Aayush Kumar
CO21BTECH11002
OS2 Theory Assignment 1

1.
Recent CPU usages, for P1 = 40
for P2 = 18
for P3 = 10
Base = 60

Using the given function,
New priority for P1 = 40/2 + 60 = 80
for P2 = 18/2 + 60 = 69
for P3 = 10/2 + 60 = 65

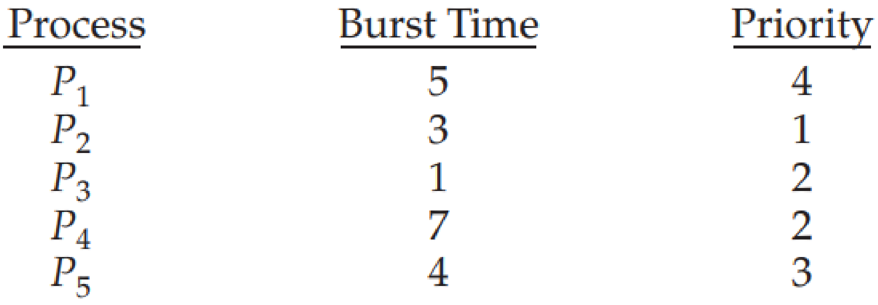
Since the values are decreasing, the scheduler will lower the relative priority of a CPU-bound process.

2.
Output at line C:- 5
Output at line P:- 0

Reason:-

The ‘value’ variable in not shared between parent and child process. The child process receives a copy of the ‘value’ variable. On the other hand, threads share the same memory space as the process in which they are created. Hence, the thread changes the value of child process’s ‘value’ variable to 5. But this change is not reflected in the ‘value’ variable of main process.

3.



Process	Burst Time	Priority
P_1	5	4
P_2	3	1
P_3	1	2
P_4	7	2
P_5	4	3

a)

FCFS:-

Each process is executed as they come.

P1	P2	P3	P4	P5	
0	5	8	9	16	20

SJF:-

Processes are executed based on their CPU burst time, with the process with smallest burst getting executed first.

P3	P2	P5	P1	P4	
0	1	4	8	13	20

Non-Preemptive Priority:-

Process with higher priority getting executed first.

P1	P5	P3	P4	P2	
0	5	9	10	17	20

RR:-

Each process is executed for a given time (quantum), then whichever process remains is executed for the rest duration.

P1	P2	P3	P4	P5	P1	P2	P4	P5	P1	P4	
0	2	4	5	7	9	11	12	14	16	17	20

b)

Avg execution time = (5+3+1+7+4)/5 = 4

Avg waiting time for FCFS = (0+5+8+9+16)/5 = 7.6
Turnaround for FCFS = 4+7.6 = 11.6

Avg waiting time for SJF = (0+1+4+8+13)/5 = 5.2
Turnaround for SJF = 4+5.2 = 9.2

Avg waiting time for Non-Preemptive Priority = (0+5+9+10+17)/5 = 8.2
Turnaround for Non-Preemptive Priority = 4+8.2 = 12.2

Avg waiting time for RR = (14+9+4+15+12)/5 = 10.8
Turnaround for FCFS = 4+10.8 = 14.8

c)
Waiting time in FCFS, for P1 = 0
for P2 = 5
for P3 = 8
for P4 = 9
for P5 = 16

Waiting time in SJF, for P1 = 8
for P2 = 1
for P3 = 0
for P4 = 13
for P5 = 4

Waiting time in Non-Preemptive Priority, for P1 = 0
for P2 = 17
for P3 = 9
for P4 = 10
for P5 = 5

Waiting time in FCFS, for P1 = 0+7+7 = 14
for P2 = 2+7 = 9
for P3 = 4
for P4 = 5+5+5 = 15
for P5 = 7+5 = 12

d)
From the results in part b, we can see that SJF algorithm results in minimum average time.

4.

a)
For $\beta > \alpha > 0$, The process that is already running will continue to have higher priority, hence first it will finish and only then other processes will start. Also, the process that has been waiting for the longest time will start first since all the waiting processes gain priority at the same rate. This essentially results in a First Come First Serve (FCFS) algorithm.

b)
For $\alpha < \beta < 0$, The process that is running as well as the processes that are waiting both will have reduction in their priorities. But the running process's priority will reduce at a lower rate, hence it will continue to run until a new process arrives.

Whenever a new process arrives, it will start executing the and currently running process will go into waiting list. Also the process that has been waiting the longest will have lowest priority, hence it will be executed at last. This results in a Last Come First Serve algorithm.

5.

p1 = 50, p2 = 75, t1 = 25, t2 = 30

a)
Consider P1 to have higher priority, then, P1 will start and run till 25. Then P2 will start at 25 and continue to run. When P1 arrives again at 50, it will have higher priority and thus will start running and run till 75. This way, P2 will not be able to meet its deadline.

P1	P2	P1	P2
0	25	50	75

Now, consider P1 to have lower priority, then, P2 will start and run till 30. Then P1 will start at 30 and finish at 55. This way, P1 will not be able to meet its deadline.

P2	P1	
0	30	55

Thus it is not possible to schedule these processes using rate-monotonic scheduling such that they both meet the deadlines.

b)
At start, P1 will have higher priority as its deadline is 50. P1 will run till 25. At 25 P2 starts. P1 will come at 50 but it will have lower priority as its deadline will be 100. P2 will continue to run and finish at 55. P1 will start at 55 ans run till 80. P2 arrives at 75 but it will have lower priority as its deadline will be 150. After P1 completes at 80, P2 will start and run till 110. P1 arrives at 100 but have the same deadline as P2, so, P2 continues to run. P1 starts at 110 and runs till 135. Now, the CPU remains idle till 150 when both the processes arrive and the cycle continues.

P1	P2	P1	P2	P1		
0	25	55	80	110	135	150