Name: Monika Raghav

Roll.No.: 2301420010

Course: B.Tech(CSE) Data Science

ASSIGNMENT -1

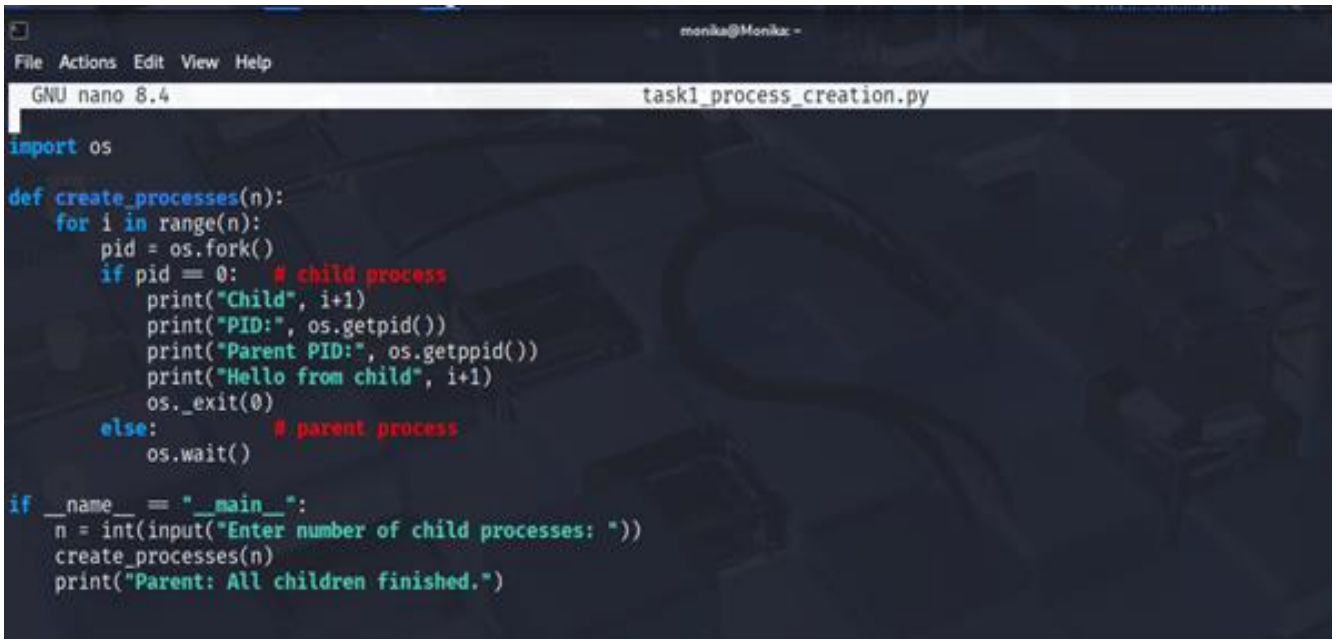
Experiment Title: Process Creation and Management Using Python OS Module

Task 1: Process Creation Utility

Write a Python program that creates N child processes using os.fork(). Each child prints:
- Its PID
- Its Parent PID
- A custom message

The parent should wait for all children using os.wait().

Code:

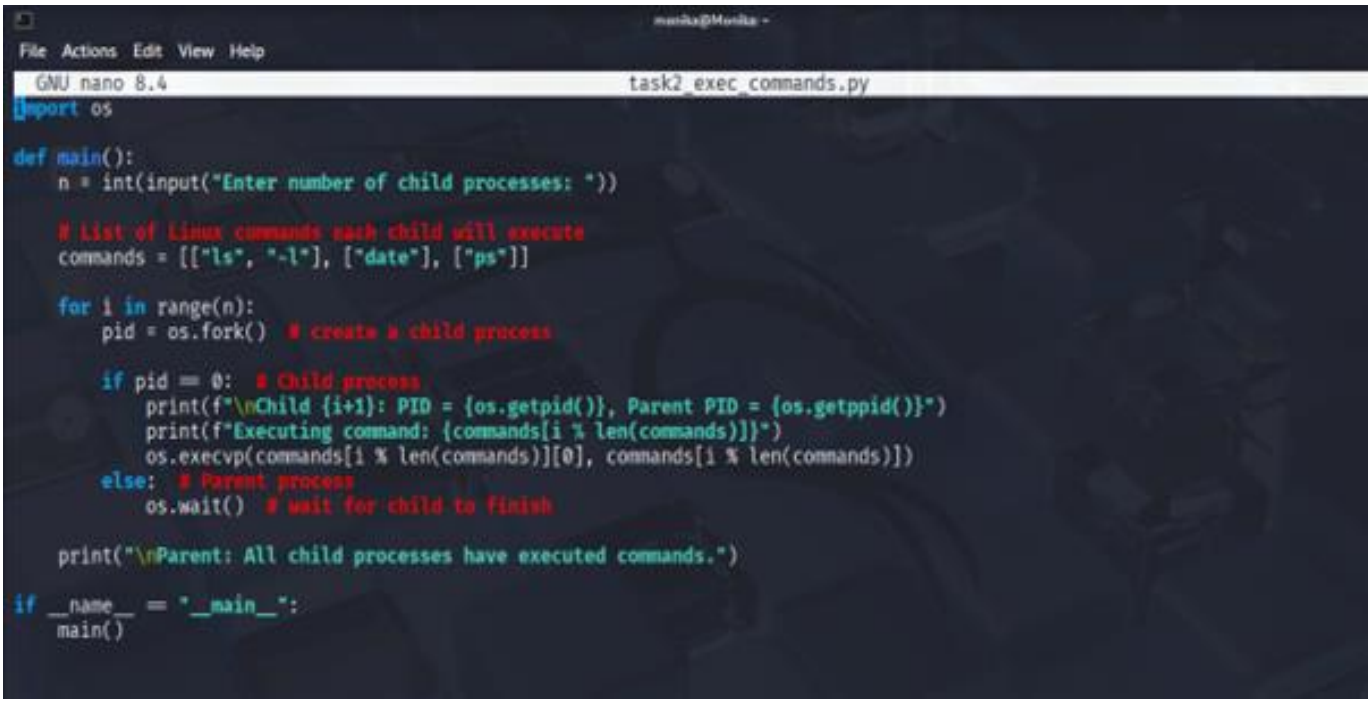


Output:

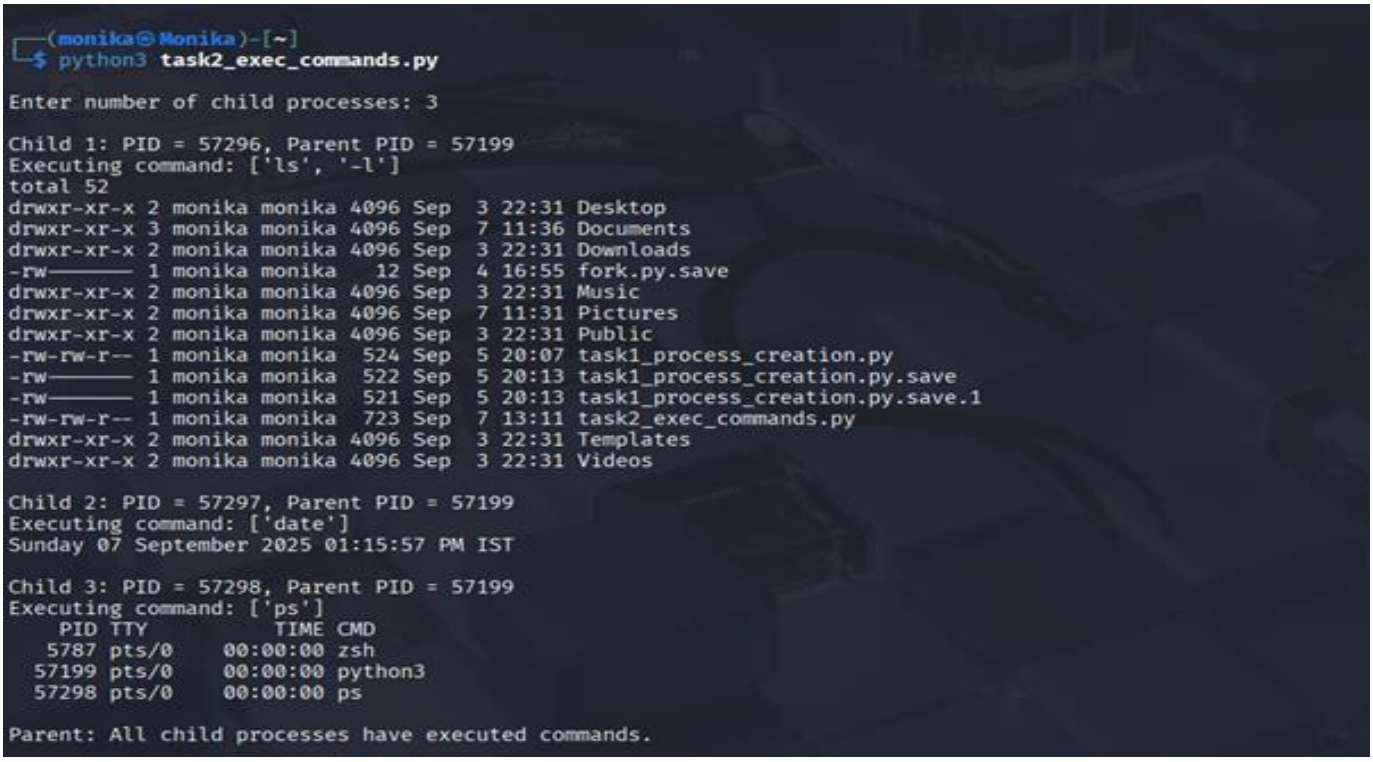
Task 2: Command Execution Using exec()

Modify Task 1 so that each child process executes a Linux command (ls, date, ps, etc.) using os.execvp() or subprocess.run().

Code:



Output:



Task 3: Zombie & Orphan Processes

Zombie: Fork a child and skip wait() in the parent.
Orphan: Parent exits before the child finishes.
Use ps -el | grep defunct to identify zombies.

Code:

```
 GNU nano 8.4                                  task3_zombie_orphan.py *
import os
import time

# ZOMBIE PROCESS

print("---- ZOMBIE PROCESS ----")
pid = os.fork()

if pid == 0:
    # Child process
    print(f"Child (Zombie) PID = {os.getpid()}")
    os._exit(0)  # Child exits immediately
else:
    # Parent does NOT call wait(), so child becomes zombie
    print(f"Parent PID = {os.getpid()} - Child PID = {pid} created")
    print("Sleeping for 10 seconds so child becomes zombie...")
    time.sleep(10)
    print("Parent done sleeping. You can check zombie with 'ps -el | grep defunct'.")

time.sleep(2)  # small delay

# ORPHAN PROCESS

print("\n---- ORPHAN PROCESS ----")
pid2 = os.fork()

if pid2 == 0:
    # Child process
    print(f"Orphan Child PID = {os.getpid()} starting...")
    time.sleep(5)
    print(f"Orphan Child PID = {os.getpid()} done.")
else:
    # Parent exits immediately
    print(f"Parent PID = {os.getpid()} exiting immediately, child becomes orphan")
```

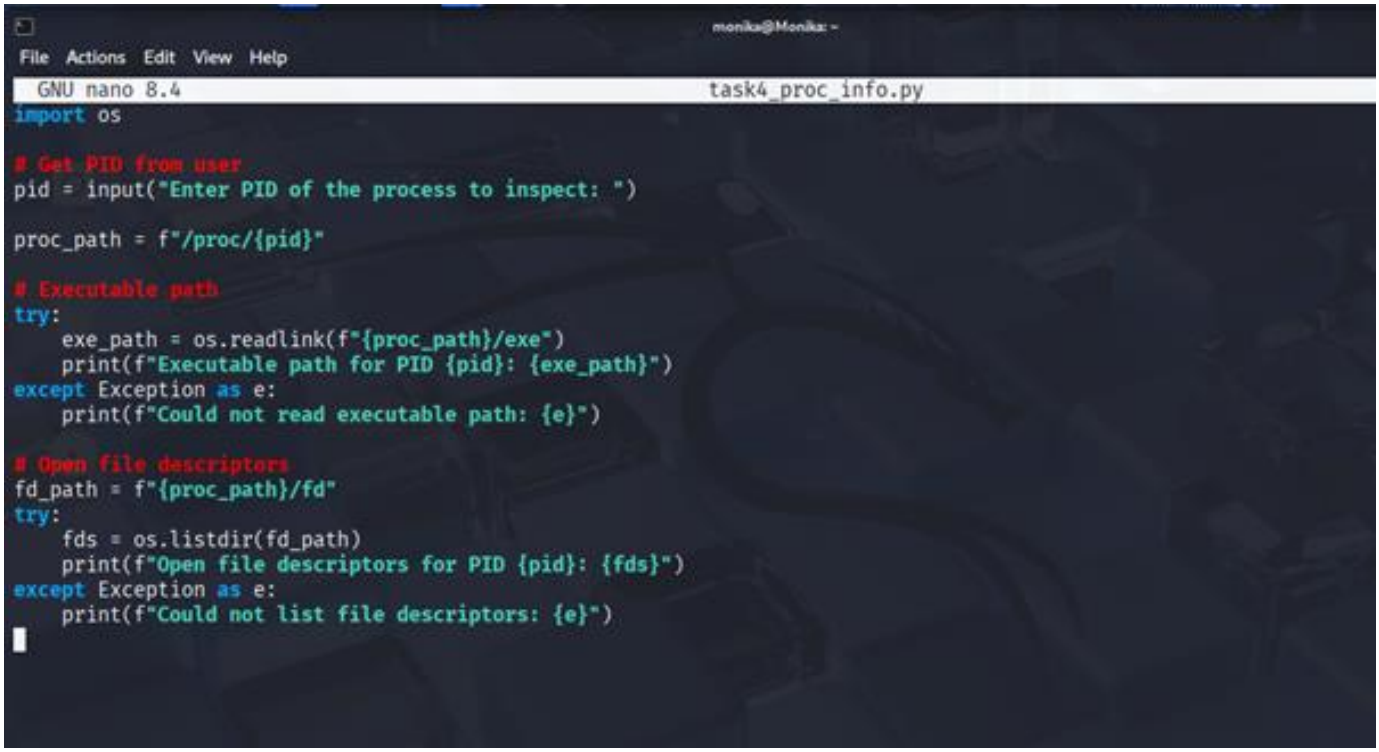
Output:

```
┌──(monika㉿Monika)-[~]
└─$ python3 task3_zombie_orphan.py

---- ZOMBIE PROCESS ----
Child (Zombie) PID = 109149
Parent PID = 109148 - Child PID = 109149 created
Sleeping for 10 seconds so child becomes zombie...
Parent done sleeping. You can check zombie with 'ps -el | grep defunct'.

---- ORPHAN PROCESS ----
Parent PID = 109148 exiting immediately, child becomes orphan
Orphan Child PID = 109246 starting...

┌──(monika㉿Monika)-[~]
└─$ Orphan Child PID = 109246 done.
ps -ps -el | grepunct
```

Task 4: Inspecting Process Info from /proc

Take a PID as input. Read and print:
- Process name, state, memory usage from /proc/[pid]/status
- Executable path from /proc/[pid]/exe
- Open file descriptors from /proc/[pid]/fd

Code:

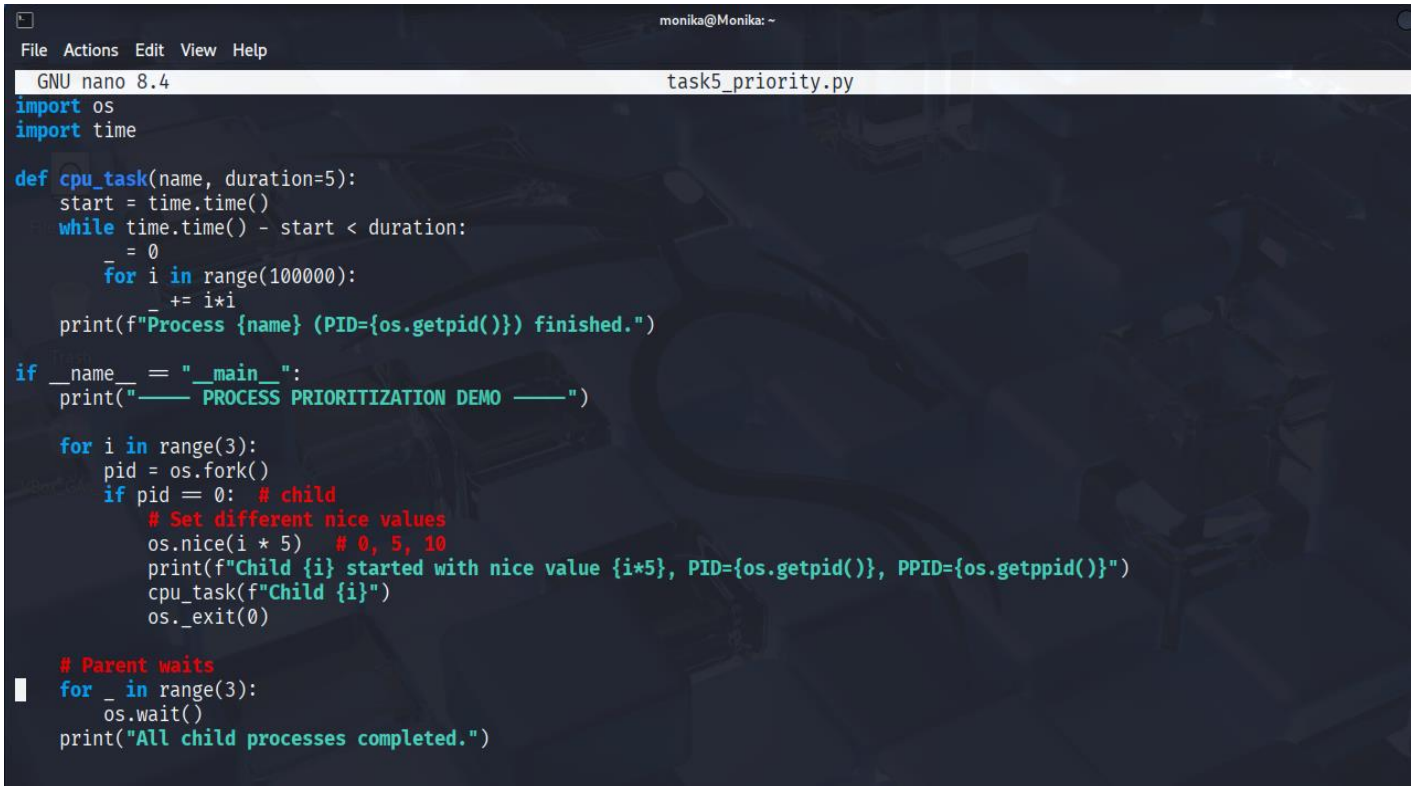


Output:

Task 5: Process Prioritization

Create multiple CPU-intensive child processes. Assign different nice() values. Observe and log execution order to show scheduler impact.

Code:



Output: