Theory Assignment 1

Aaryan, CO21BTECH11001

Q.1

Ans.

Given : base = 60

Recent CPU usage : P1 = 40, P2 = 18, P3 = 10

$priority\ =\ (recent\ CPU\ usage/2\ +\ base)$

New priority numbers will be:

$P1\ =\ 40/2\ +\ 60\ =\ 80$

$P2\ =\ 18/2\ +\ 60\ =\ 69$

$P3\ =\ 10/2\ +\ 60\ =\ 65$

Since P1 had maximum CPU usage, so P1 was most CPU bound process. Since there is an inverse relationship between priority and priority number, therefore, UNIX scheduler decreases relative priority of CPU bound process.

Q.2

Ans.

The output of LINE C will be *“CHILD : value = 5”*
The output of LINE P will be *“PARENT : value = 0”*

Initially, a global variable “value” is created and initialized with 0. When the program enters the main function, all thread variables are declared and a child process is created with fork process.
Fork creates a child process which is assigned a copy of all the variables of parent process. Therefore, the value of variable “value” will be 0 when parent process is executed. pid value of parent process is more than 0. Therefore when

it enters the condition *if (pid > 0),* it waits for the child process to complete and then prints the variable “value”, which is 0.
When child process is executed, a new thread is created by the process. When *runner* function of thread is called, it changes value of variable “value” to 5. Therefore, when child process prints the variable “value”, it prints 5.

Q.3

Ans.

Process	Burst Time	Priority
P1	5	4
P2	3	1
P3	1	2
P4	7	2
P5	4	3

a. Gantt Charts are following:

FCFS:

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Process	P1	P1	P1	P1	P1	P2	P2	P2	P3	P4	P4	P4	P4	P4	P4	P4	P5	P5	P5	P5

SJF:

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Process	P3	P2	P2	P2	P5	P5	P5	P5	P1	P1	P1	P1	P1	P4	P4	P4	P4	P4	P4	P4

Non - Preemptive Priority :

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Process	P1	P1	P1	P1	P1	P5	P5	P5	P5	P3	P4	P4	P4	P4	P4	P4	P4	P2	P2	P2

RR:

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Process	P1	P1	P2	P2	P3	P4	P4	P5	P5	P1	P1	P2	P4	P4	P5	P5	P1	P4	P4	P4

b. Turnaround time = Completion time - Arrival time

FCFS:

Process	P1	P2	P3	P4	P5
Turnaround Time	5	8	9	16	20

SJF:

Process	P1	P2	P3	P4	P5
Turnaround Time	13	4	1	20	8

Non - Preemptive Priority :

Process	P1	P2	P3	P4	P5
Turnaround Time	5	20	10	17	9

RR:

Process	P1	P2	P3	P4	P5
Turnaround Time	17	12	5	20	16

c. Waiting time = Turn around time - Burst time

FCFS:

Process	P1	P2	P3	P4	P5
Waiting Time	0	5	8	9	16

SJF:

Process	P1	P2	P3	P4	P5
Waiting Time	8	1	0	13	4

Non - Preemptive Priority :

Process	P1	P2	P3	P4	P5
Waiting Time	0	17	9	10	5

RR:

Process	P1	P2	P3	P4	P5
Waiting Time	12	9	4	13	12

d. Average waiting time for :

$FCFS = (0 + 5 + 8 + 9 + 16)/5 = 7.6$
$SJF = (8 + 1 + 0 + 13 + 4)/5 = 5.2$
$NPP = (0 + 17 + 9 + 10 + 5)/5 = 8.2$
$RR = (12 + 9 + 4 + 13 + 12)/5 = 10$
Therefore, SJF results in shortest waiting time.

Q. 4

Ans.

Rate of priority change of a running process = $\boldsymbol{\alpha}$
Rate of priority change of a process which is waiting = $\boldsymbol{\beta}$

a. $\boldsymbol{\beta} > \boldsymbol{\alpha} > 0$
   In this case, priority of the process which is running will increase faster than the process which is waiting. Most likely, the process which will arrive first will have the maximum priority. Therefore, this situation is equivalent to First Come First Serve (FCFS).

b. $\boldsymbol{\alpha} < \boldsymbol{\beta} < 0$
   In this case, the priority of the process which is waiting will decrease faster than the process which is running. Therefore, if a new process arrives, then it is most likely to have a higher priority than any other process (running as well as waiting), and it will start executing. Therefore, this situation is equivalent to Last In First Out (LIFO).

Q.5

Ans.

$P_1 : p_1 = 50,\ t_1 = 25$
$P_2 : p_2 = 75,\ t_2 = 30$

a. Let's consider priority of P1 is more than P2. So, P1 will start at time 0 and finish at time 25. Then, P2 will start executing and continue till 50, because P1 will again start executing at time 50. Now, P1 will end at time 75 and first execution of P2 is still left. Therefore, P2 can't meet its deadline.

   Now, let's consider priority of P2 is more than P1. So, P2 will start at time 0 and finish at time 30. Then, P1 will start executing and finish at 55. Therefore, P1 can't meet its deadline.
   In both cases, we can't schedule these using rate-monotonic scheduling.

b. Deadline of P1 is 50 and that of P2 is 75.
   So, if we schedule processes using earliest deadline first, then P1 will be scheduled first.
   So, P1 will start at time 0 and finish at time 25.
   New deadline of P1 is 100 and that of P2 is 75.

   So, P2 will start executing and continue till 55.

New deadline of P2 is 150 and that of P1 is 100.

So, P1 will start at time 55 and end at 80.
New deadline of P1 as well as P2 is 150.
This will continue.