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:-

Process P) P2 P3 Calculate First-com	Amival Time C 1 2 average waiting 1e, thist seved CF	Borst CT TAT WT Time 5 5 5 0 3 8 7 4 6 14 12 6 1 time Using CFS) scheduling.
TAT =	P1 P2 P3 5 8 14 CT - AT TAT - BT	NT = 0 + 4 + 6 $= 10$ $= 3$ $= 10$ $= 3.33$

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:--

	5 9 8 13
Q.2	SJE Extended to Start St
	Process Amival Burst CT TAT WT Or
	P1 0 3 3 3 0
ofile	P2 1 5 13 12 7 P3 2 1 4 2 1
	P4 1 9 3 1 1 9 9 5 3 1
	Gantt [P1 P3 P4 P2] Chart 6 3 4 8 13 Avg 3+12+2+5 [AT 4
	1xt. ma_spaniqub < txt. +soilab = 22 :
	AV9 TATO = 5.5
	with count sort bainst txt luniq -c

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

Process	Arrival Tin	ne Burst	Time
-	-		

| P1 | 0 | 4 |

| P2 | 1 | 5 |

| P3 | 2 | 2 |

| P4 | 3 | 3 |

Calculate the average turnaround time using Round Robin scheduling.

				97	AGE .	
>	Consider R times & b	bollowing	process	s with	mi,	
	times 4 'b	ourst timbs	1 + +	he time	quan-	a
	bor Round					
	Process	Amival	Burst	Constition	TO	
	1,000	Time	Time	Time	(1)7	WT
1			(11.16	11.72		1
1	P)	0	4	10	16	
	PZ	1	5	15	14	69
1	P3	3	2	016	4	2
1	Py	3	3	13	10	7
1	0 1 1-1-					1
1	Calculate	mg TA-				
		0119	1 dupl	fleat		
	P1 P2 P3 PU	1 P1 P2 P1	1/2			
(5 2 4 6 8	3 (0 12	19 16	rai	10)	
		A	19 TAT	= 10	114+1	itle
	3 6304 64	(2-1) 11	int 4		4	
	150 1111			= 8	8 0	
	E CONTRACTOR			-	-29.	T

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:
 - \circ Before fork(): x = 5
 - After fork(), it increments x: x = 5 + 1 = 6
- Child Process:
 - \circ 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.