National Textile University

Department of Computer Science

Subject:

Operating System

Submitted to:

Sir Nasir

Submitted by:

Ahmad Fawad

Reg number:

1129

Assignment No:

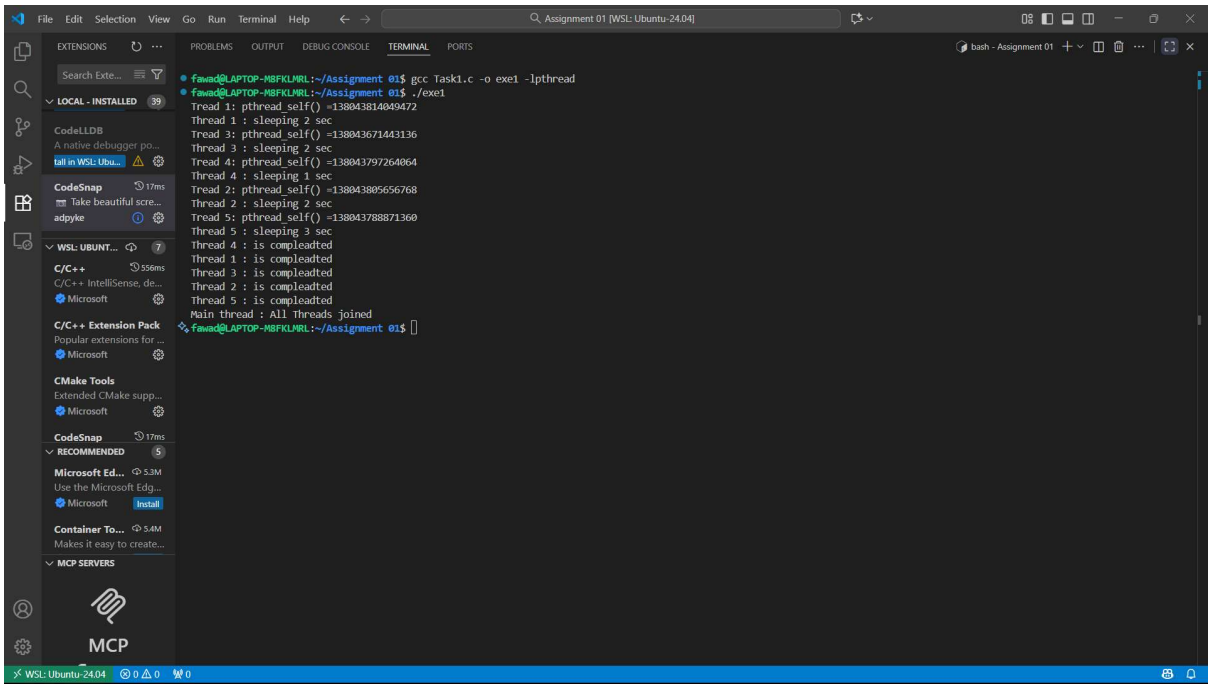
01

Semester:

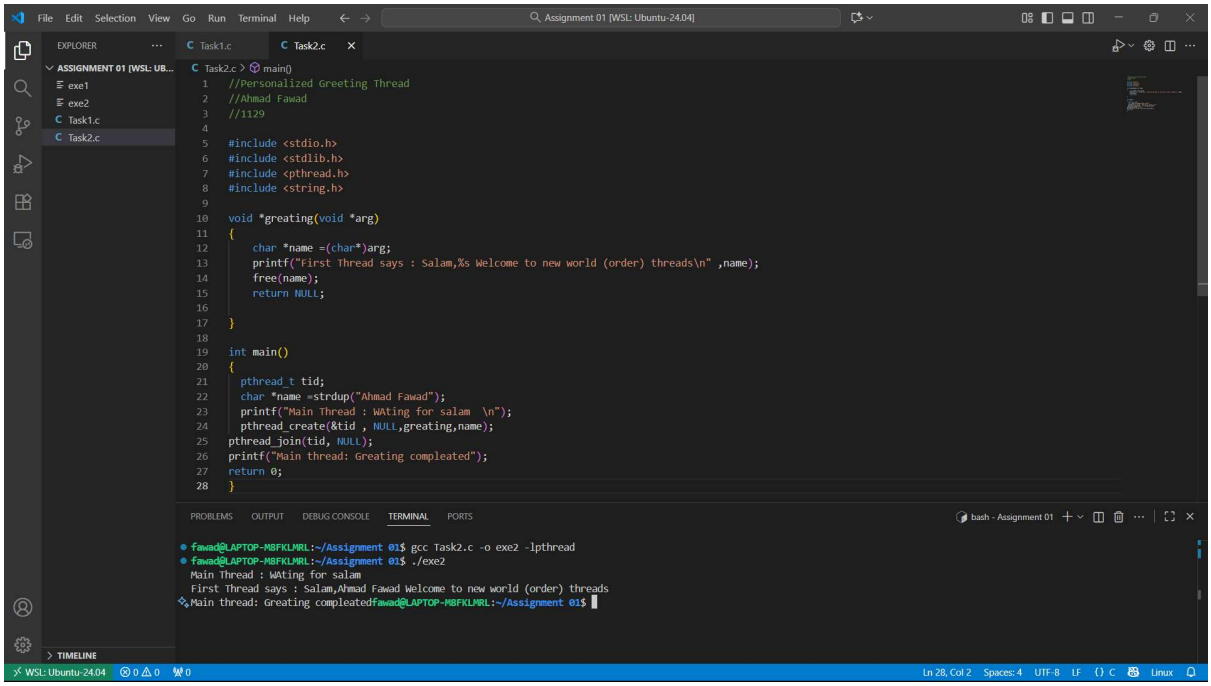
Task 1:

```c
//Thread Information Display
//Name: Ahmad Fawad
// RegNo: 1129

#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <unistd.h>

void *thread_function(void *arg)
{
    int *num = (int*)arg;
    int id = *num;
    pthread_t tid = pthread_self();

    printf("Tread %d: pthread_self() =%lu\n",id ,(unsigned long)tid);

    int sleep_time = (rand() %3)+1;
    printf("Thread %d : sleeping %d sec\n",id ,sleep_time);
    sleep(sleep_time);
    printf("Thread %d : is compleadted\n",id);
    free(arg);
    return NULL;

}

int main()
{
    pthread_t threads[5];
    for (int i =0; i<5;i++)
    {
        int  *arg = malloc(sizeof(int));
        *arg = i +1 ;
        pthread_create(&threads[i] , NULL,thread_function,arg);

    }
    for (int i = 0; i < 5; i++)
    {
        pthread_join(threads[i],NULL);
    }
    printf("Main thread : All Threads joined \n");
    return 0;

}
```

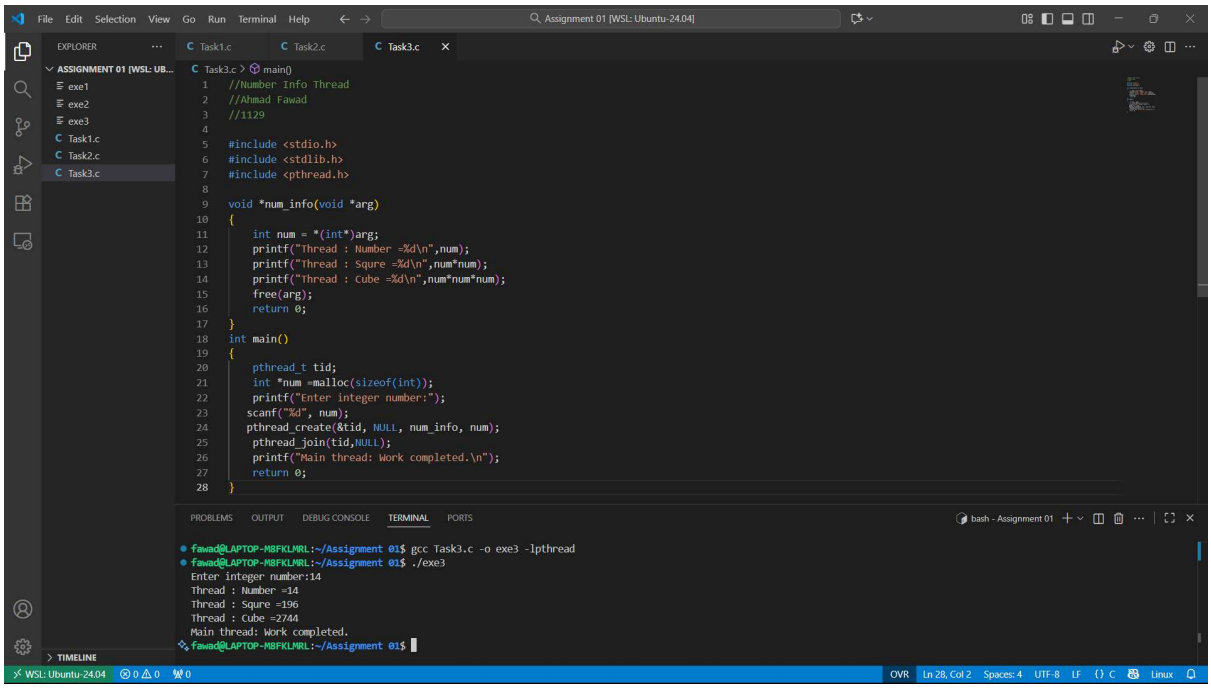
Output:



Task2:



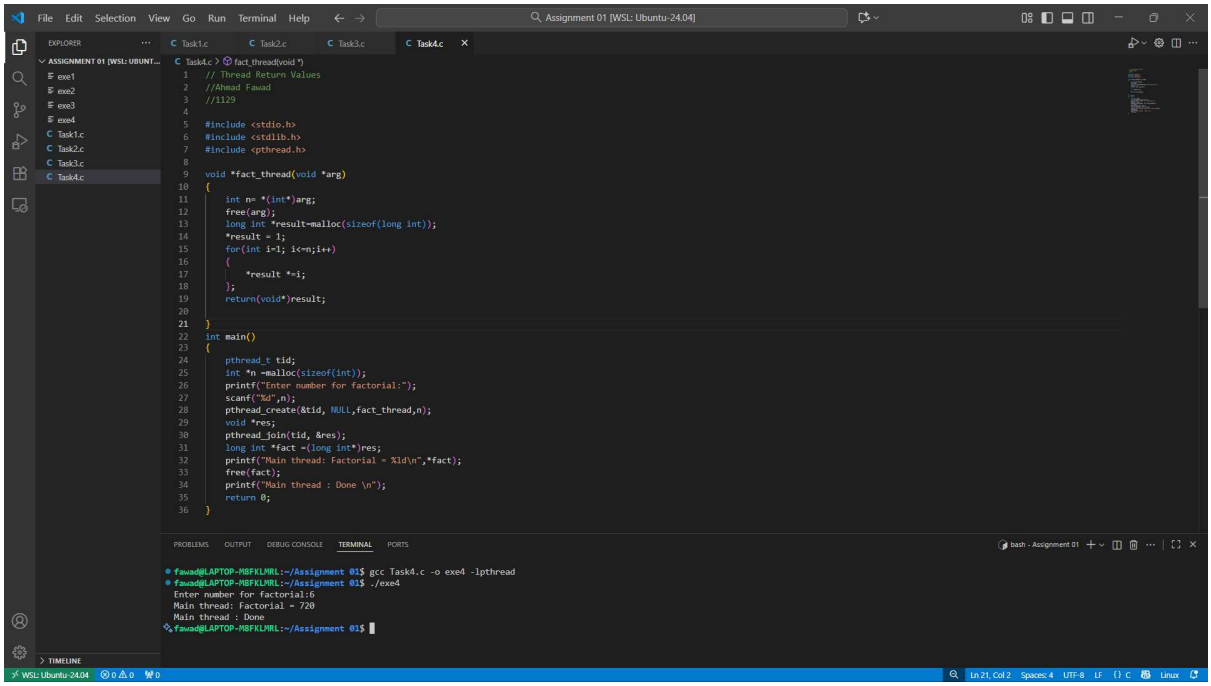
Task3:



```c
//Number Info Thread
//Ahmad Fawad
//1129

#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *num_info(void *arg)
{
    int num = *(int*)arg;
    printf("Thread : Number =%d\n",num);
    printf("Thread : Squre =%d\n",num*num);
    printf("Thread : Cube =%d\n",num*num*num);
    free(arg);
    return 0;
}
int main()
{
    pthread_t tid;
    int *num =malloc(sizeof(int));
    printf("Enter integer number:");
   scanf("%d", num);
   pthread_create(&tid, NULL, num_info, num);
    pthread_join(tid,NULL);
    printf("Main thread: Work completed.\n");
    return 0;
}
```

```
fawad@LAPTOP-M8FKLMRL:~/Assignment 01$ gcc Task3.c -o exe3 -lpthread
fawad@LAPTOP-M8FKLMRL:~/Assignment 01$ ./exe3
Enter integer number:14
Thread : Number =14
Thread : Squre =196
Thread : Cube =2744
Main thread: Work completed.
fawad@LAPTOP-M8FKLMRL:~/Assignment 01$
```

Task4:



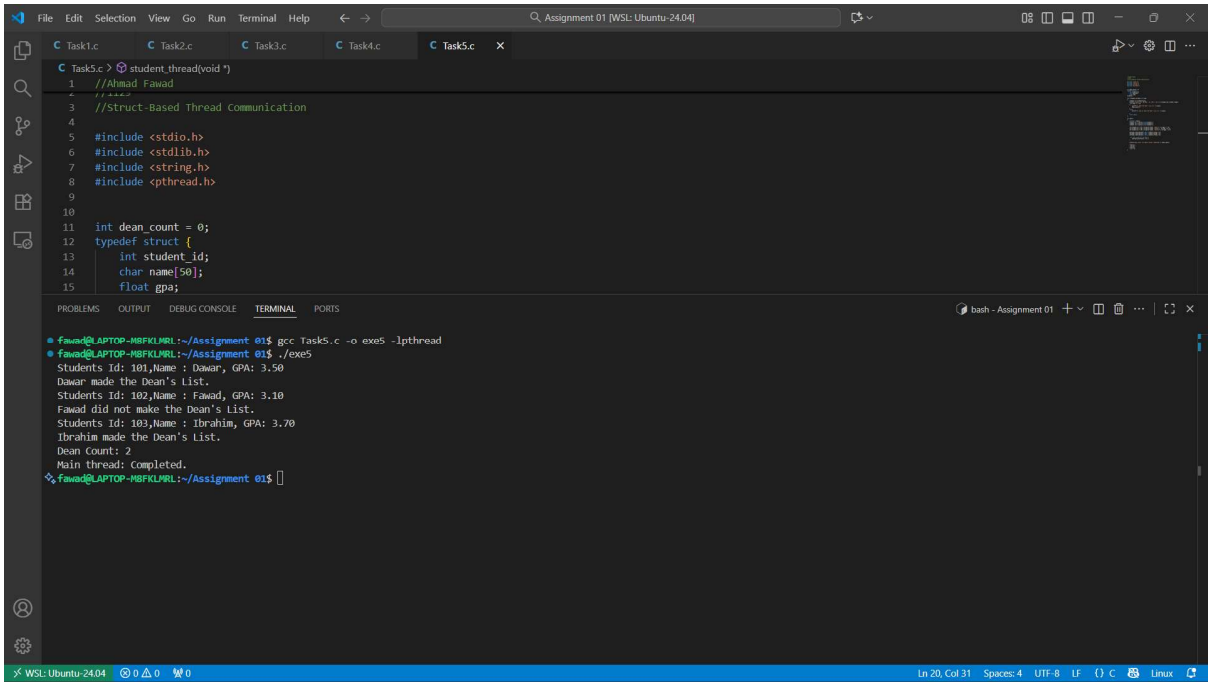
```c
// Thread Return Values
//Ahmad Fawad
//1129

#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *fact_thread(void *arg)
{
    int n= *(int*)arg;
    free(arg);
    long int *result=malloc(sizeof(long int));
    *result = 1;
    for(int i=1; i<=n;i++)
    {
        *result *=i;
    };
    return(void*)result;

}
int main()
{
    pthread_t tid;
    int *n =malloc(sizeof(int));
    printf("Enter number for factorial:");
    scanf("%d",n);
    pthread_create(&tid, NULL,fact_thread,n);
    void *res;
    pthread_join(tid, &res);
    long int *fact =(long int*)res;
    printf("Main thread: Factorial = %ld\n",*fact);
    free(fact);
    printf("Main thread : Done \n");
    return 0;
}
```

```
fawad@LAPTOP-M8FKLMRL:~/Assignment 01$ gcc Task4.c -o exe4 -lpthread
fawad@LAPTOP-M8FKLMRL:~/Assignment 01$ ./exe4
Enter number for factorial:6
Main thread: Factorial = 720
Main thread : Done
fawad@LAPTOP-M8FKLMRL:~/Assignment 01$
```

Task5:

```c
//Ahmad Fawad
//1129
//Struct-Based Thread Communication

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <pthread.h>


int dean_count = 0;
typedef struct {
    int student_id;
    char name[50];
    float gpa;
}student;

void *student_thread(void *arg)
{
    student *s =(student*)arg;
    printf("Students Id: %d,Name : %s, GPA: %.2f\n",s->student_id,s->name,s->gpa);
    if(s->gpa >= 3.5f)
    {
        printf("%s made the Dean's List.\n", s->name);
        dean_count++;
    }
    else{
        printf("%s did not make the Dean's List.\n", s->name);

    }
    return NULL;
}

int main()
{
    pthread_t tid[3];
    student *s1 = malloc(sizeof(student));
    student *s2 = malloc(sizeof(student));
    student *s3 = malloc(sizeof(student));

    s1->student_id = 101; strcpy(s1->name, "Dawar"); s1->gpa = 3.5f;
    s2->student_id = 102; strcpy(s2->name, "Fawad");  s2->gpa = 3.1f;
    s3->student_id = 103; strcpy(s3->name, "Ibrahim"); s3->gpa = 3.7f;

    pthread_create(&tid[0], NULL, student_thread, s1);
    pthread_create(&tid[1], NULL, student_thread, s2);
    pthread_create(&tid[2], NULL, student_thread, s3);

     for (int i = 0; i < 3; i++) {
        pthread_join(tid[i], NULL);
    }

    printf("Dean Count: %d \nMain thread: Completed.\n",dean_count);

    free(s1);
    free(s2);
    free(s3);
    return 0;
}
```

Output:



Section-B: Short Questions

Q1.Define an Operating System in a single line

Operating system is a bridge between the user and hardware . it can make sure that each programe run smoothly. It can manage the computer hardware and software resources

Q2. What is the primary function of the CPU scheduler?

Primary Function of CPU Schedular is which process should run next on cpu. It can decide best process according to shedular algorithm.

Q3. List any three states of a process.

Three main state of processes are:
1:ready : its mean process is ready to run but wait for cpu
2:Running: Process being runining in cpe
3:waiting:process in blocked due to i/o.

Q4. What is meant by a Process Control Block (PCB)?

A PCB is a data structure in the OS which can store the information about process like Pid,State,Register and memory detail.It help the OS to switch between process.

Q5. Differentiate between a process and a program.

A program is a set of instructions stored on disk while a process is the running That instructions of that program in memory. When a program instruction starts execution it becomes a process.

Q6. What do you understand by context switching?

When CPU wants to change one process to another . it can save the crunt process state and load the state of another process Is known as context switching

Q7. Define CPU utilization and throughput.

CPU utilization mean how much time the CPU is busy to complete the tasks
while throughput time is considerd the number of task cpu completed in given time.

Q8. What is the turnaround time of a process?

It is a total time of a process that it take wating execution as well as i/o processes

Q9. How is waiting time calculated in process scheduling?

It can represent the total time spend by the process in ready queue for Cpu execution
So we can find it by subtracting the burst time from turnaround time

Q10. Define response time in CPU scheduling.

Response time is a first response from the cpu after submission the task.

Q11. What is preemptive scheduling?

Preemptive scheduling allows the CPU to be taken from a running process and given to another process with higher priority or shorter job first time

Q12. What is non-preemptive scheduling?

In non-preemptive scheduling once a process starts using the CPU it runs until it finishes or waits for I/O. No other process can interrupt it .

Q13. State any two advantages of the Round Robin scheduling algorithm.

Every process get equal share of CPU time
It is suitable for time-sharing systems and provides quick responses for small tasks.

Q14. Mention one major drawback of the Shortest Job First (SJF) algorithm.

Its big drawback is that it can prefer the shortest jobs which caused starvation for the other processes

Q15. Define CPU idle time.

A time in which CPU is free . not a single process is ready to run

Q16. State two common goals of CPU scheduling algorithms.

CPU Utilization keep the cpu busy as much as possible
minimize the waiting time improve system and user effecency

Q17. List two possible reasons for process termination.

1.Process can complete its task so terminate
2. due to an error, such as memory overflow or killed by OS

Q18. Explain the purpose of the wait() and exit() system calls.

The exit() call ends a process and returns its status to the OS. The wait() call allows the parent process to pause until its child process finishes execution.

Q19. Differentiate between shared memory and message-passing models of IPC.

Shared Memory	Message Passing
Processes communicate by sharing a common memory space.	Processes communicate by sending and receiving messages.
Faster communication	Slower due to kernel mediation for message transfer.
Requires synchronization .	Easier to synchronize; handled by OS.

Q20. Differentiate between a thread and a process.

Process	Thread
Independent program with its own memory space.	A smaller unit of a process sharing the same memory.
Heavyweight (fork)	Lightweight faster to create and switch (pthread).
Communication between processes is slower.	Threads communicate easily via shared memory.

Q21. Define multithreading.

Running multipule threades in a single process to perform a different tasks at the same time to improve the usage of CPU

Q22. Explain the difference between a CPU-bound process and an I/O bound process.

A CPU-bound process mainly uses the processor for computation like calculations, while an I/O-bound process spends most time waiting for input/output operations to complete.

Q23. What are the main responsibilities of the dispatcher?

The main task of dispatecher is give the cpu control to that process which slected by scheduler by performing context switching

Q24. Define starvation and aging in process scheduling.

Starvation is when the process never get the CPU for process dua to low priority .
againg is technique to hight the priority to dicard the starvation

Q25. What is a time quantum (or time slice)?

It is a fixed time that gives to the each process before switching another process . used in round robin

Q26. What happens when the time quantum is too large or too small?

If too large it behaves like FCFS and increases waiting time. If too small too many context switches occur which can wasting CPU time.

Q27. Define the turnaround ratio (TR/TS).

It is a ratio between total time process served
It shows how much total time a process took compared to its actual CPU service time.

Q28. What is the purpose of a ready queue?

The ready queue hold all the processes which are ready and wating for the CPU to be free .
shedular can slelect next process in ready queue

Q29. Differentiate between a CPU burst and an I/O burst.

CPU Burst the period when a process actively uses the CPU for computation.
I/O Burst is the period when a process waits for or performs input/output operations.

Q30. Which scheduling algorithm is starvation-free, and why?

Round Robin is starvation free because every process gets an equal time share in a cyclic order so none are ignored.

Q31. Outline the main steps involved in process creation in UNIX.

Main steps in process creation in UNIX:

1. fork() Creates a new child process.
2. exec() Replaces the child's memory with a new program.
3. wait() Parent waits for the child to finish.
4. exit() –Child terminates and releases resources.

Q32. Define zombie and orphan processes.

A zombie process is which have completed its task and remain in execution list . and the orphan process is whos parents process complete his task before it without wating

Q33. Differentiate between Priority Scheduling and Shortest Job First (SJF).

Priority scheduling selects the process with the highest priority, while SJF selects the one with the shortest CPU burst time. Both aim to improve efficiency.

Q34. Define context switch time and explain why it is considered overhead.

Context switch time is the time spent saving and loading process information during switching. It's considered overhead because no actual processing work happens then.

Q35. List and briefly describe the three levels of schedulers in an OS.

- LongTerm Scheduler Selects which processes are admitted into the ready queue.
- ShortTerm Scheduler Chooses which ready process runs next on the CPU.
- MediumTerm Scheduler Temporarily suspends or resumes processes to control multitasking and memory use

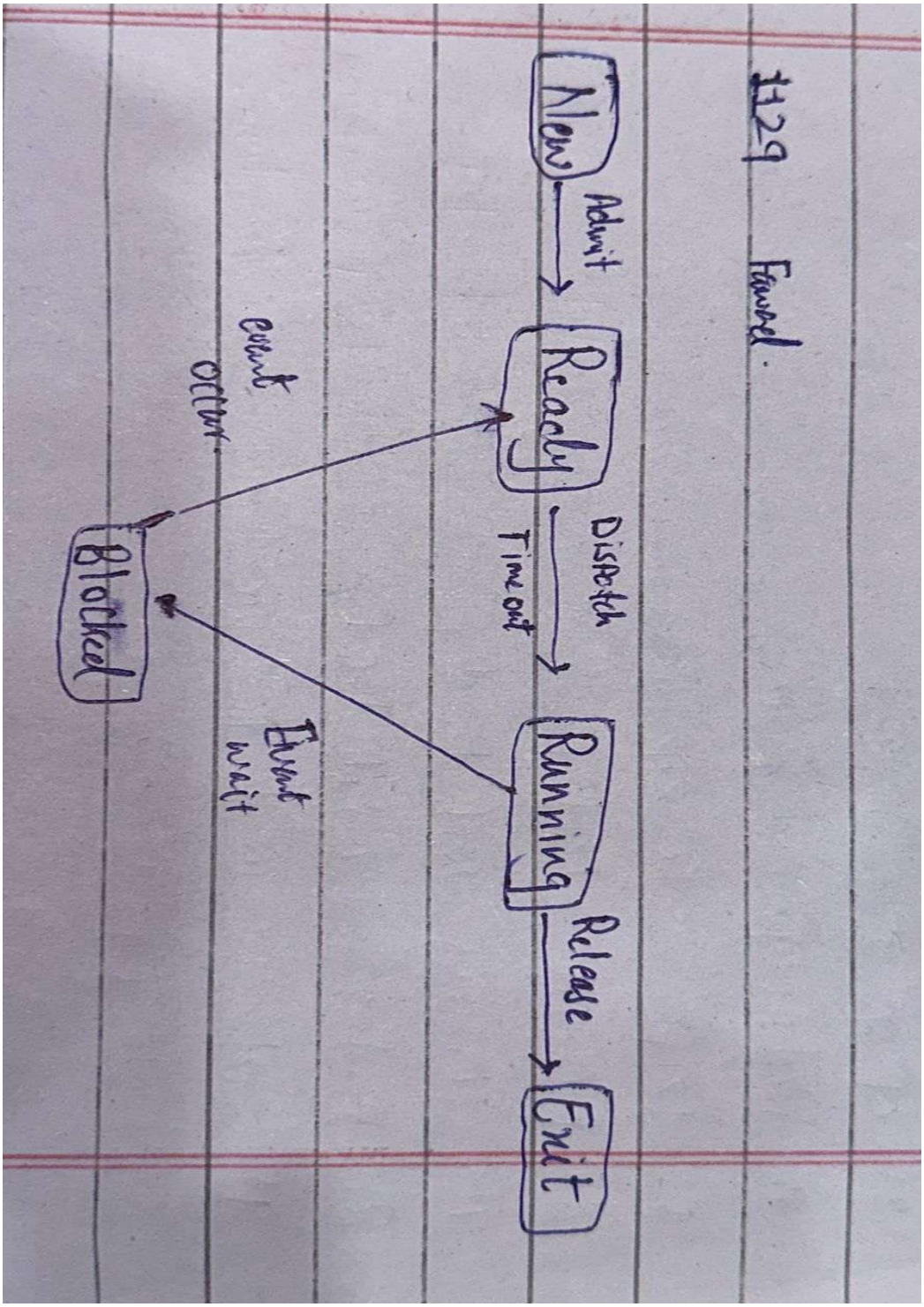
Q36. Differentiate between User Mode and Kernel Mode in an OS.

User mode is where normal user programs run with limited access. Kernel mode is a privileged mode used by the OS to control hardware and perform critical operations.

Section-C: Technical / Analytical Questions

1. Describe the complete life cycle of a process with a neat diagram showing transitions between New, Ready, Running, Waiting, and Terminated states.

- New: Process is being created.
- Ready: Process is ready to run and waiting in ready queue.
- Running: CPU is executing the process.
- Waiting/Blocked: Process waits for I/O or event.
- Terminated: Process finished or killed.



2. Write a short note on context switch overhead and describe what information must be saved and restored.

Context switch overhead is the extra time the CPU spends saving the state of the current process and loading the state of the next process , during which no productive work of those processes is done. Minimizing unnecessary switches improves performance.

What must be saved / restore

- Context of Processor.

- Process state.
- Memory management.
- Scheduling.

3. List and explain the components of a Process Control Block (PCB).

1. Process stste: running,waiting
2. Program Id: unique identifier of each process
3. Program counter: Adress of next instruction to execute
4. Cpu Register: store process data during execution
5. Memory management: Details like base and limit register
6. Accounting Info: CPU usage,process priority
7. I/O Status: List i/o devices used by process

4.Long-Term, Medium-Term, Short-Term Schedulers

Long-Term Scheduler	Short-Term Scheduler	Medium-Term Scheduler
Selects processes from secondary storage (job pool) and loads them into memory.	Selects one ready process for CPU execution.	Temporarily removes and later reintroduces processes to control load.
Slowest of all schedulers.	Fastest; runs most frequently.	Intermediate speed.
Rare or absent in time-sharing systems.	Always present; key in time-sharing systems.	Used mainly in time-sharing systems for swapping.
Admits new jobs into memory.	Dispatches ready processes to CPU.	Suspends and later resumes processes.

5. Explain CPU Scheduling Criteria (Utilization, Throughput, Turnaround, Waiting, and Response) and their optimization goals.

1. CPU Utilization: keep CPU busy as much as possible. Goal: high %tage
2. Throughput: processes completed per time unit. Goal: maximize
3. Turnaround Time: finish time arrival time. Goal: minimize average
4. Waiting Time: total time in ready queue. Goal: minimize average
5. Response Time: time from submission to first response. Goal: minimize, important for interactive systems

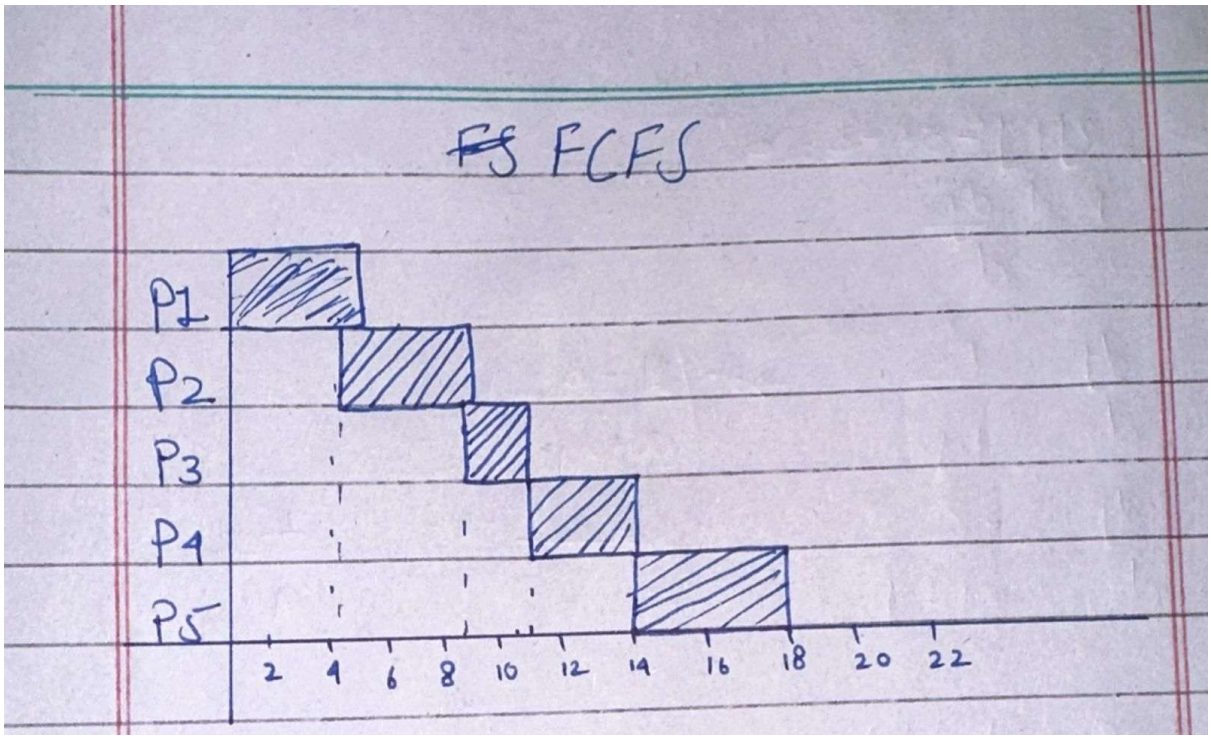
Section-D: CPU Scheduling Calculations

• Perform the following calculations for each part (A–C).
• a) Draw Gantt charts for FCFS, RR (Q=4), SJF, and SRTF.
• b) Compute Waiting Time, Turnaround Time, TR/TS ratio, and CPU Idle Time.
• c) Compare average values and identify which algorithm performs best.

Part-A :

Process	Arrival Time	Service Time
P1	0	4
P2	2	5
P3	4	2
P4	6	3
P5	9	4

Gant chart:

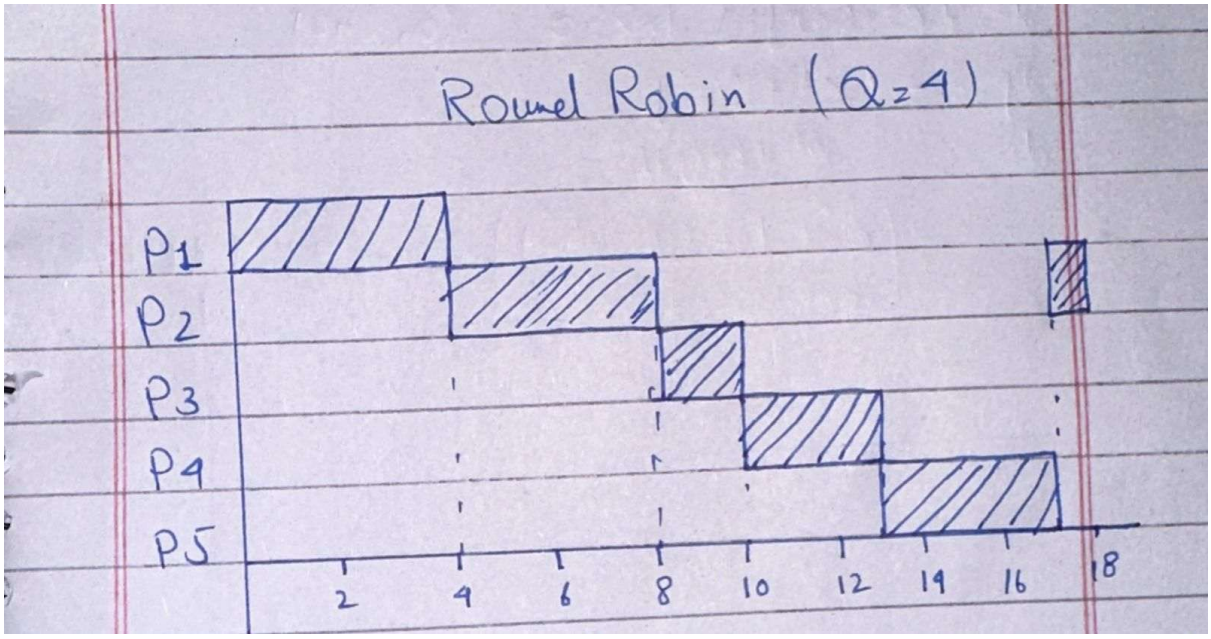


Completion, Turn Around, Waiting:
P1: Finish=4, Turn Around=4, Waiting=0
P2: Finish=9, Turn Around=7, Waiting=2
P3: Finish=11, Turn Around=7, Waiting=5
P4: Finish=14, Turn Around=8, Waiting=5
P5: Finish=18, Turn Around=9, Waiting=5

Averages:
Avg Wait Time= (0+2+5+5+5)/5 = 3.4
Avg Turn Around= (4+7+7+8+9)/5 = 7.0
Avg TR/TS = 2.16
CPU Idle = 0

Round Robin:

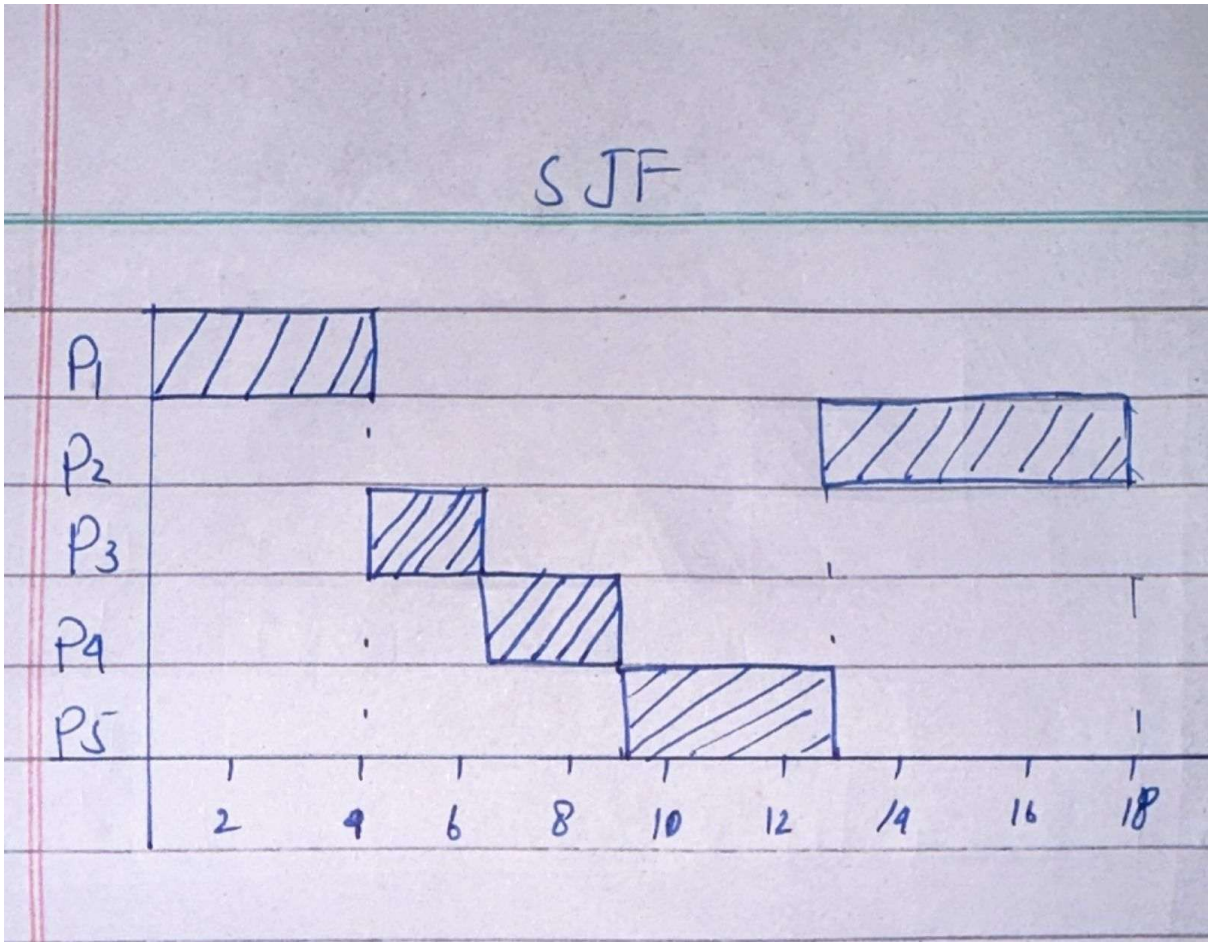
Gant Chart:



Completion, Turn Around, Waiting:
P1: Finish=4, Turn Around=4, Waiting=0
P2: Finish=18, Turn Around=16, Waiting=11
P3: Finish=10, Turn Around=6, Waiting=4
P4: Finish=13, Turn Around=7, Waiting=4
P5: Finish=17, Turn Around=8, Waiting=4

Averages:
Avg Wait Time= (0+11+4+4+4)/5 = 4.6
Avg Turn Around= (4+16+6+7+8)/5 = 8.2
Avg TR/TS = 2.31
CPU Idle = 0

Shortest Job First:



Completion, Turn Around, Waiting:
P1: Finish=4, Turn Around=4, Waiting=0
P2: Finish=18, Turn Around=16, Waiting=11
P3: Finish=6, Turn Around=2, Waiting=0
P4: Finish=9, Turn Around=3, Waiting=0
P5: Finish=13, Turn Around=4, Waiting=0

Averages:
Avg Wait Time= (0+11+0+0+0)/5 = 2.2
Avg Turn Around= (4+16+2+3+4)/5 = 5.8
Avg TR/TS = 1.44
CPU Idle = 0

Conclusion (Part-A):

SJF/SRTF gives best average turnaround and smallest average waiting time for these arrivals. RR improves response but increases average turnaround vs SJF. FCFS is

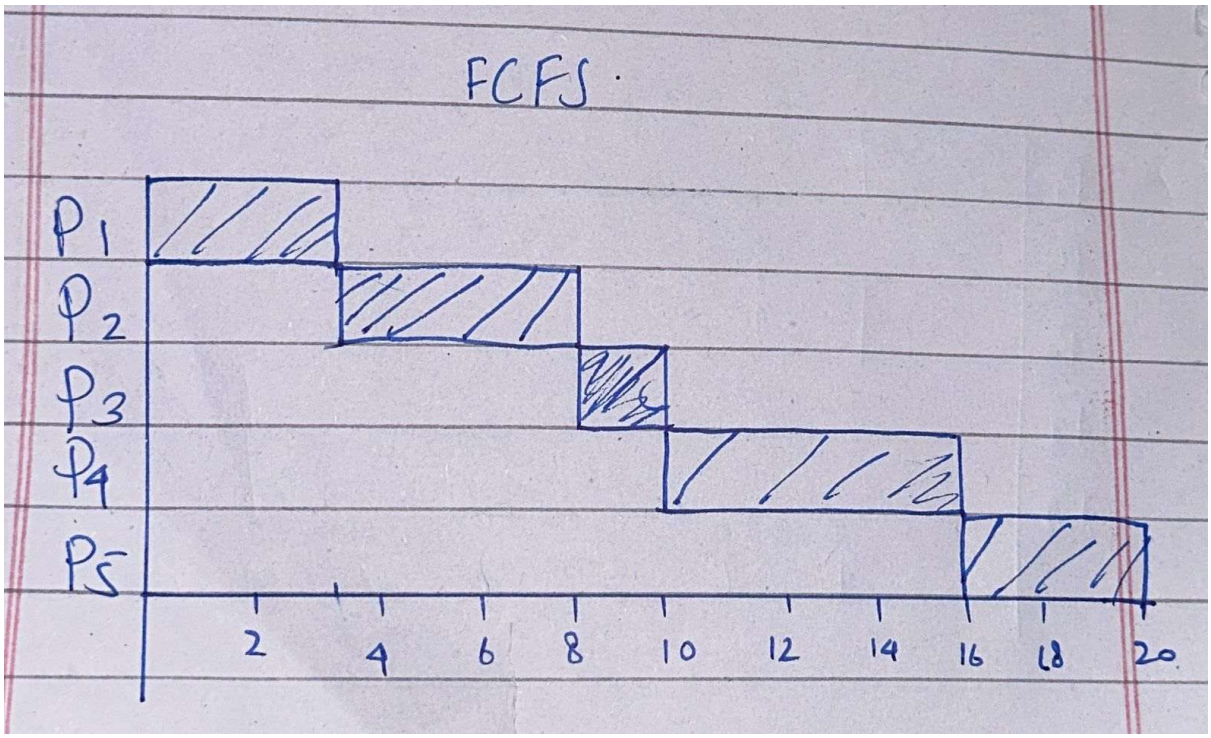
Part B:

Process	Arrival Time	Service Time (Burst Time)
P1	0	3
P2	1	5

P3	3	2
P4	9	6
P5	10	4

FCFS:

Gant chart:

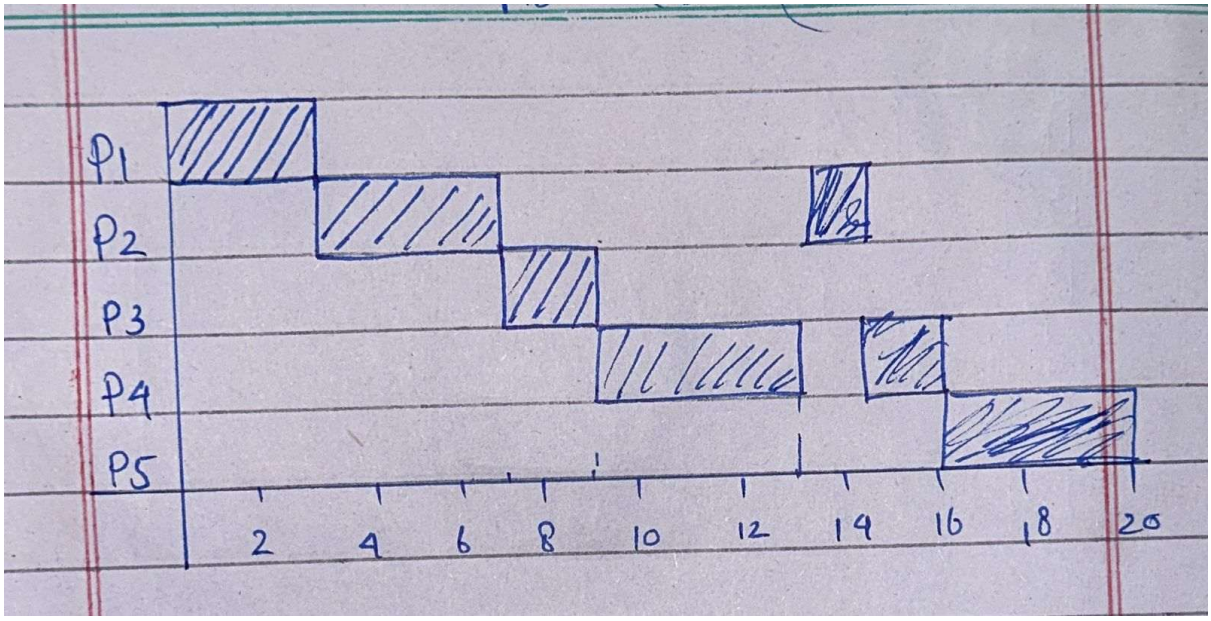


Completion, Turn Around, Waiting:
P1: Finish=3, Turn Around=3, Waiting=0
P2: Finish=8, Turn Around=7, Waiting=2
P3: Finish=10, Turn Around=7, Waiting=5
P4: Finish=16, Turn Around=7, Waiting=11
P5: Finish=20, Turn Around=10, Waiting=6

Averages:
Avg Wait Time= (0+2+5+11+6)/5 = 4.8
Avg Turn Around= (3+7+7+7+10)/5 = 6.8
Avg TR/TS = 1.91
CPU Idle = 0

Round Robin:

Gant chart:

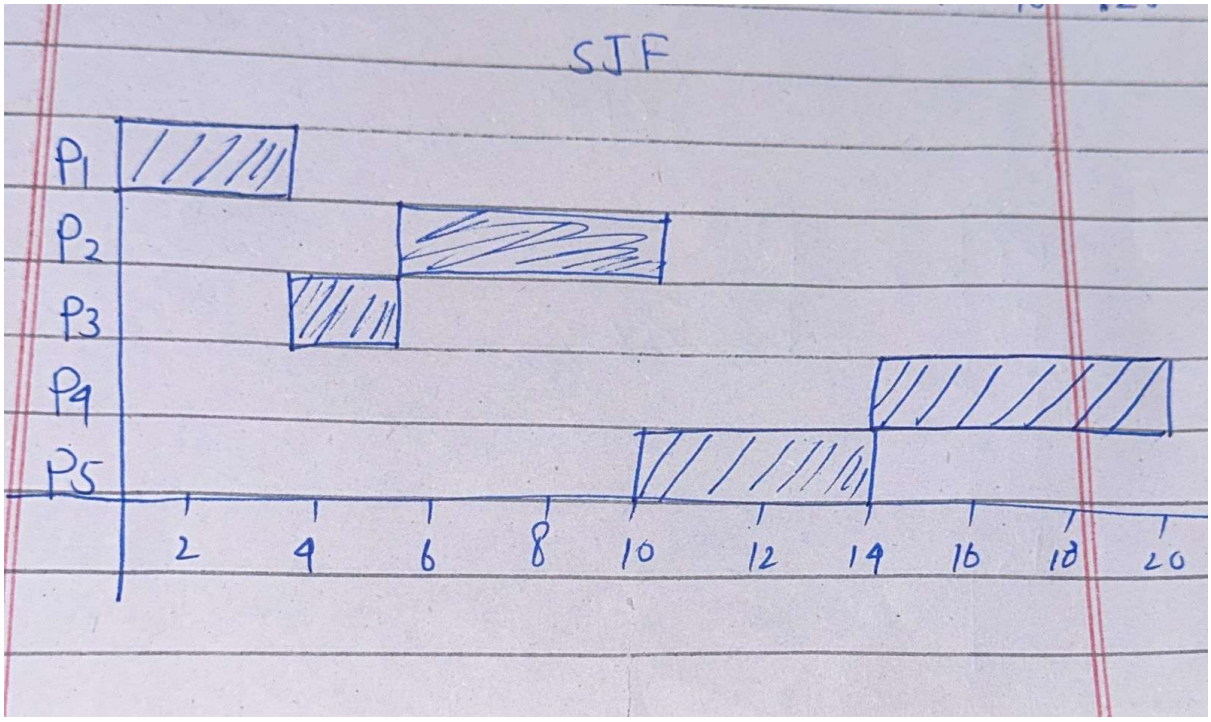


Completion, Turn Around, Waiting:
P1: Finish=3, Turn Around=3, Waiting=0
P2: Finish=14, Turn Around=13, Waiting=8
P3: Finish=9, Turn Around=6, Waiting=4
P4: Finish=16, Turn Around=7, Waiting=1
P5: Finish=20, Turn Around=10, Waiting=6

Averages:
Avg Wait Time= (0+8+4+1+6)/5 = 3.8
Avg Turn Around= (3+13+6+7+10)/5 = 7.8
Avg TR/TS = 2.05
CPU Idle = 0

Shoertest job First:

Gant chart:



Completion, Turn Around, Waiting:
P1: Finish=3, Turn Around=3, Waiting=0
P2: Finish=10, Turn Around=9, Waiting=4
P3: Finish=5, Turn Around=2, Waiting=0
P4: Finish=20, Turn Around=11, Waiting=5
P5: Finish=14, Turn Around=4, Waiting=6

Averages:
Avg Wait Time= (0+4+0+5+0)/5 = 1.8
Avg Turn Around= (3+9+2+11+4)/5 = 5.8
Avg TR/TS = 1.33
CPU Idle = 0

Conclusion:

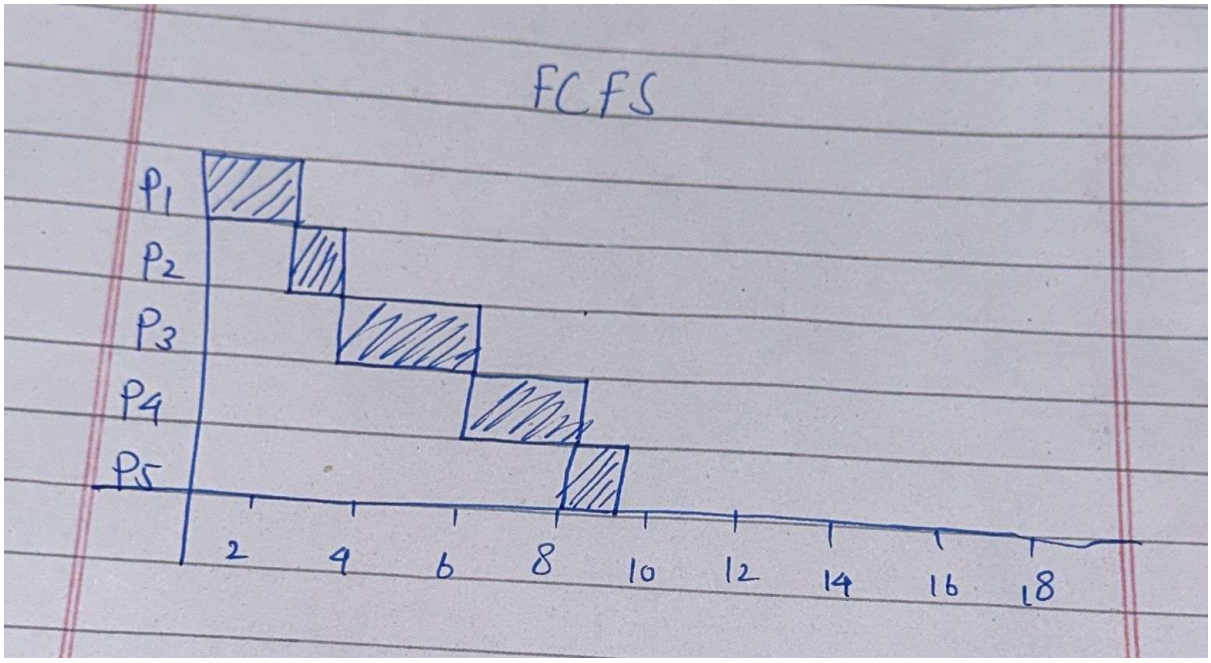
SJF gives best average waiting & turnaround; RR gives fairness but higher average turnaround than SJF.

Part C

Process	Arrival Time	Service Time (Burst)
P1	0	2
P2	1	1
P3	2	3
P4	3	2
P5	4	1

FCFS:

Gant chart:

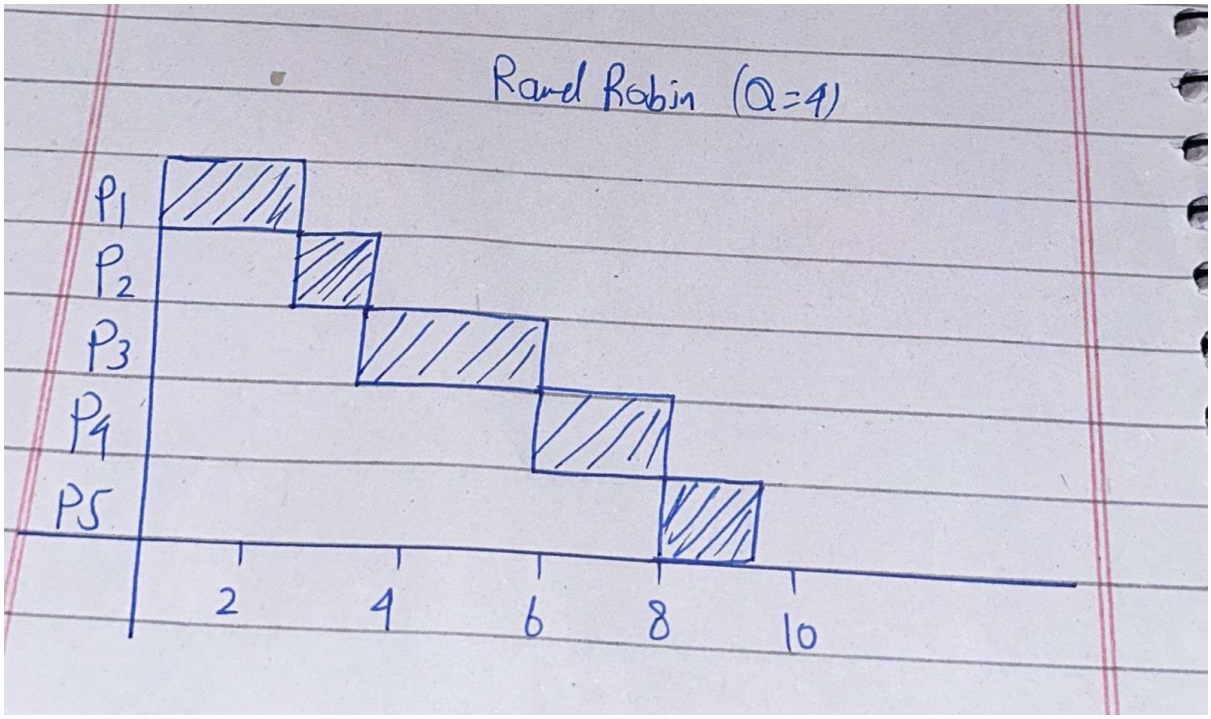


Completion, Turn Around, Waiting:
P1: Finish=2, Turn Around=2, Waiting=0
P2: Finish=3, Turn Around=3, Waiting=1
P3: Finish=6, Turn Around=4, Waiting=1
P4: Finish=8, Turn Around=5, Waiting=3
P5: Finish=9, Turn Around=5, Waiting=4

Averages:
Avg Wait Time= (0+1+1+3+4)/5 = 1.8
Avg Turn Around= (2+4+4+5+5)/5 = 3.6
Avg TR/TS = 2.37
CPU Idle = 0

Round Robin :

Gant chart:

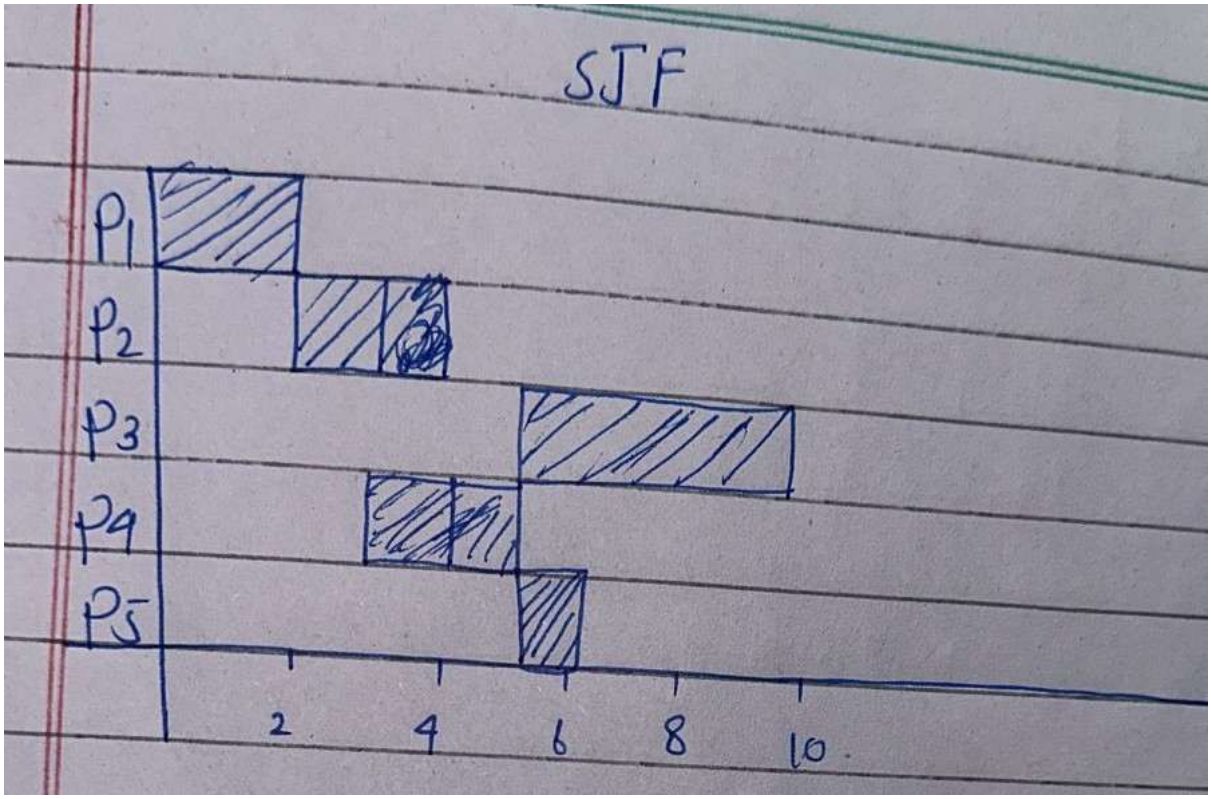


Completion, Turn Around, Waiting:
P1: Finish=2, Turn Around=2, Waiting=0
P2: Finish=3, Turn Around=3, Waiting=1
P3: Finish=6, Turn Around=4, Waiting=1
P4: Finish=8, Turn Around=5, Waiting=3
P5: Finish=9, Turn Around=5, Waiting=4

Averages:
Avg Wait Time= (0+1+1+3+4)/5 = 1.8
Avg Turn Around= (2+4+4+5+5)/5 = 3.6
Avg TR/TS = 2.37
CPU Idle = 0

SJF:

Gant chart:



Completion, Turn Around, Waiting:
P1: Finish=2, Turn Around=2, Waiting=0
P2: Finish=3, Turn Around=2, Waiting=1
P3: Finish=9, Turn Around=7, Waiting=4
P4: Finish=5, Turn Around=2, Waiting=0
P5: Finish=6, Turn Around=2, Waiting=1

Averages:
Avg Wait Time= (0+1+4+0+1)/5 = 1.2
Avg Turn Around= (2+2+7+2+2)/5 = 3.0
Avg TR/TS = 2.5
CPU Idle = 0

Conclusion:

with these sample times SJF gives lower waiting times than FCFS.