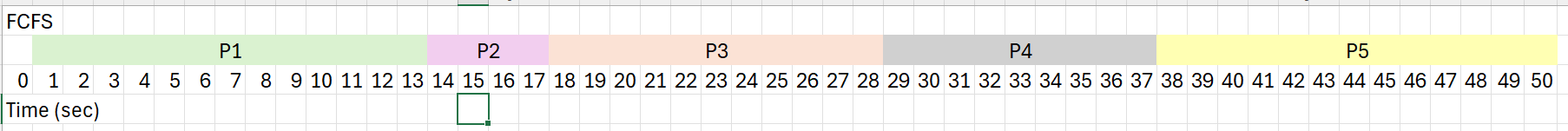
Assignment 2  
Part 1:

a) [1 mark] Consider the following set of processes. Each process has a single CPU burst and does not perform any I/O.  
Process Arrival Time (sec) Execution Time (sec)  
P1 0 13  
P2 6 4  
P3 15 11  
P4 21 9  
P5 24 13  
With the help of Gantt charts, compute the mean turnaround time for the following scheduling algorithms:  
(i) FCFS (First Come First Serve)   
(ii) Round Robin (time slice of 4 sec).  
(iii) Shortest Job First with preemption  
(iv) Multiple queues with feedback (high priority queue: quantum = 2; mid-priority queue: quantum = 3; low priority queue: FIFO  
  
A) (i) FCFS

  
**Turnaround Times:**

P1 = 13

P2 = 11

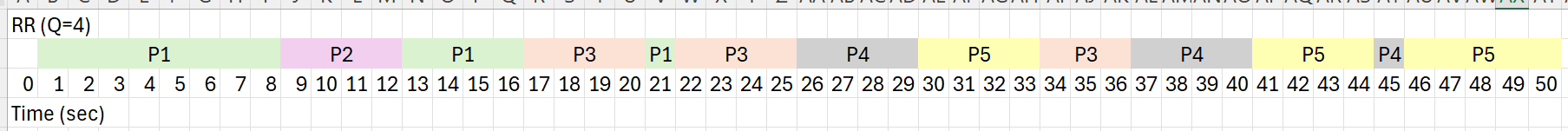
P3 = 13

P4 = 16

P5 = 26

**Mean Turnaround Time** ​=15.8 sec

(ii) Round Robin (time slice of 4 sec).

  
**Turnaround Times:**

P1 = 21

P2 = 12-6 = 6

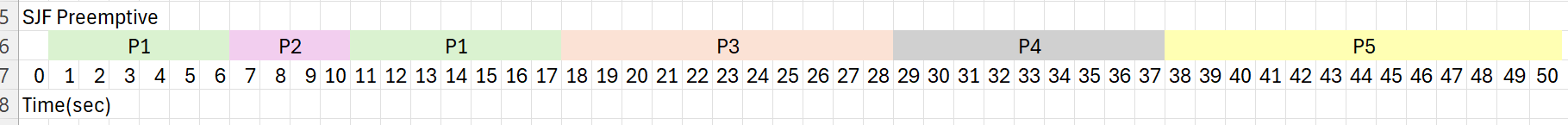
P3 = 36-15 = 21

P4 = 45-21 = 24

P5 = 50-24 = 26

**Mean Turnaround Time** ​= 19.6 sec

(iii) Shortest Job First with preemption



**Turnaround Times:**

P1 = 17

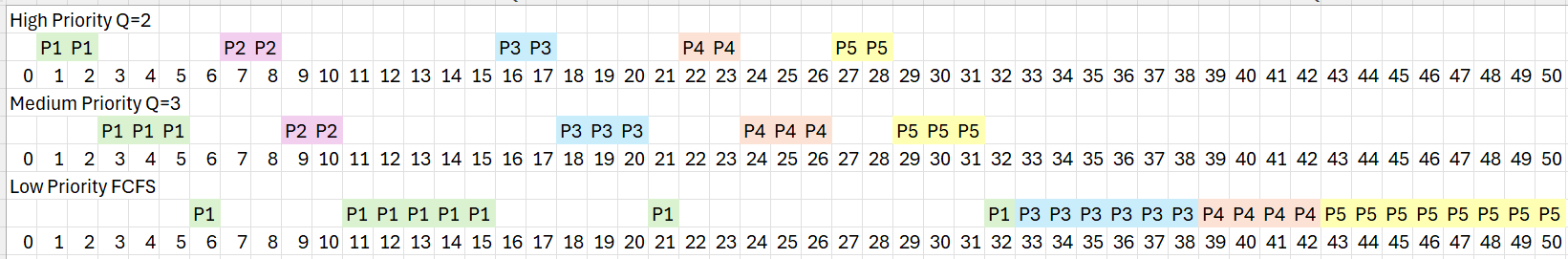
P2 = 10-6 = 4

P3 = 28-15 = 13

P4 = 37-21 = 16

P5 = 50-24 = 26

**Mean Turnaround Time** ​= 15.2 sec

(iv) Multiple queues with feedback (high priority queue: quantum = 2; mid-priority queue: quantum = 3; low priority queue: FIFO  


**Turnaround Times:**

P1 = 32

P2 = 10-6 = 4

P3 = 38-15 = 23

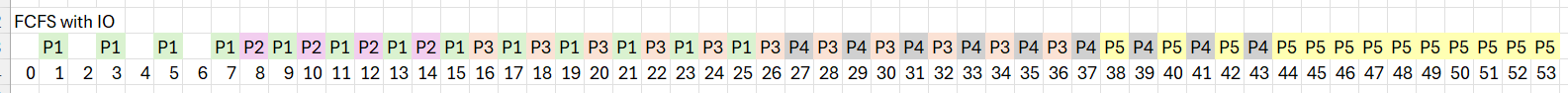
P4 = 42-21 = 21

P5 = 50-24 = 26

**Mean Turnaround Time** ​= 21.2 sec  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b) [1 mark] Now assume that each process in part a) requests to do an I/O every 1 sec., and the duration of each of these I/O is 1 sec. Create new Gantt diagrams considering the I/O operations and repeat all the parts done in part a) using this new input trace.

(i) FCFS with IO

  
**Turnaround Times:**

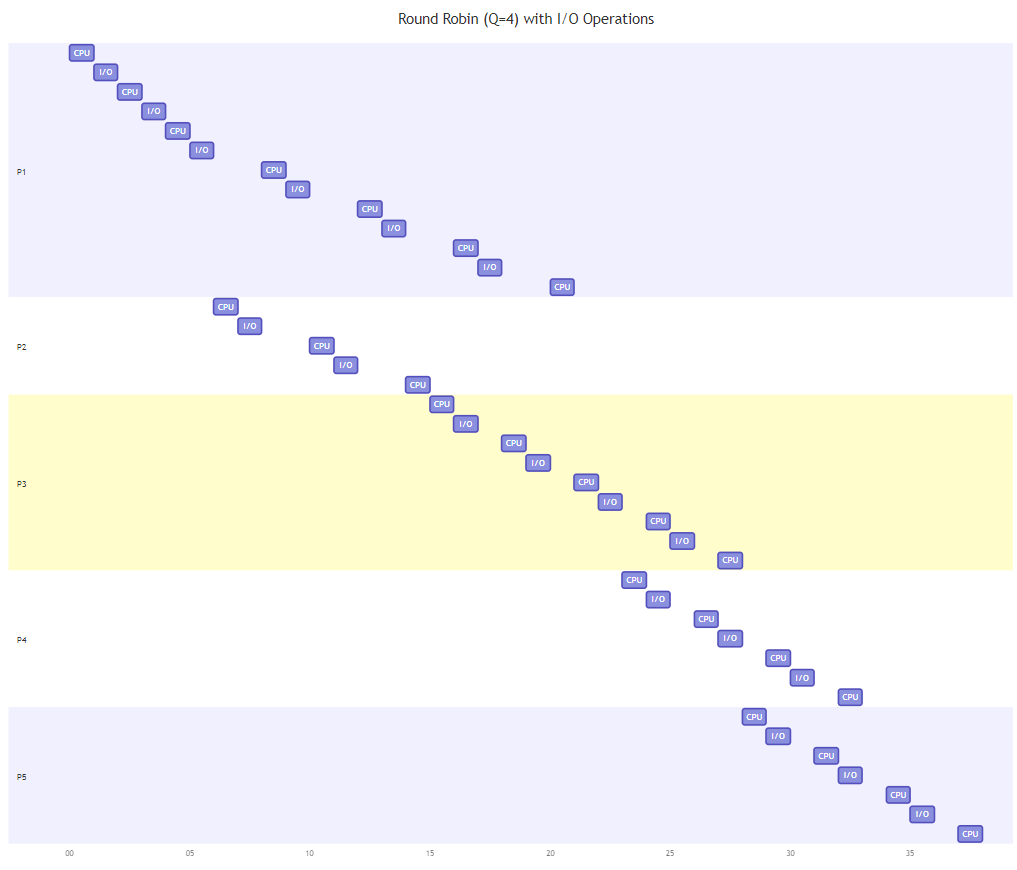
P1 = 25-0 = 25

P2 = 14-8 = 6

P3 = 36-16 = 20

P4 = 43-27 = 16

P5 = 53-38=15

**Mean Turnaround Time** ​= (25+6+20+16+15)/5 = 16.4sec  
  
(ii) Round Robin (time slice of 4 sec).  
  
**Turnaround Times:**

P1 = 21

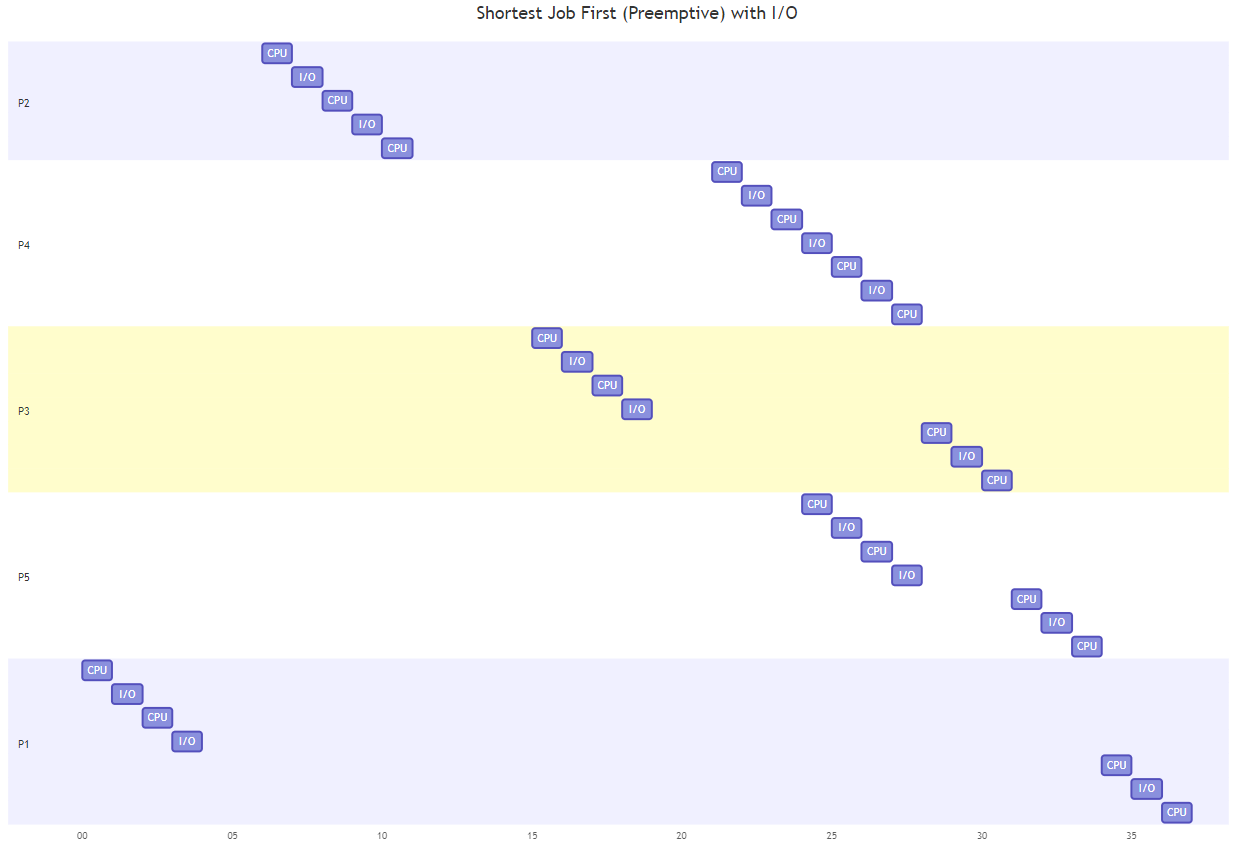
P2 = 15-6 = 9

P3 = 28 – 15 = 13

P4 = 33 – 21 = 12

P5 = 38 – 24 = 14

**Mean Turnaround Time** ​=  (21 + 9 + 13 + 12 + 14) / 5 = 13.8 sec

(iii) Shortest Job First with preemption  
  
**Turnaround Times:**

P1 = 37

P2 = 11 – 6 =5

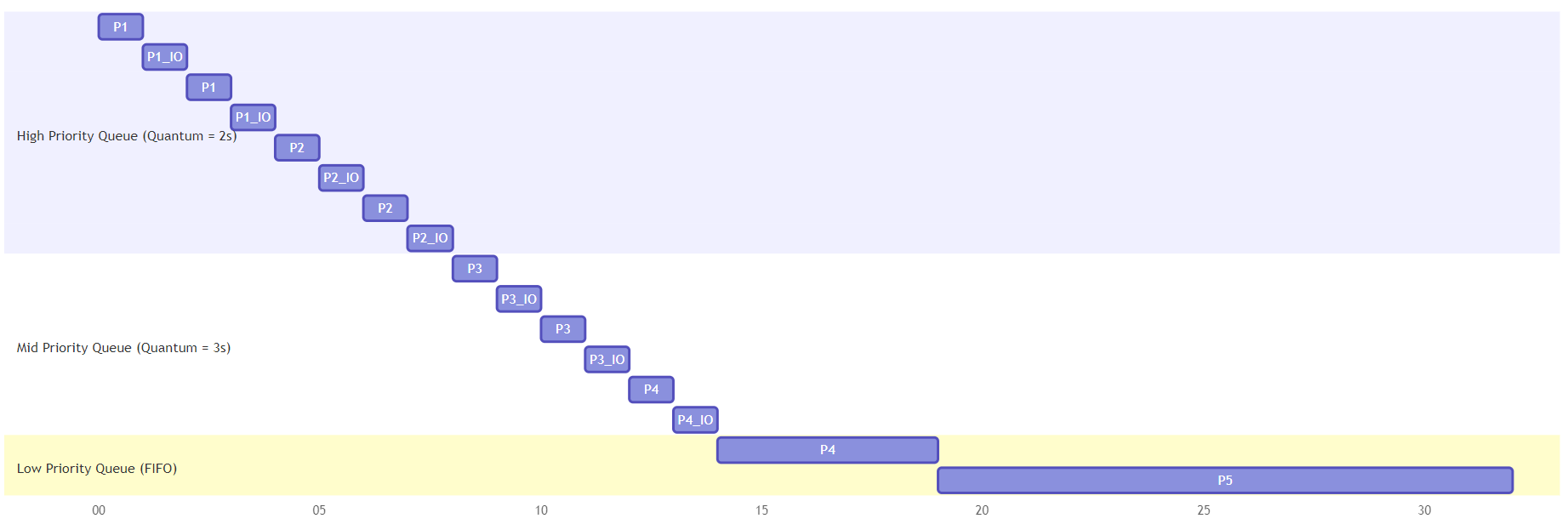
P3 = 31 – 15 = 16

P4 = 28 – 21 = 7

P5 = 34 – 24 = 10

**Mean Turnaround Time** ​= (37 + 5 + 16 + 7 + 10) / 5 = 15 sec

(iv) Multiple queues with feedback (high priority queue: quantum = 2; mid-priority queue: quantum = 3; low priority queue: FIFO



**Turnaround Times:**

P1 = 17-0 = 17

P2 = 10-6 = 4

P3 = 28-15 = 13

P4 = 33-21=12

P5 = 50-24=26

**Mean Turnaround Time** ​= (17+4+13+12+26)/5=14.4 seconds

Note: Assume low priority queue processes run to completion, and do not get preempted by IO  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Answer c) Memory management [1 mark]

Consider a multiprogrammed system that uses multiple partitions (of variable size) for memory management. A linked list of holes called the free list is maintained by the operating system to keep track of  
the available memory in the system. At a given point in time the free list consists of holes with sizes: 102K, 205K, 43K, 180K, 70K, 125K, 91K, and 150K  
The free list is also ordered in the sequence given above: the first hole in the list is of size 102K words, which is followed by a hole of size 205K words and so on. There are a number of Jobs arriving to the system with different memory requirements; they arrive in the following order:

Job No. Arrival Time Memory Requirement (words)  
1 t1 122K  
2 t2 105K  
3 t3 203K  
4 t4 90K  
[Given t1 < t2 < t3 < t4]

Determine which free partition will be allocated to each process for the following algorithms:  
(i) First Fit (ii) Best Fit (iii) Worst Fit  
Show all your work. Based on the calculations and results obtained from these algorithms, analyze which scheduling algorithm and memory management strategy is the most efficient according to different metrics. Justify your answer.

A) Memory Allocation Using Different Algorithms

(i) First Fit

In the First Fit algorithm, we allocate the first hole that is large enough for the job.

* Job 1 (122K): Allocated to 205K (remaining: 83K)
* Job 2 (105K): Allocated to 180K (remaining: 75K)
* Job 3 (203K): No allocation (not enough memory)
* Job 4 (90K): Allocated to 102K (remaining: 12K)

(ii) Best Fit

In the Best Fit algorithm, we allocate the smallest hole that is large enough for the job.

* Job 1 (122K): Allocated to 125K (remaining: 7K)
* Job 2 (105K): Allocated to 150K (remaining: 45K)
* Job 3 (203K): Allocated to 205K (remaining: 2K)
* Job 4 (90K): Allocated to 102K (remaining: 12K)

(iii) Worst Fit

In the Worst Fit algorithm, we allocate the largest hole available.

* Job 1 (122K): Allocated to 205K (remaining: 83K)
* Job 2 (105K): Allocated to 180K (remaining: 75K)
* Job 3 (203K): No allocation (not enough memory)
* Job 4 (90K): No allocation (not enough memory)

Analysis: First Fit allocated three jobs: Jobs 1, 2, and 4. However, it couldn't allocate Job 3, leaving eight holes remaining in the free list after the allocations.

Best Fit also allocated three jobs—Jobs 1, 2, and 4—while leaving Job 3 unallocated. This resulted in just four holes left in the free list, which shows it was effective at minimizing wasted space.

Worst Fit only managed to allocate two jobs, Jobs 1 and 2, leaving two jobs—Jobs 3 and 4—without allocation. This method resulted in six holes still available, highlighting its inefficiency in this scenario.

When we look at efficiency overall, Best Fit emerges as the most effective strategy. It managed to allocate three jobs while keeping the remaining memory space to a minimum, although it does run the risk of creating fragmentation over time. First Fit is quicker in terms of allocation, but it’s not as space-efficient as Best Fit. Worst Fit, meanwhile, fell short by not allocating all the jobs and leaving larger free spaces, which makes it less effective in this case.