



ASSIGNMENT # 6

NAME : SHAHERYAR ASHFAQ

ROLL NO : **20P-0128**

SECTION : BS-CS **4-B**

SUBJECT : OPERATING SYSTEM

main.py

```
1 from multiprocessing import Process
2 import os
3
4 def info(title):
5     print(title)
6     print('module name:', __name__)
7     print('parent process:', os.getppid())
8     print('process id:', os.getpid())
9
10 def f(name):
11     info('function f')
12     print('hello', name)
13
14 if __name__ == '__main__':
15     info('main line')
16
17     p1 = Process(target=f, args=('bob',))
18     p2 = Process(target=f, args=('lumber 1 bob',))
19     p3 = Process(target=f, args=('lumber 2 bob',))
20
21
22     p1.start()
23     p2.start()
24     p3.start()
25
26
27     p1.join()
28     p2.join()
29     p3.join()
30
31     print("Process p1 is alive: {}".format(p1.is_alive()))
32     print("Process p2 is alive: {}".format(p2.is_alive()))
33     print("Process p3 is alive: {}".format(p3.is_alive()))
```

```
main line
module name: __main__
parent process: 3901
process id: 3902
function f
module name: __main__
parent process: 3902
process id: 3906
hello bob
function f
module name: __main__
parent process: 3902
process id: 3907
hello lumber 1 bob
function f
module name: __main__
parent process: 3902
process id: 3908
hello lumber 2 bob
Process p1 is alive: False
Process p2 is alive: False
Process p3 is alive: False

...Program finished with exit code 0
Press ENTER to exit console.
```

PROCESS CLASS

Process class is an abstraction that sets up another Python process, provides it to run code and a way for the parent to control the start and termination of process { using start() & join() }. It works only on processors having multiple cores or system with multiple processors.

Visual Representation

