Assignment - 1
Operating System
KCS-401

Abhay Shanker Pathak
United Institute of Technology, Prayagraj
IT-Department

March 2020

About

This assignment contains questions and thier solutions
All the questions are categorized sectionwise

Contents

List of Tables

List of Figures

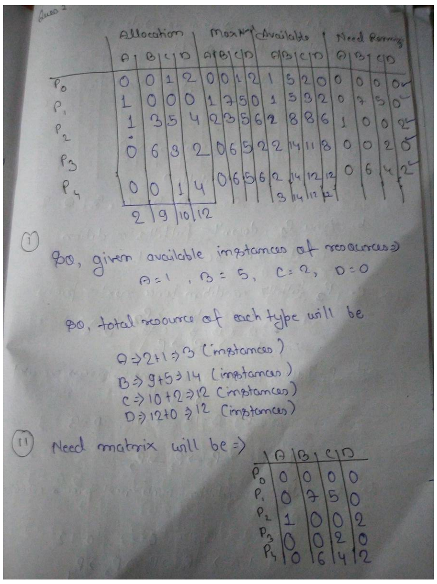
1 Question

Consider the following snapshot at time T0 of the system and answer the following questions using Banker's algorithm

1. **Compute the total number of resource of each type.**
2. **Compute the need matrix**
3. **Is the system in a safe mode?**
4. **If a request from P1 arrives for (0, 4, 2, 0), can the request granted immediately?**

	Allocation	**Max**	**Available**
	a b c d	a b c d	a b c d
P0	0 0 1 2	0 0 1 2	1 5 2 0
P1	1 0 0 0	1 7 5 0	
P2	1 3 5 4	2 3 5 6	
P3	0 6 3 2	0 6 5 2	
P4	0 0 1 4	0 6 5 6	

Table 1: data table, banker's algorithm



Ques 1

	Allocation A B C D	Max Need A B C D	Available A B C D	Need Remaining A B C D
P_0	0 0 1 2	0 0 1 2	1 5 2 0	0 0 0 0
P_1	1 0 0 0	1 7 5 0	1 5 3 2	0 7 5 0
P_2	1 3 5 4	2 3 5 6	2 8 8 6	1 0 0 2
P_3	0 6 3 2	0 6 5 2	2 14 11 8	0 0 2 0
P_4	0 0 1 4	0 6 5 6	2 14 12 12 3 14 12 12	0 6 4 2
	2 9 10 12			

(i) So, given available instances of resources ⇒
A = 1, B = 5, C = 2, D = 0

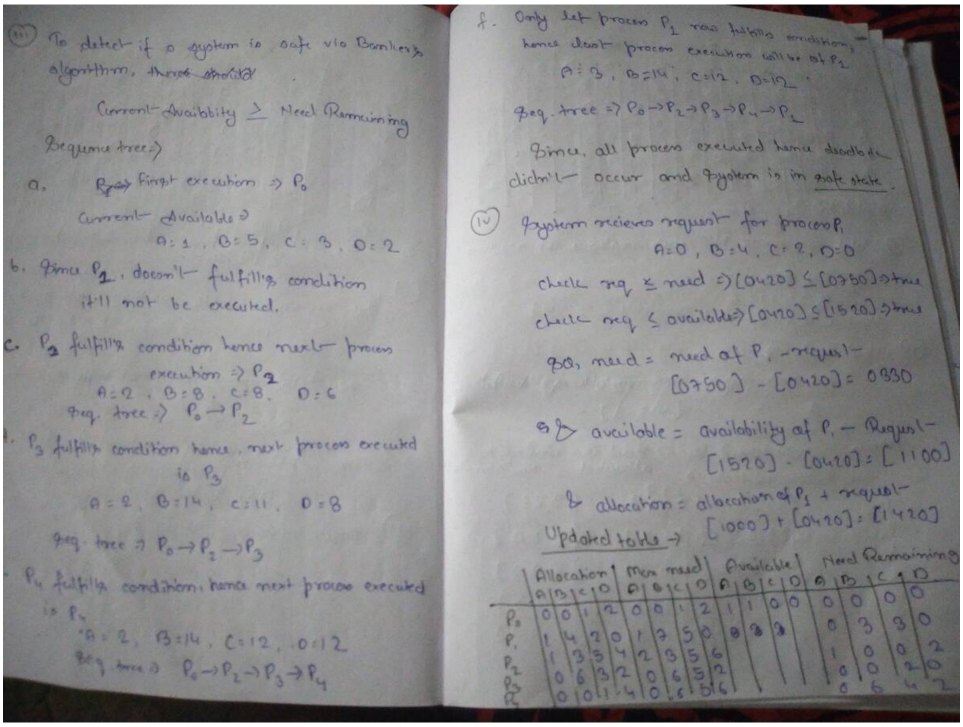
So, total resource of each type will be

A ⇒ 2+1 ⇒ 3 (instances)
B ⇒ 9+5 ⇒ 14 (instances)
C ⇒ 10+2 ⇒ 12 (instances)
D ⇒ 12+0 ⇒ 12 (instances)

(ii) Need matrix will be ⇒

	A	B	C	D
P_0	0	0	0	0
P_1	0	7	5	0
P_2	1	0	0	2
P_3	0	0	2	0
P_4	0	6	4	2

Figure 1: page 1 of solution for ques 1



(iii) To detect if a system is safe via Banker's algorithm, ~~thrust should~~

Current Availbity $\geq$ Need Remaining

Sequence tree $\Rightarrow$

a. ~~P_1~~ First execution $\Rightarrow P_0$

Current Available $\Rightarrow$

$A = 1,\ B = 5,\ C = 3,\ D = 2$

b. Since P_1, doesn't fulfilling condition it'll not be executed.

c. P_2 fulfilling condition hence next process execution $\Rightarrow P_2$

$A = 2,\ B = 8,\ C = 8,\ D = 6$

Seq. tree $\Rightarrow P_0 \rightarrow P_2$

d. P_3 fulfills condition hence, next process executed is P_3

$A = 2,\ B = 14,\ C = 11,\ D = 8$

Seq. tree $\Rightarrow P_0 \rightarrow P_2 \rightarrow P_3$

e. P_4 fulfills condition, hence next process executed is P_4

$A = 2,\ B = 14,\ C = 12,\ D = 12$

Seq. tree $\Rightarrow P_0 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4$

f. Only left process P_1 now fulfills condition, hence last process execution will be of P_1

$A = 3,\ B = 14,\ C = 12,\ D = 12$

Seq. tree $\Rightarrow P_0 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow P_1$

Since, all process executed hence deadlock didn't occur and system is in safe state.

(iv) System recieves request for process P_1

$A = 0,\ B = 4,\ C = 2,\ D = 0$

check req $\leq$ need $\Rightarrow [0420] \leq [0750] \Rightarrow$ true

check req $\leq$ available $\Rightarrow [0420] \leq [1520] \Rightarrow$ true

So, need = need of P_1 − request

$[0750] - [0420] = 0330$

& available = availability of P_1 − Request

$[1520] - [0420] = [1100]$

& allocation = allocation of P_1 + request

$[1000] + [0420] = [1420]$

Updated table $\rightarrow$

	Allocation				Max need				Available				Need Remaining			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	1	0	0	0	0	0	0
P_1	1	4	2	0	1	7	5	0	[illegible]	[illegible]	[illegible]		0	3	3	0
P_2	1	3	5	4	2	3	5	6					1	0	0	2
P_3	0	6	3	2	0	6	5	2					0	0	2	0
P_4	0	0	1	4	0	6	5	6					0	6	4	2

Figure 2: page 2 of solution for ques 1

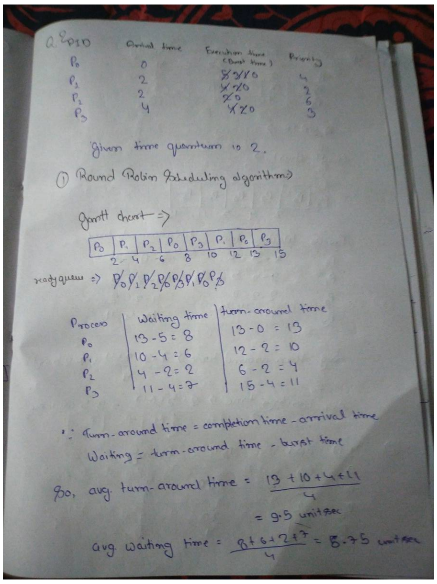
2 Question

Consider the set of process given in the table and draw the Gantt Chart for preemptive cases and find out the average waiting time, average turnaround time, throughput and CPU utilization for following scheduling algorithm.
Note: Larger priority number has higher priority

- **a Round Robin**
- **b SRT**
- **c Priority**

Process ID	Arrival time	Execution time	Priority
P0	0	5	4
P1	2	4	2
P2	2	2	6
P3	4	4	3

Table 2: Details to process



Q.2

PID	Arrival time	Execution time (Burst time)	Priority
P_0	0	$\cancel{5}\,\cancel{3}\,\cancel{1}\,0$	4
P_1	2	$\cancel{4}\,\cancel{2}\,0$	2
P_2	2	$\cancel{2}\,0$	6
P_3	4	$\cancel{4}\,\cancel{2}\,0$	3

Given time quantum is 2.

(1) Round Robin Scheduling algorithm:

Gantt chart =>

P_0	P_1	P_2	P_0	P_3	P_1	P_0	P_3
2	4	6	8	10	12	13	15

ready queue => $\cancel{P_0}\,\cancel{P_1}\,\cancel{P_2}\,\cancel{P_0}\,\cancel{P_3}\,\cancel{P_1}\,\cancel{P_0}\,\cancel{P_3}$

Process	Waiting time	turn-around time
P_0	13 - 5 = 8	13 - 0 = 13
P_1	10 - 4 = 6	12 - 2 = 10
P_2	4 - 2 = 2	6 - 2 = 4
P_3	11 - 4 = 7	15 - 4 = 11

∵ Turn-around time = completion time - arrival time

Waiting = turn-around time - burst time

So, avg. turn-around time = $\frac{13+10+4+11}{4}$

= 9.5 unit sec

avg. waiting time = $\frac{8+6+2+7}{4}$ = 5.75 unit sec

Figure 3: page 1 of solution for ques 2

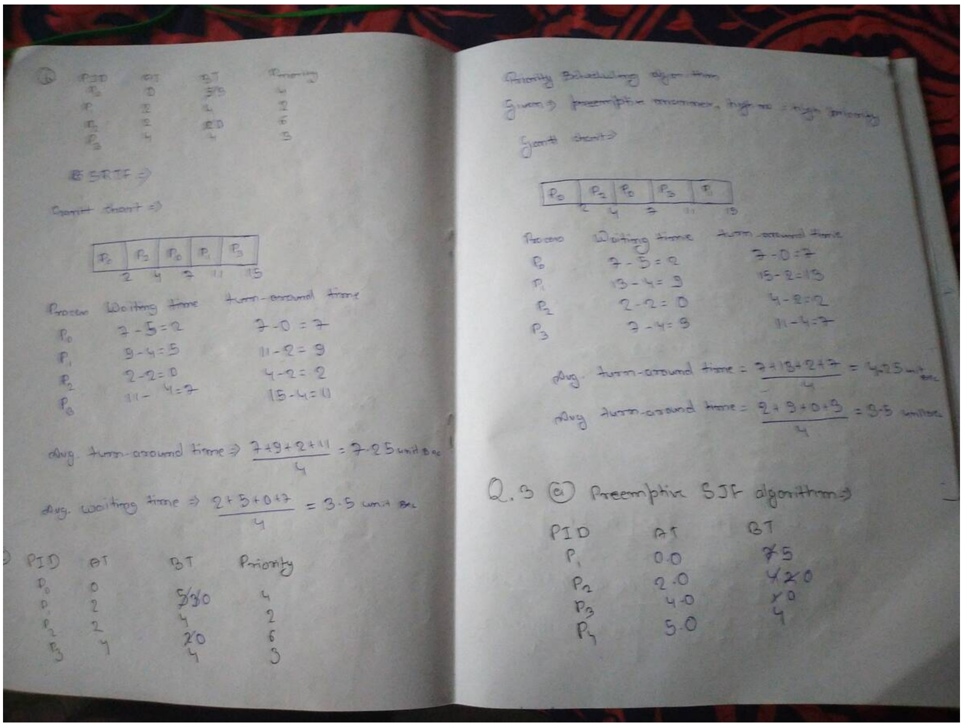


Figure 4: page 2 of solution for ques 2

3 Question

Consider Non-preemptive and preemptive SJF algorithm; find out average waiting time in both.

Process ID	Arrival Time	Burst Time
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

Table 3: Data to solve

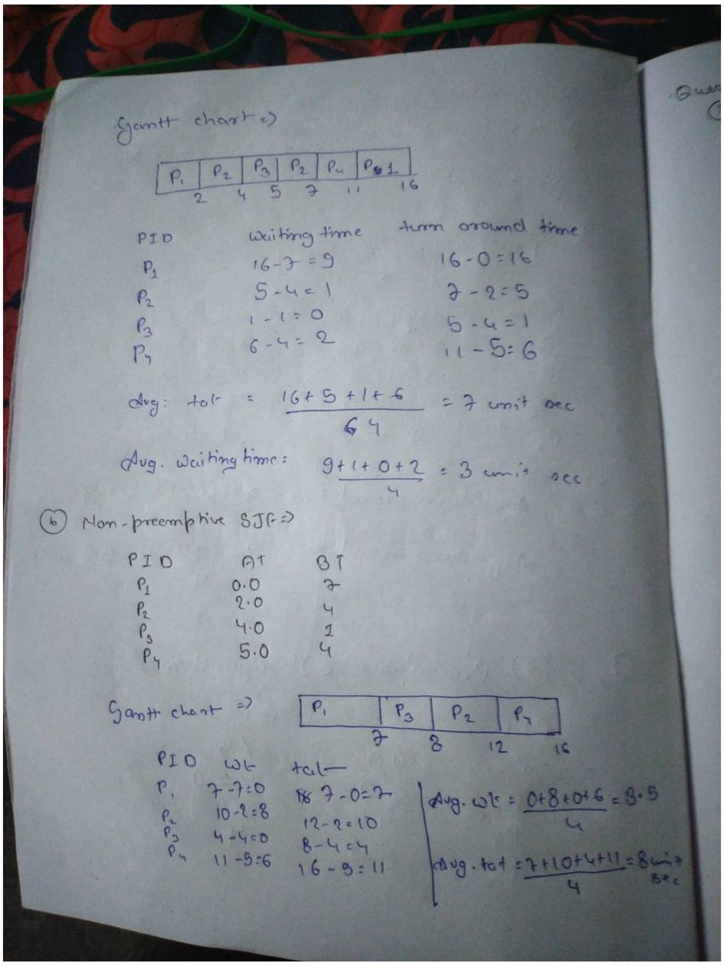


Figure 5: page 1 of solution for ques 3

4 Question

Consider the following set of processes:

Process ID	Arrival Time	Burst Time	Priority
P1	0	6	3
P2	1	4	1
P3	2	5	2
P4	3	8	4

Table 4: Data chart to process

(i) **Draw the Gantt charts and find the average waiting time and average turnaround time using SRTF, Round robin(time quantum=3) and preemptive priority scheduling.**

(ii) **If the schedulers takes 0.2 unit of CPU time in Context Switching, calculate the percentage of CPU time wasted in each case.**

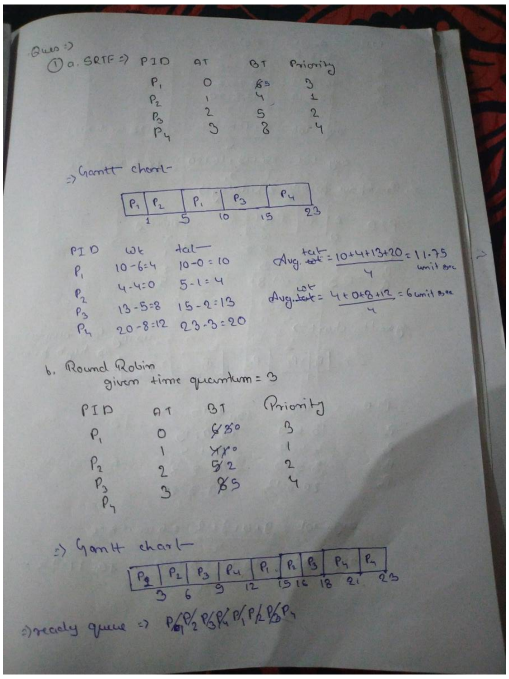


Figure 6: page 1 of solution for ques 4

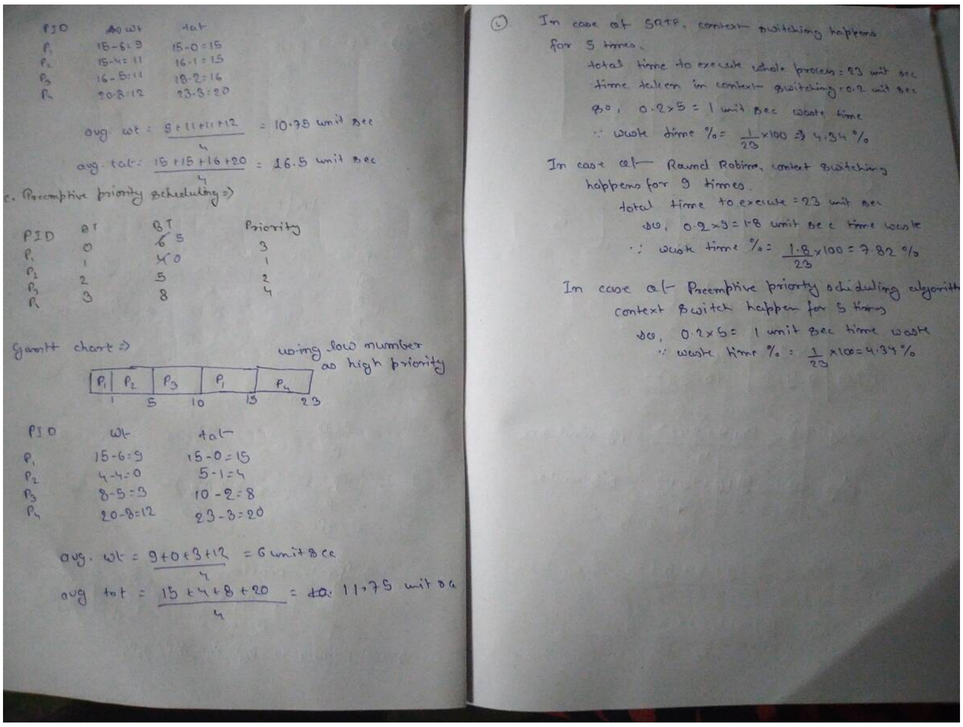


Figure 7: page 2 of solution for ques 4

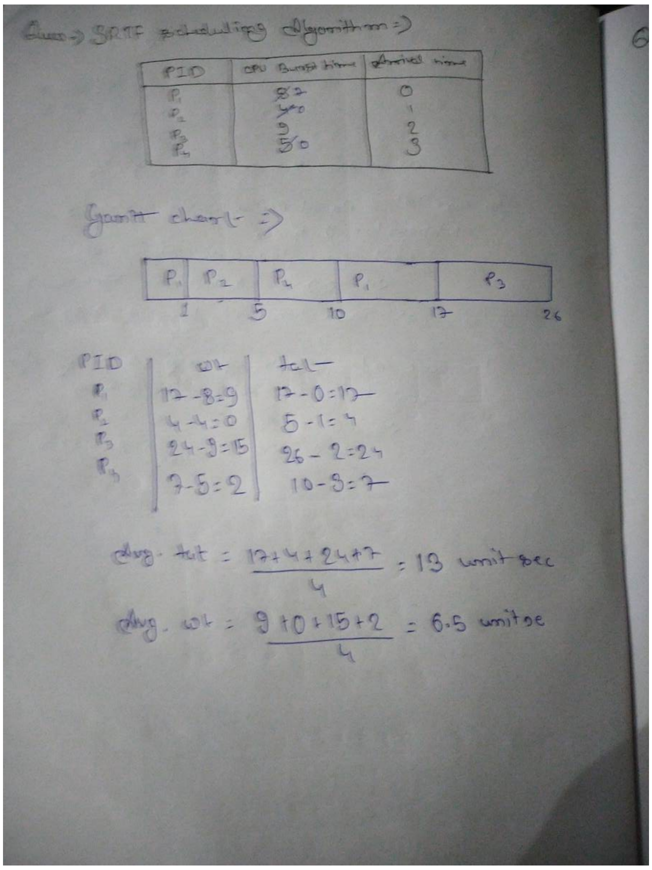
5 Question

Consider following set of process, CPU burst time and Arrival time given below:

Process ID	**Arrival Time**	**Burst Time**
P1	8	0
P2	4	1
P3	9	2
P4	5	3

Table 5: Provided Data to solve

Draw the Gantt charts and find the average waiting time and average turnaround time using SRTF CPU scheduling algorithm.


Ques ⇒ SRTF scheduling Algorithm :⇒

PID	CPU Burst time	Arrival time
P_1	~~8~~ 7	0
P_2	~~4~~ 0	1
P_3	9	2
P_4	~~5~~ 0	3

Gantt chart :⇒

P_1	P_2	P_4	P_1	P_3
1	5	10	17	26

PID	wt	tat
P_1	17 - 8 = 9	17 - 0 = 17
P_2	4 - 4 = 0	5 - 1 = 4
P_3	24 - 9 = 15	26 - 2 = 24
P_4	7 - 5 = 2	10 - 3 = 7

$$\text{Avg. tat} = \frac{17+4+24+7}{4} = 13 \text{ unit sec}$$

$$\text{Avg. wt} = \frac{9+0+15+2}{4} = 6.5 \text{ unit se}$$

Figure 8: page 1 of solution for ques 5

6 Question

List various performance criteria for scheduling algorithms. Five process A, B, C, D and E require CPU burst 3, 5, 2, 5 and 5 units respectively. Their arrival times in the system are 0, 1, 3, 9 and 12 respectively. Draw Gantt chart and compute the average turnaround time and average waiting time of these processes for the Shortest Job First(SJF) and Shortest Remaining Time First(SRTF).

There are several different criteria to consider when trying to select the "best" scheduling algorithm for a particular situation and environment, including:

6.0.1 CPU utilization -

Ideally the CPU woulb be busy 100% of the time, so as to waste 0 CPU cycles. On a real system CPU usage should range from 40% (lightly load) to 90 % (heavily loaded.)

6.0.2 Throughput -

Number of process completed per unit time. May range from 10 per second to 1 hour depending on the specific processes.

6.0.3 Turnaround time -

Time required for a particular process to complete, from submission time to completion. (Wall clock time.)

6.0.4 Waiting time -

How much time processes spen in the ready queue waiting their turn to get on the CPU.

6.0.5 Load average -

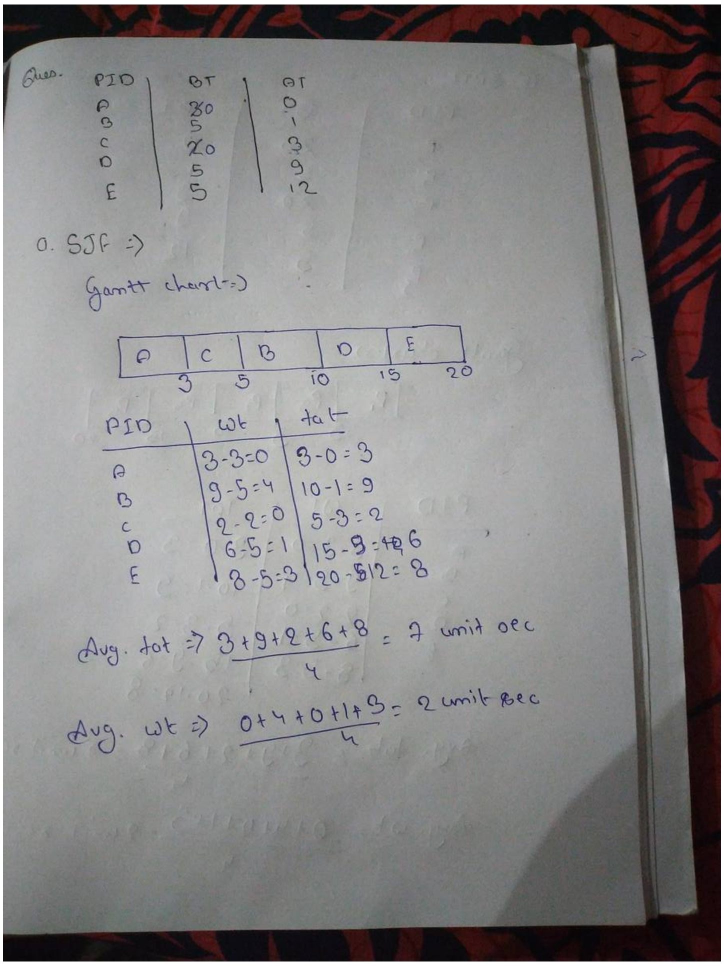
The average number of processes sitting in the ready queue waiting their turn to get into the CPU. Reported in 1-minute, 5-minute, and 15-minute averages(number aren't unique) by "uptime" and "who".

6.0.6 Response time -

The time taken in an interactive program from the issuance of a command to the *commence* of a response to that command.

In general one wants to optimize the average value of a criteria (Maximize CPU utilization and throughput, and minimize all the others.) However some times one wants to do something different, such as to minimize the maximum response time.

Sometimes it is most desirable to minimize the variance of a criteria than the actual value, i.e., users are more accepting of a consistent predictable system than an inconsistent one, even if it is a little bit slower.


Ques.

PID	BT	AT
A	30	0
B	5	1
C	20	3
D	5	9
E	5	12

a. SJF =>

Gantt chart =>

A	C	B	D	E

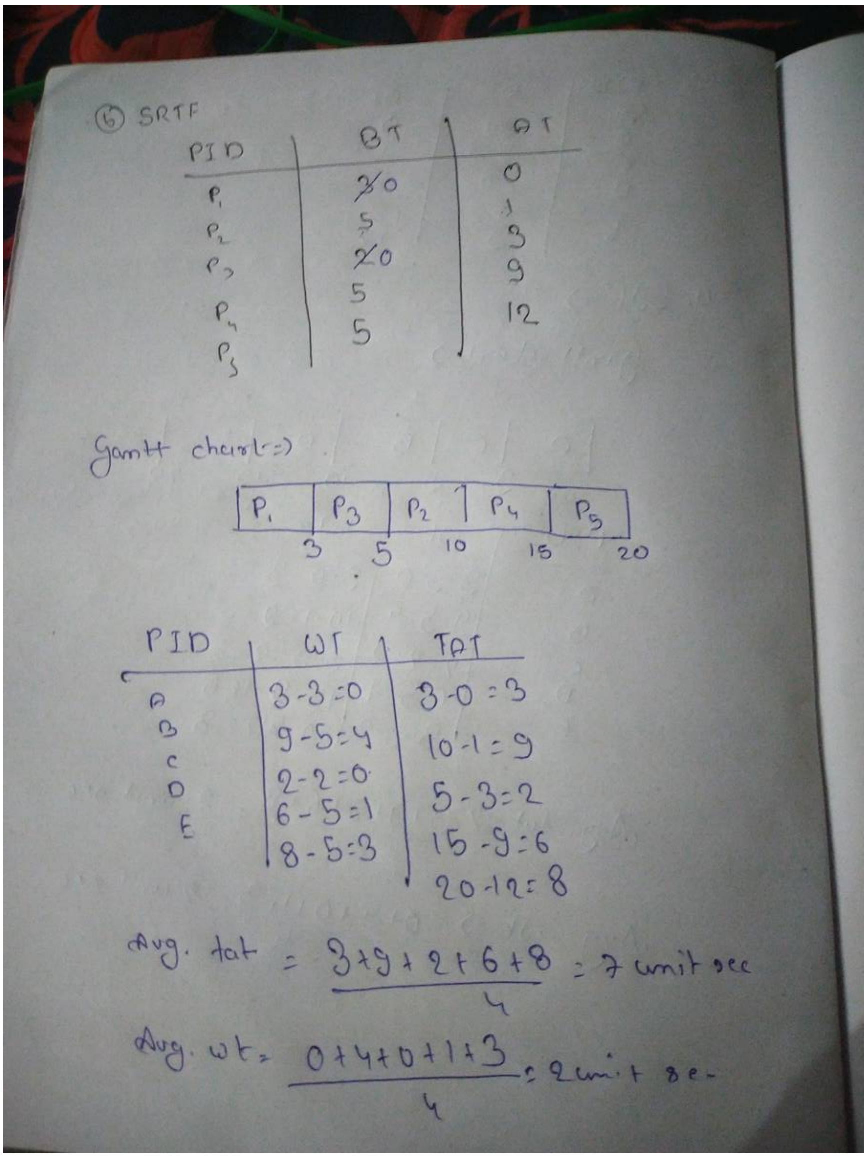
3 5 10 15 20

PID	wt	tat
A	3-3=0	3-0=3
B	9-5=4	10-1=9
C	2-2=0	5-3=2
D	6-5=1	15-9=6
E	8-5=3	20-12=8

Avg. tat => $\frac{3+9+2+6+8}{4}$ = 7 unit sec

Avg. wt => $\frac{0+4+0+1+3}{4}$ = 2 unit sec

Figure 9: page 1 of solution for ques 6



⑥ SRTF

PID	BT	AT
P_1	~~3~~0	0
P_2	5	1
P_3	~~2~~0	3
P_4	5	9
P_5	5	12

Gantt chart :-)

P_1	P_3	P_2	P_4	P_5
3	5	10	15	20

PID	WT	TAT
A	3-3=0	3-0=3
B	9-5=4	10-1=9
C	2-2=0	5-3=2
D	6-5=1	15-9=6
E	8-5=3	20-12=8

Avg. tat $= \frac{3+9+2+6+8}{4} = 7$ unit sec

Avg. wt $= \frac{0+4+0+1+3}{4} = 2$ unit se-

Figure 10: page 2 of solution for ques 6

7 Question

Describe the banker's algorithm for safe allocation. Consider a system with five process and three resource types and at time 'T' the following snapshot of the system has been taken:

	Allocated			Maximum			Available		
PID	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	1	1	2	4	3	3	3	1	0
P2	2	1	2	3	2	2			
P3	4	0	1	9	0	2			
P4	0	2	0	7	5	3			
P5	1	1	2	11	2	3			

Table 6: Data to solve with Banker's Algorithm

(i) **Determine the total amount of resources of each type.**

(ii) **Compute the need matrix.**

(iii) **Determine the state is safe or not using Banker's algorithm.**

(iv) **Would the following request be granted in the current state?**

  (a) **P1<3, 3, 1>**

  (b) **P2<2, 1, 0>**

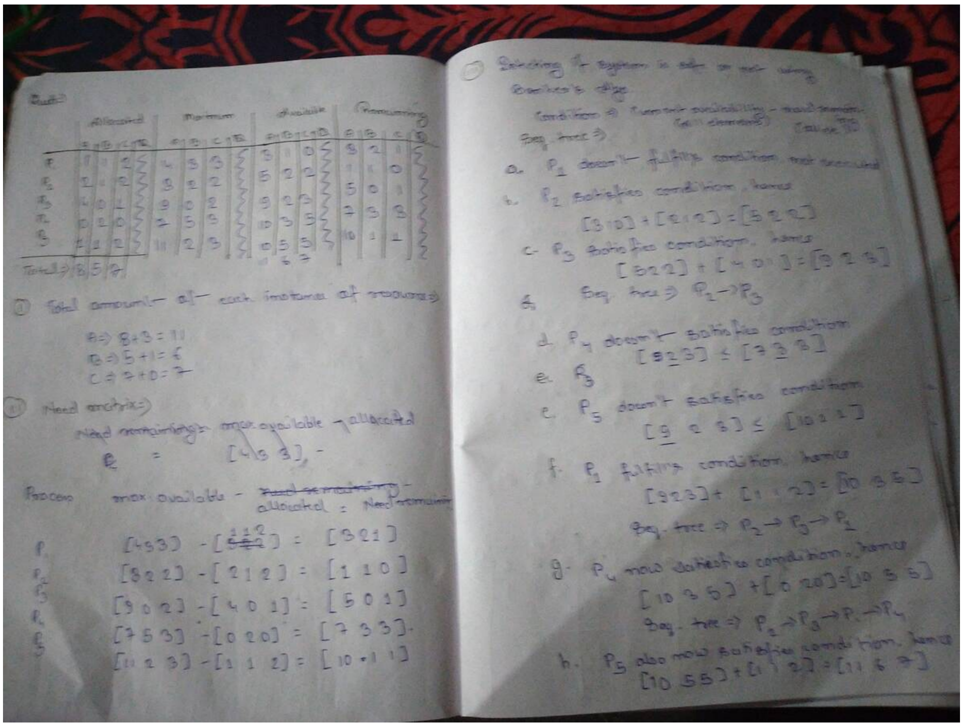
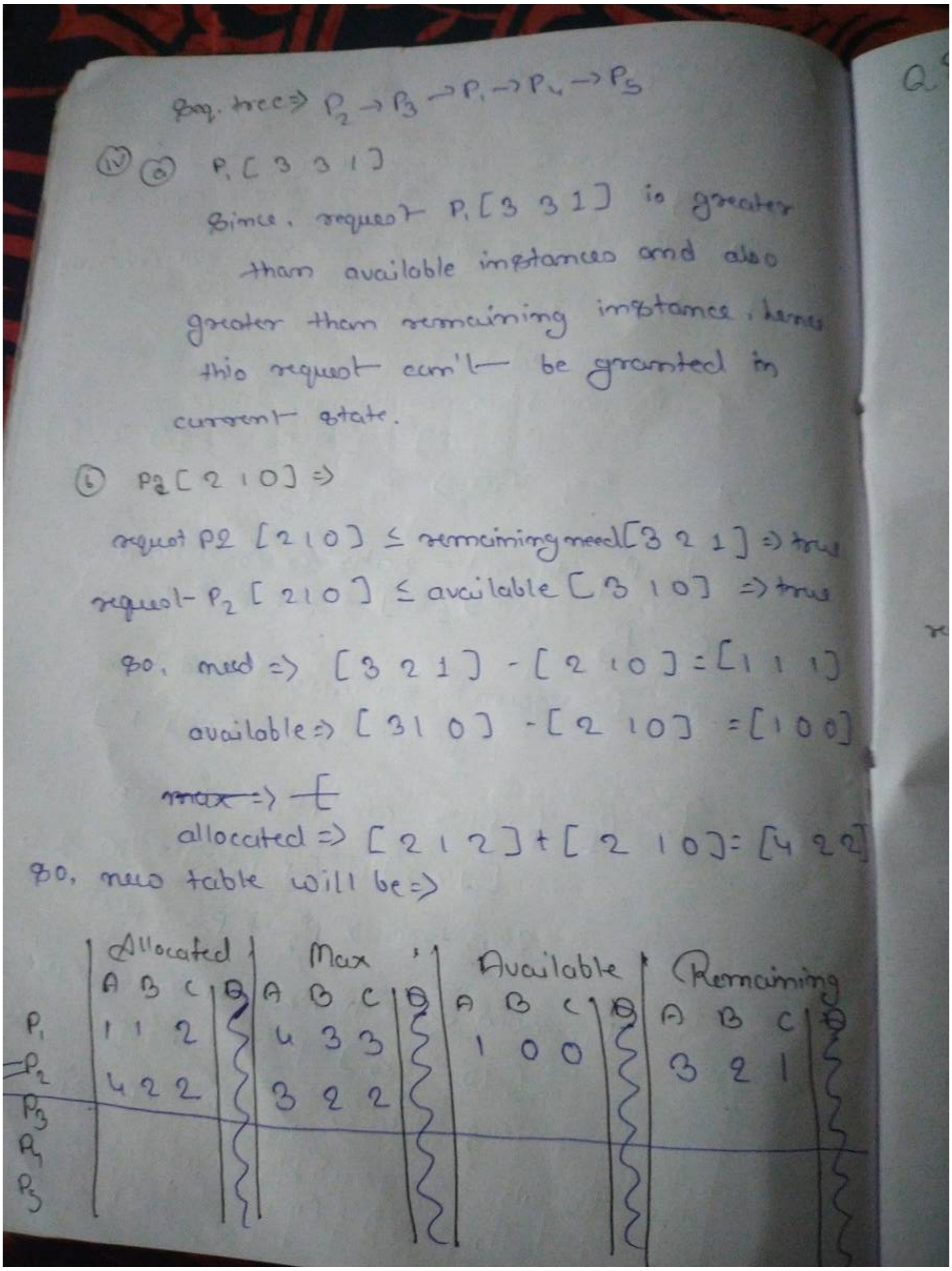


Figure 11: page 1 of solution for ques 7



Seq. tree ⇒ $P_2 \to P_3 \to P_1 \to P_4 \to P_5$

(iv) (a) P_1 [3 3 1]

Since, request P_1 [3 3 1] is greater than available instances and also greater than remaining instance, hence this request can't be granted in current state.

(b) P_2 [2 1 0] ⇒

request P2 [2 1 0] ≤ remaining need [3 2 1] ⇒ true

request P_2 [2 1 0] ≤ available [3 1 0] ⇒ true

So, need ⇒ [3 2 1] − [2 1 0] = [1 1 1]

available ⇒ [3 1 0] − [2 1 0] = [1 0 0]

~~max~~ ⇒ [

allocated ⇒ [2 1 2] + [2 1 0] = [4 2 2]

So, new table will be ⇒

	Allocated A	B	C	Max A	B	C	Available A	B	C	Remaining A	B	C
P_1	1	1	2	4	3	3	1	0	0	3	2	1
P_2	4	2	2	3	2	2						
P_3												
P_4												
P_5												

Figure 12: page 2 of solution for ques 7