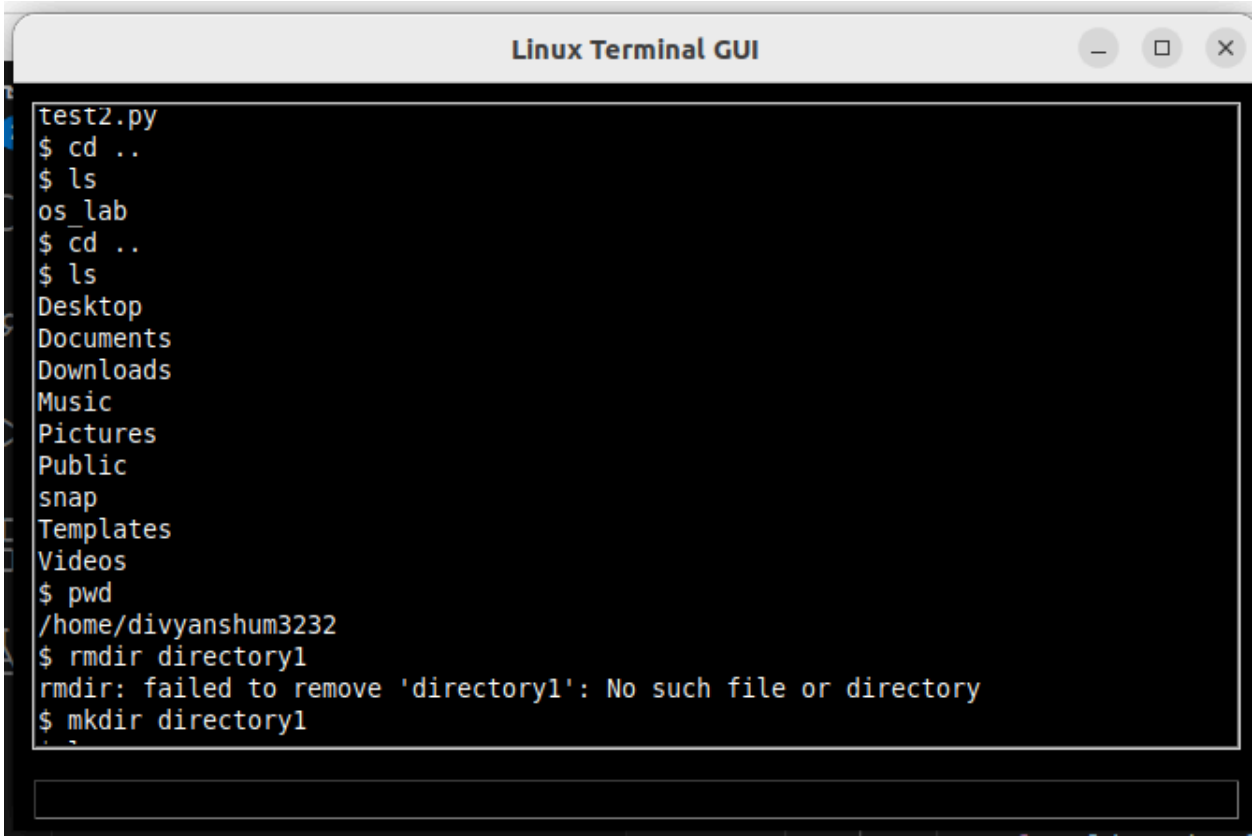
```
background="black",fg="white")
output_text.pack(pady=10, padx=10)
output_text.insert(tk.END, "")
command_entry = tk.Text(root, height=1, width=80,background="black",
fg="white")
command_entry.pack(pady=5, padx=10)
command_entry.bind('<Return>', execute_command)
root.mainloop()
```

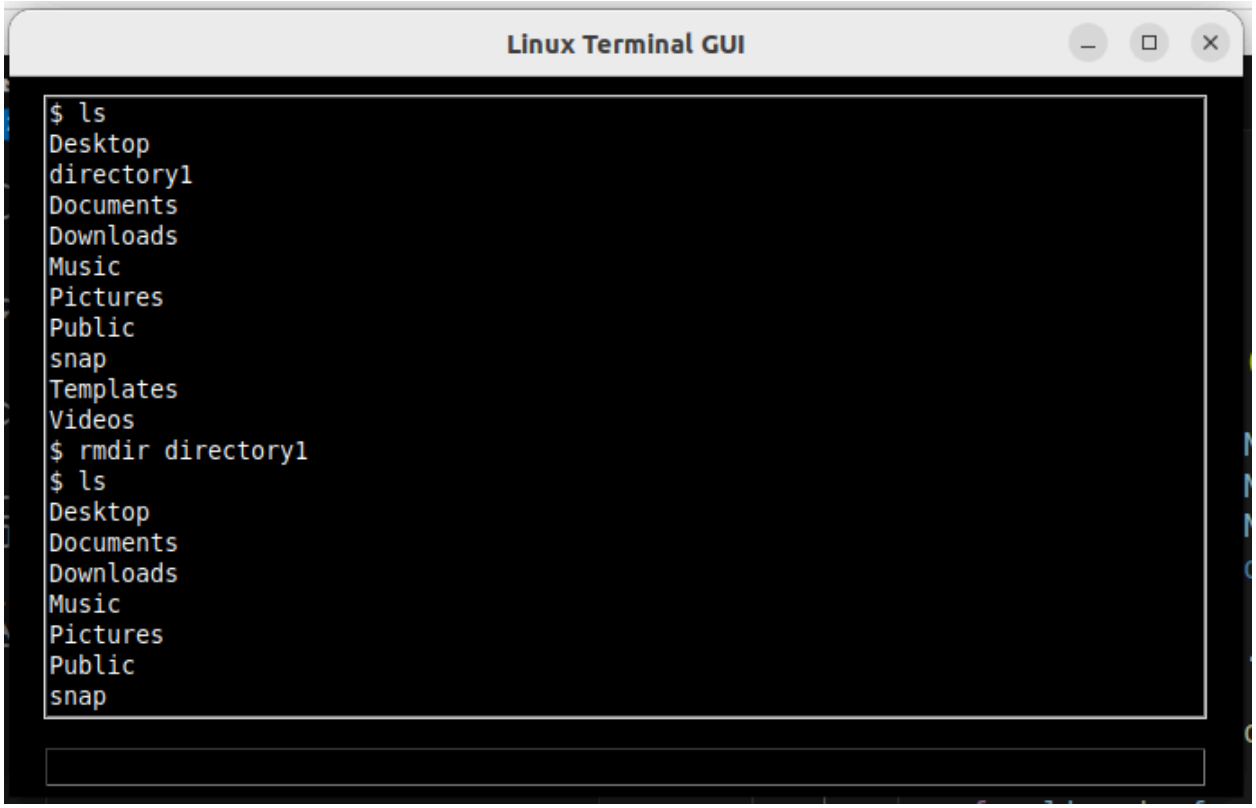
OUTPUT:

- ls,pwd,touch,rm commands:

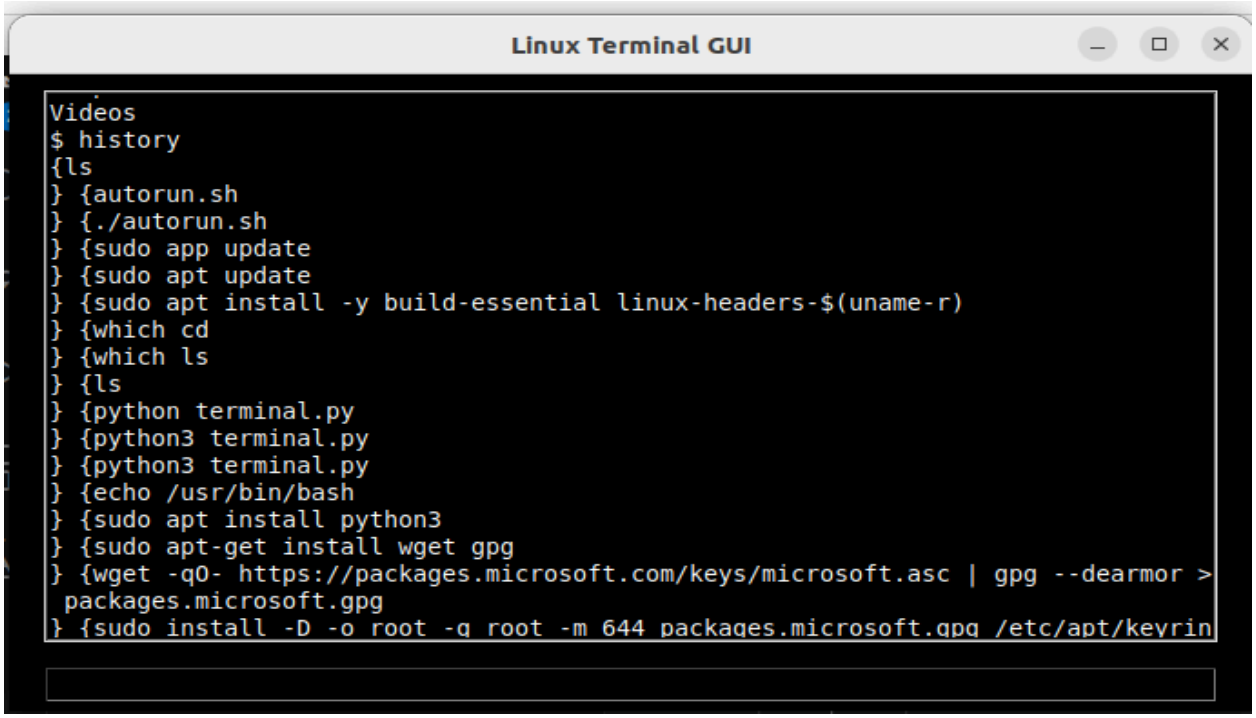


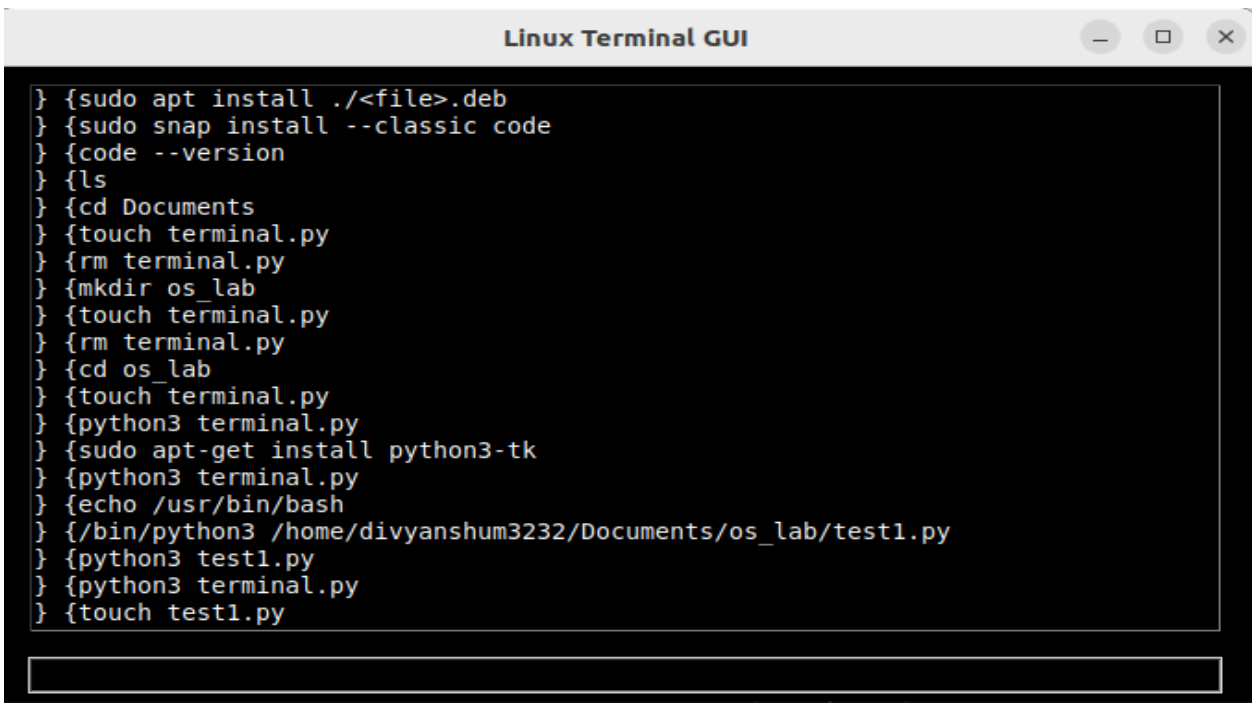
- cd,mkdir,rmdir commands:





- history command:





CONCLUSION:

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