Output analysis:

Test case	Number of processes created	Analysis
1	2 (parent + child)	Single fork() creates one child
2	3 (1 parent + 2 children)	Contains to separate fork() calls
3	2 (parent + child)	Only one fork() occurs
4	3 (one parent, two children)	A second form occurs after program 1 runs
5	1	Fork was not called

Analysis: the child always runs before the parent resumes, as outlined in the assignment that the child must have higher priority

Memory allocation and loading programs:

Test case	Loaded programs	Memory partitions in use	Analysis
1	Program1: 10MB Program2: 15MB	Partition 4 then partition 3	Larger program causes longer load time
2	Program1: 10MB	Partition 4, then 3, then 2	Test case demonstrated precess replacement
3	Program2 twice:15MB	Partition 4	I/O interrupts occur after execution begins
4	Program1: 10MB	Partition 4 then 3	Multiple syscalls and CPU bursts extend runtime
5	No program loaded	none	Only system calls and CPU bursts occur, no memory reallocation

Across the 5 test cases the simulation consistently reproduced expected IS process management behaviour. Fork() cloned PCB state and immediately scheduled the child. Exec() replaced the running process's memory image and updated PCB and partition entries. Program loading time depended on program size, and system calls followed the required kernel-mode transition steps. Test case 5 provides a contrast since it is a test case that doesn't create and processes or reassigns memory. The tests show the simulator behaves according to the assignment rules and expectations as proven by the outputs.

Github links:
https://github.com/Nhfaris627/SYSC4001_A2_P3

https://github.com/EnderAres183/SYSC4001_A2_P2