National University of Computer & Emerging Sciences
Department of Computer Science
Operating System Lab
Lab # 06

Process Creation Part-I

Instructions:

1. Make a word document with the convention "SECTION_ROLLNO _LAB-NO".
2. You have to submit a Word File containing your codes with comments and screenshots of their running outputs.
3. Plagiarism is strictly prohibited, negative marks would be given to students who cheat

Objectives – **Introduction of fork, exit(), wait(), waitpid(), getpid() and getppid()**

Problem statement 1– Write a program to create child processes from same parent using "n" fork() system calls where n should be greater than 2.

Problem statement 2– Write a code which accepts a command-line arguments n and forks a child process. The parent should calculate the product of numbers 1 to n and the child process should calculate the sum of numbers 1 to n. Separate your sum and product codes into functions.

Problem statement 3– The program declares a counter variable, set to zero, before fork()ing. After the fork call, we have two processes running in parallel, both incrementing their own version of counter. Each process will run to completion and exit. Because the processes run in parallel, we have no way of knowing which will finish first. Running this program will print something similar to what is shown below, though results may vary from one run to the next. For example:

Output:

```
--beginning of program
parent process: counter=1
parent process: counter=2
parent process: counter=3
child process: counter=1
parent process: counter=4
child process: counter=2
parent process: counter=5
```

```
child process: counter=3
--end of program--
child process: counter=4
child process: counter=5
--end of program—
```