CS433 Written Homework 3

(50 points) All question numbers refer to exercises in the textbook (10th edition). Make sure you use the right textbook and answer the right questions. Students must finish written questions individually. Type your answers and necessary steps clearly.

1. **(5 points) 5.11** Of these two types of programs:

   a. I/O-bound

   b. CPU-bound

   which is more likely to have voluntary context switches, and which is more likely to have nonvoluntary context switches? Explain your answer.

   A CPU-bound program is more likely to have more nonvoluntary context switches, because it's design to maximize the processing of processes, and it suffers context switches when higher-priority processes adhere the queue or when time slices expire. An I/O-bound progam is more likely to have voluntary context switches, once the CPU suffers context switches to access resorces.

2. **(10 points) 5.12.** Discuss how the following pairs of scheduling criteria conflict in certain settings.

   a. CPU utilization (efficiency) and response time

   For a program to have a better response time, round-robin-like algorithms are implemented because a time quantum is defined, which results in a better response time; but with it, it comes more context switches, that will lead to less CPU efficiency, once more time is spent in switching between processes.

   b. Average turnaround time and maximum waiting time

   The smallest average turnaround time influences the maximum waiting time, which will be the smallest waiting time for a set of processes. Basically, both averages are related in a way that, for a set of processes, arranged in a way to increase the average turnaround time to the maximum, the waiting time for this same set will be the greatest it can be: they are directly proportional.

   c. I/O device utilization and CPU utilization

   There will be more CPU utilization on a processor if there is less I/O device utilization. With more context switches, there will be an increase on I/O device utilization and that will affect the amout of CPU utilization, reducing it. CPU utilization will be greater with more processing and less context switches.

3. **(5 points) 5.15.** Consider the exponential average formula used to predict the length of the next CPU burst.

What are the implications of assigning the following values to the parameters used by the algorithm?

a. $\alpha = 0$ and $\tau_0 = 100$ *milliseconds*

b. $\alpha = 0.99$ and $\tau_0 = 10$ *milliseconds*

In scenario a, we would have an algorithm that only takes into account the value of the most old prediction, which would be 100 milliseconds; In scenario b, we would have an algorithm that values most the burst time length of the most recent process, without considering the previous processes' burst-time history.

4. **(10 points) 5.17.** Consider the following set of processes, with the length of the CPU-burst time given in milliseconds:

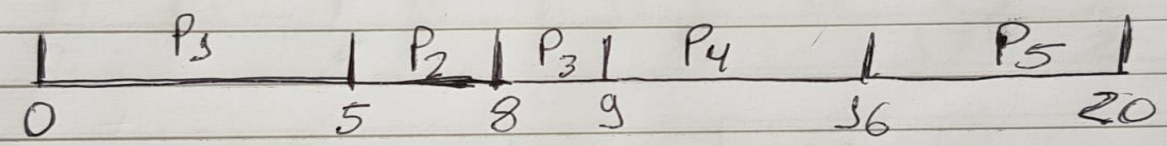
Process	Burst Time	Priority
*P*1	5	4
*P*2	3	1
*P*3	1	2
*P*4	7	2
*P*5	4	3

The processes are assumed to have arrived in the order *P*1, *P*2, *P*3, *P*4, *P*5, all at time 0.

a. Draw four Gantt charts illustrating the execution of these processes using FCFS, SJF, a non-preemptive priority (a larger priority number implies a higher priority), and RR (quantum = 2) scheduling.
b. What is the turnaround time of each process for each of the scheduling algorithms in part a?
c. What is the waiting time of each process for each of the scheduling algorithms in part a?
d. Which of the schedules in part a results in the minimal average waiting time (over all processes)?

FCFS

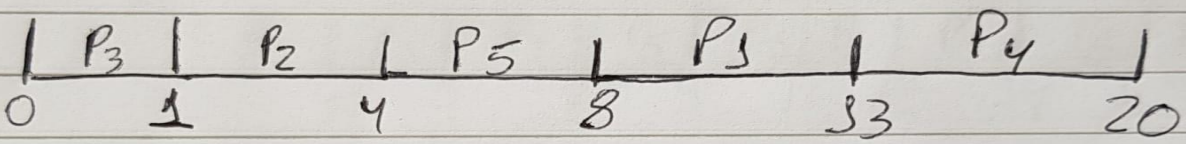
Queue: P_1 (Head), P_2, P_3, P_4, P_5



	Turnaround Time	Waiting Time
P_1	5	0
P_2	8	5
P_3	9	8
P_4	16	9
P_5	20	16

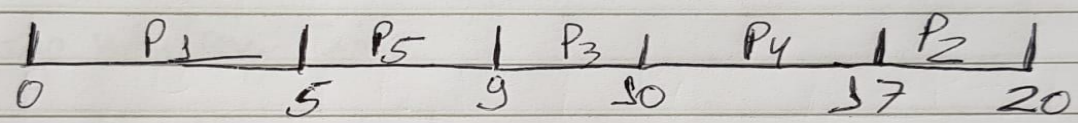
SJF

Queue: P_3 (Head), P_2, P_5, P_1, P_4



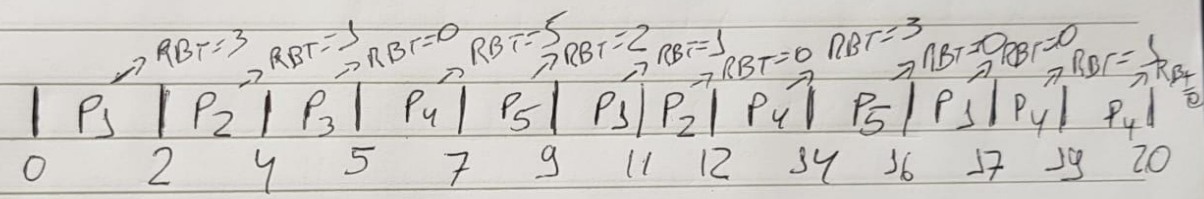
	Turnaround Time	Waiting Time
P_1	13	8
P_2	4	0
P_3	1	0
P_4	20	13
P_5	8	4

Non-preemptive Priority Queue: P_3, P_5, P_3, P_4, P_2 (Head)



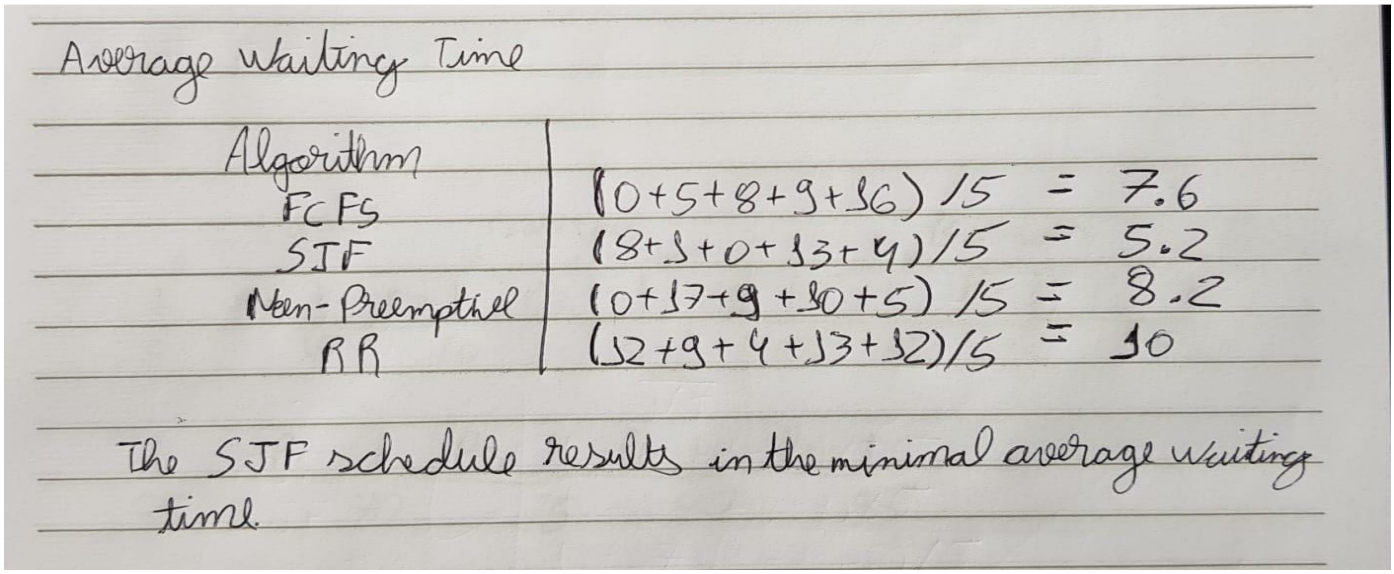
	Turnaround Time	Waiting Time
P_1	5	0
P_2	20	17
P_3	10	9
P_4	17	10
P_5	9	5

RR (Quantum = 2) Queue: P_1, P_2, P_3, P_4, P_5 (Head)



Obs: RBT is the remaining burst time.

	Turnaround Time	Waiting Time
P_1	17	12
P_2	12	9
P_3	5	4
P_4	20	13
P_5	16	12

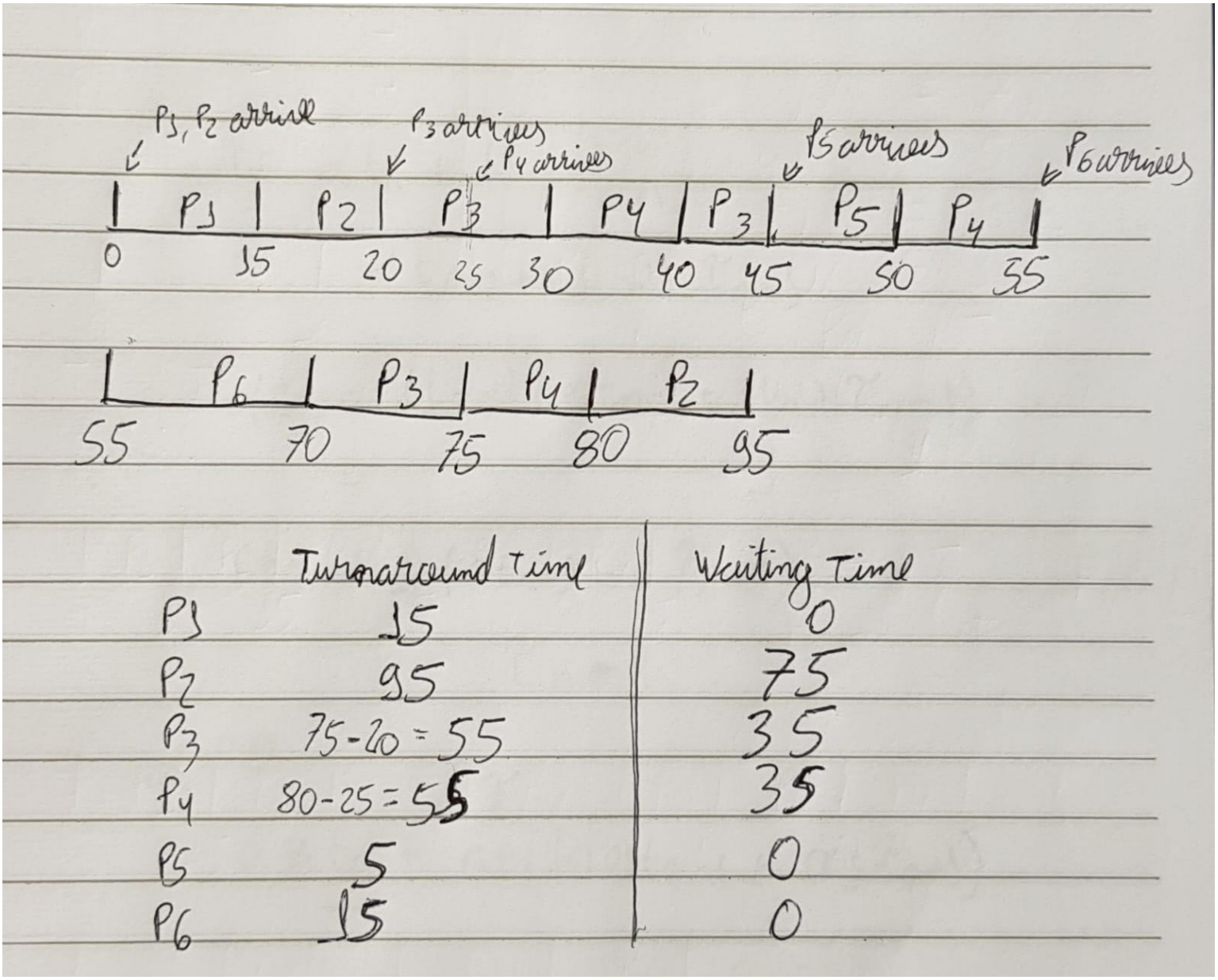


5. (10 points) 5.18 The following processes are being scheduled using a preemptive, round-robin scheduling algorithm.

Process	*Priority*	*Burst*	*Arrival*
P1	8	15	0
P2	3	20	0
P3	4	20	20
P4	4	20	25
P5	5	5	45
P6	5	15	55

Each process is assigned a numerical priority, with a higher number indicating a higher relative priority. The scheduler will execute the highest-priority process. For processes with the same priority, a round-robin scheduler will be used with a time quantum of 10 units. If a process is preempted by a higher-priority process, the preempted process is placed at the end of the queue.

a. Show the scheduling order of the processes using a Gantt chart.
b. What is the turnaround time for each process?
c. What is the waiting time for each process?



	Turnaround time	Waiting Time
P1	15	0
P2	95	75
P3	75-20=55	35
P4	80-25=55	35
P5	5	0
P6	15	0

6. (5 points) 5.20 Which of the following scheduling algorithms could result in starvation?

a. First-come, first-served

b. Shortest job first

c. Round robin

d. Priority

Shortest Job First and Priority scheduling algorithms could result in starvation.

7. **(5 points) 5.25** Explain how the following scheduling algorithms discriminate either in favor of or against short processes: a. FCFS b. RR c. Multilevel feedback queues.

   FCFS scheduling algorithms will run processes based on their arrival time order, which means that processes with a longer burst time may process before short-time processes, so short-time processes may need to wait long-burst-time processes to finish first. RR algorithms preempt processes based on a defined time quantum, which basically means that the algorithm treats processes as if they were all equal and, by that, it basically gives an opportunity to short-time processes to finish first. Multilevel feedback queues will give chances to short-time processes to finish first; if the process doesn't finish before the time quantum defined, it moves down on one priority queue, so a short-time process is likely to finish firstly than a longer burst-time process.