PART E

1. Consider the following processes with arrival times and burst times:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

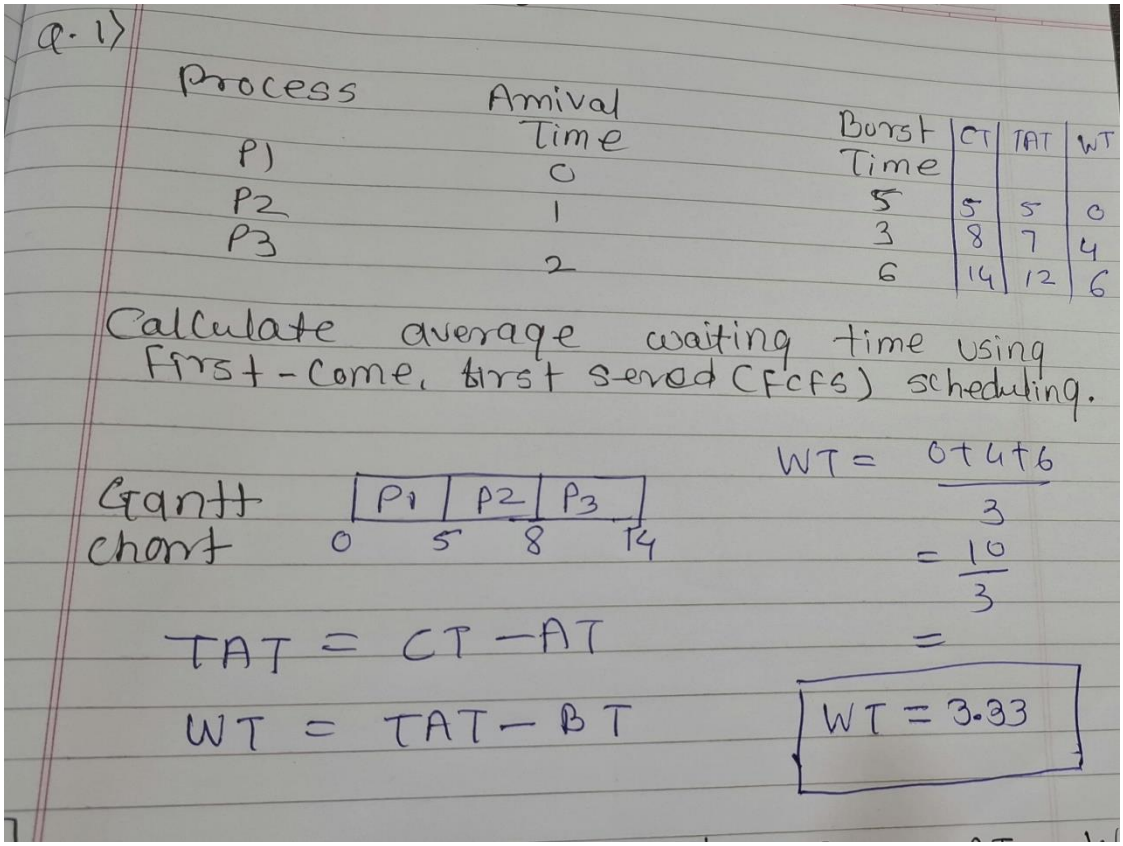
| P1 | 0 | 5 |

| P2 | 1 | 3 |

| P3 | 2 | 6 |

Calculate the average waiting time using First-Come, First-Served (FCFS) scheduling.

ANSWER:-



2. Consider the following processes with arrival times and burst times:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

| P1 | 0 | 3 |

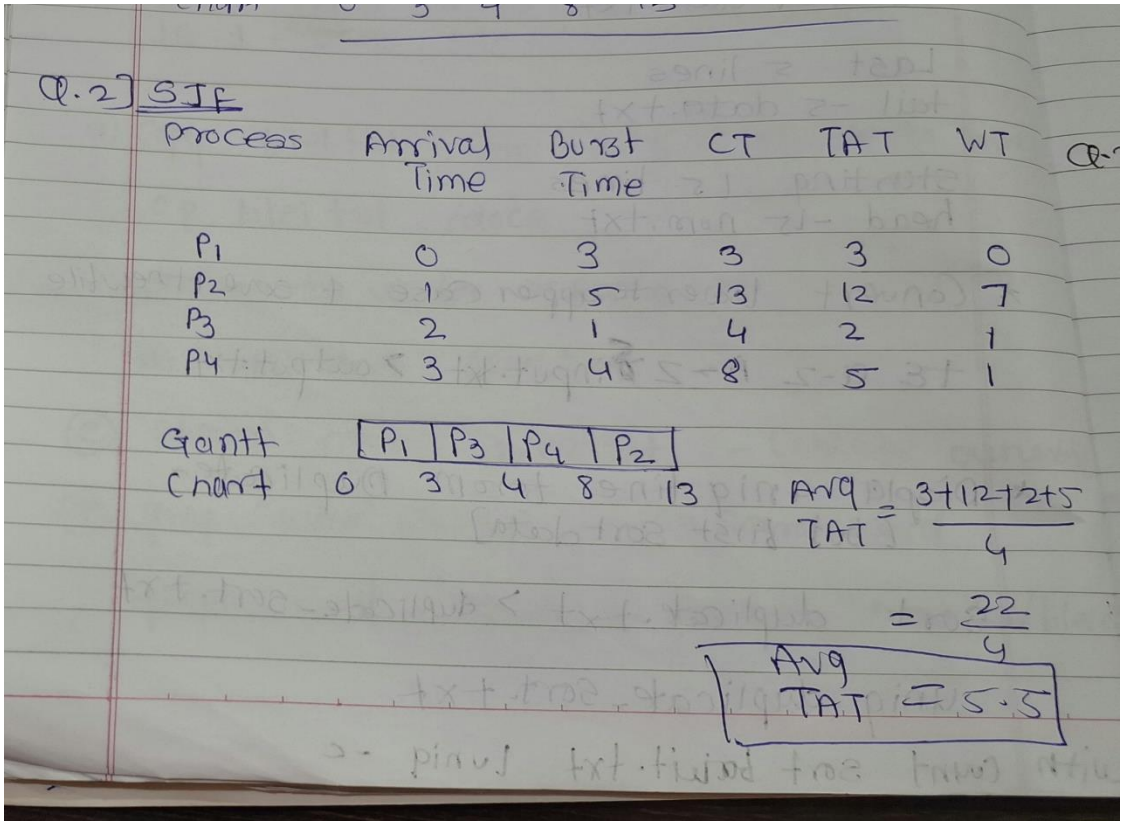
| P2 | 1 | 5 |

| P3 | 2 | 1 |

| P4 | 3 | 4 |

Calculate the average turnaround time using Shortest Job First (SJF) scheduling.

ANSWER:--



4. Consider the following processes with arrival times and burst times, and the time quantum for Round Robin scheduling is 2 units:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

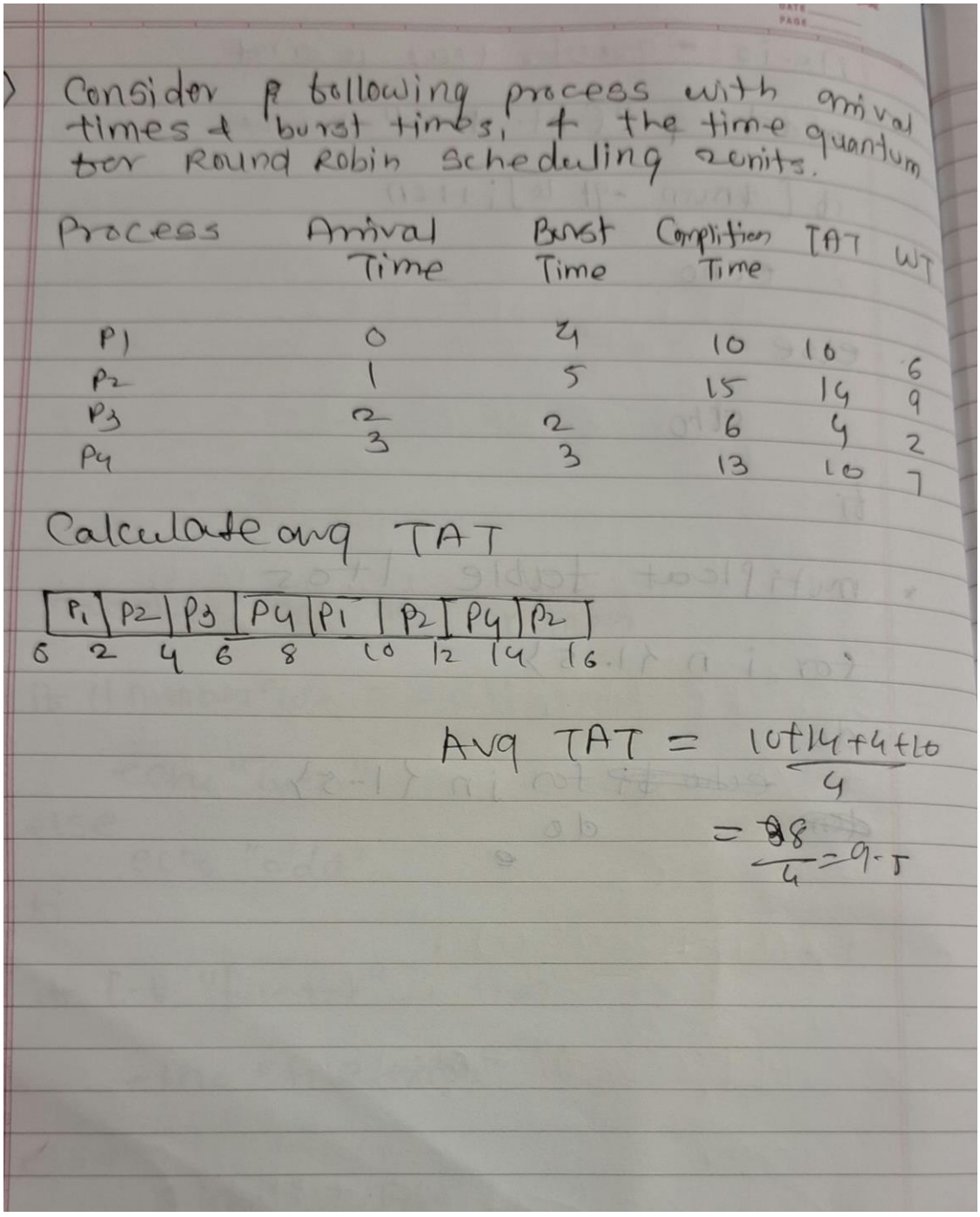
| P1 | 0 | 4 |

| P2 | 1 | 5 |

| P3 | 2 | 2 |

| P4 | 3 | 3 |

Calculate the average turnaround time using Round Robin scheduling.



Consider the following process with arrival times & burst times, & the time quantum for Round Robin scheduling 2 units.

Process	Arrival Time	Burst Time	Completion Time	TAT	WT
P1	0	4	10	10	6
P2	1	5	15	14	9
P3	2	2	6	4	2
P4	3	3	13	10	7

Calculate avg TAT

P1	P2	P3	P4	P1	P2	P4	P2
0	2	4	6	8	10	12	14	16

Avg TAT = $\frac{10+14+4+10}{4}$

$= \frac{38}{4} = 9.5$

5. Consider a program that uses the fork() system call to create a child process. Initially, the parent

process has a variable x with a value of 5. After forking, both the parent and child processes

increment the value of x by 1.

What will be the final values of x in the parent and child processes after the fork() call?

ANSWER:--

When a program uses the fork() system call to create a child process, the parent and child processes each have their own separate memory space. Here's how the variable x behaves:

1. **Initial State**: The parent process has a variable x with a value of 5.
2. **Forking**: When fork() is called, the child process gets a copy of the parent process's memory space, including the value of x.
3. **Post-Fork Execution**:
    - Both the parent and child processes will increment their own copy of x by 1.
    - After incrementing, the parent process will have x = 6, and the child process will also have x = 6.

Detailed Breakdown:

- **Parent Process**:
    - Before fork(): x = 5
    - After fork(), it increments x: x = 5 + 1 = 6
- **Child Process**:
    - Receives x = 5 from the parent process.
    - After fork(), it increments its own x: x = 5 + 1 = 6

Final Values:

- **Parent Process**: x = 6
- **Child Process**: x = 6

In summary, after the fork() call, both the parent and child processes will have their own separate copies of x, and both will have the final value of 6.