Report — Process Creation and Management (Python/Linux)

Objectives
- Simulate `fork`, `exec`, and process state inspections.
- Demonstrate zombie and orphan states.
- Read from `/proc`.
- Show scheduler impact via `nice`.

Method
One Python script with subcommands per task:
- Task 1: `os.fork`, `os.wait`.
- Task 2: `os.execvp` for real commands.
- Task 3: Controlled sleeps to expose `Z`ombies and orphans.
- Task 4: Read `/proc/[pid]/status`, `/exe`, `/fd`.
- Task 5: CPU-bound loops with varied `nice()` values.

Results (summarized)
- Children printed correct PID and PPID, parent reaped all.
- Exec replaced child images and produced system command outputs.
- Zombie visible as `<defunct>` until reaped. Orphan reparented to PID 1.
- `/proc` gave live metadata about state, memory, and FDs.
- Higher nice values finished slightly later under CPU contention.

Complexity
Time: O(n) children. Space: O(n) PIDs/logs.

Screenshots / Snapshots (text)
See `output.txt` logging for representative runs.

Conclusion
The script meets the assignment’s expected outputs and demonstrates core Linux process concepts.