Round Robin Scheduling

Round-Robin (RR) Scheduling Algorithm

i) Quantum is allocated to each process for execution.

ii) Once a process is executed for a given period of time, it is preempted and other process executes for a given period of time.

Example. Calculate Average Waiting time for the following:-

Process	Burst Time
P_1	21
P_2	3
P_3	6
P_4	2

Assume all process arrives at 0 and Quantum = 5.

Round- Robin (RR) Scheduling Algorithm

↳ (i) Quantum is allocated to each process for execution. — Small amount of time. (2) Pre-emptive.

(ii) Once a process is executed for a given period of time, it is preempted and other process executes for a given period of time. — time Quantum.

Example. Calculate Average Waiting time for the following:-

Process	Burst Time
P_1	21
P_2	3
P_3	6
P_4	2

Assume all process arrives at 0 and Quantum = 5.

Round-Robin is a Preemptive Scheduling Algorithm and is the most commonly used algorithm in CPU Scheduling.In this, each Process is assigned with a fix time interval to execute, which is called ***quantum.*** In the round-robin, the Process gets preempted if it is executed in the given time period, and the rest of the Processes executes accordingly. Round robin is a starvation free CPU scheduling algorithm as all the Processes get fair sharing of CPU. Round-Robin is a Preemptive version of the first-come-first-serve scheduling. Time-sharing is the objective of this algorithm. In the round-robin algorithm, each Process is cyclically executed.

In Round Robin Scheduling, CPU is assigned to the process on the basis of FCFS for a fixed amount of time. This fixed amount of time is called as **time quantum** or **time slice**. After the time quantum expires, the running process is preempted and sent to the ready queue. Then, the processor is assigned to the next arrived process. It is always preemptive in nature.

Definition: Round robin scheduling is the preemptive scheduling in which every process get executed in a cyclic way, i.e. in this a particular time slice is allotted to each process which is known as time quantum. Every process, which is present in the queue for processing, CPU is assigned to that process for that time quantum. Now, if the execution of the process gets completed in that time quantum, then the process will get terminate otherwise the process will again go to the ready queue, and the previous process will wait for the turn to complete its execution.

What do you need to know about Round Robin Scheduling?

Round-robin (RR) is one of the pre-emptive algorithms employed by process and network schedulers in computing. As the term is generally used, time slices are assigned to each process in equal portions and in circular order, handling all processes without priority. Round-robin scheduling is simple, easy to implement, and starvation-free.

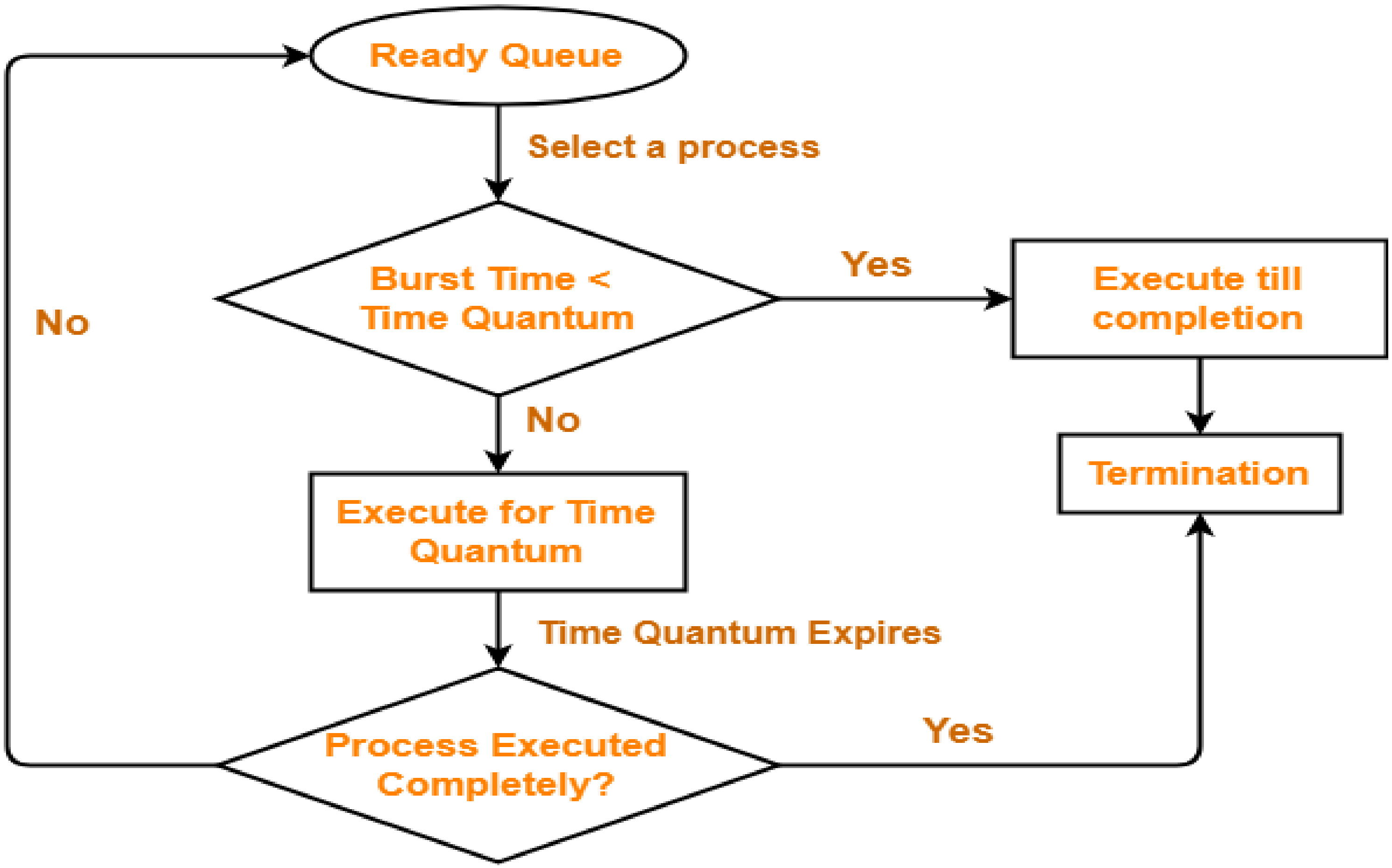
Time Quantum:

A Time Quantum (TQ) is the amount of time share give to each process, interrupting the job if it is not completed by then.

Request Queue:

A request queue is the order by which processes arrive and are handled

Round Robin Scheduling is FCFS Scheduling with preemptive mode.



Round Robin Scheduling

Characteristics of Round-Robin Scheduling

Here are the important characteristics of Round-Robin Scheduling:

- Round robin is a pre-emptive algorithm
- The CPU is shifted to the next process after fixed interval time, which is called time quantum/time slice.
- The process that is preempted is added to the end of the queue.
- Round robin is a hybrid model which is clock-driven
- Time slice should be minimum, which is assigned for a specific task that needs to be processed. However, it may differ OS to OS.
- It is a real time algorithm which responds to the event within a specific time limit.
- Round robin is one of the oldest, fairest, and easiest algorithm.
- Widely used scheduling method in traditional OS.

Advantages-

- It gives the best performance in terms of average response time.
- It is best suited for time sharing system, client server architecture and interactive system.

Disadvantages-

- It leads to starvation for processes with larger burst time as they have to repeat the cycle many times.
- Its performance heavily depends on time quantum.
- Priorities can not be set for the processes.

Advantages of the Round-Robin Scheduling Algorithm

The advantages of the round-robin scheduling algorithm are:

1. Round-Robin is independent of starvation or convoy effect.
2. In Round-Robin, every process gets a fair allocation of CPU.
3. If the number of processes present in the running queue is known, then it becomes easy to estimate the worst-case response time of the process.
4. Handles all the processes without priority.
5. Round-Robin is easy to implement.

Disadvantages of Round-Robin Scheduling Algorithm

The disadvantages of the round-robin scheduling algorithm are:

1. Average Waiting Time is Higher.
2. It depends on the length of the time slice.
3. If the time quantum is more than the response time is also more.
4. If the time quantum is less than the context switching overhead is more.

Advantages of round robin scheduling

- It is simple.
- It is easy to implement.
- It deals with all process without any priority.
- In this, all jobs get easily allocated to CPU.
- Like first come first serve scheduling, in this no problem of convoy effect or starvation is there.
- Round robin scheduling does not depend upon burst time. So it can be easily implementable on the system.

Disadvantages of round robin scheduling

- Since round robin scheduling depends upon time quantum. So deciding a perfect time quantum for scheduling is a very difficult task.
- If the time quantum is higher, then the response time of the system will also be higher.
- If the time quantum is lower, then there is higher context switching overhead.

Note-01:

With decreasing value of time quantum,

- Number of context switch increases
- Response time decreases
- Chances of starvation decreases

Thus, smaller value of time quantum is better in terms of response time.

Note-02:

With increasing value of time quantum,

- Number of context switch decreases
- Response time increases
- Chances of starvation increases

Thus, higher value of time quantum is better in terms of number of context switch.

Note-03:

- With increasing value of time quantum, Round Robin Scheduling tends to become FCFS Scheduling.
- When time quantum tends to infinity, Round Robin Scheduling becomes FCFS Scheduling.

Note-04:

- The performance of Round Robin scheduling heavily depends on the value of time quantum.
- The value of time quantum should be such that it is neither too big nor too small.

Example of Round Robin Scheduling (Preemptive)

Round-Robin (RR) Scheduling Algorithm

i) Quantum is allocated to each process for execution. Small amount of time. (2) Pre-emptive.

ii) Once a process is executed for a given period of time, it is preempted and other process executes for a given period of time. time Quantum.

Example. Calculate Average Waiting time for the following:-

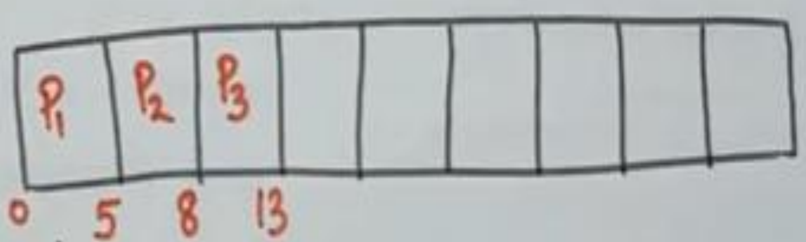
Process	Burst Time
P_1	21
P_2	3
P_3	6
P_4	2

Assume all process arrives at 0 and Quantum = 5.

ample. Calculate Average Waiting time for the following:-

Process	Burst Time
P_1	21/16
P_2	3
P_3	6/1
P_4	2

Assume all process arrives at 0 and Quantum = 5.



Gantt chart.

Example. Calculate Average Waiting time for the following:-

Process	Burst Time
P_1	21/16
P_2	3
P_3	6/1
P_4	2

Assume all process arrives at 0 and Quantum = 5.

Gantt chart.

P_1	P_2	P_3	P_4					

0 5 8 13 15

Once a process is executed for a given time Quantum.
period of time, it is preempted and other / process executes for a given period of time.

Ex. Calculate Average Waiting time for the following:-

Process	Burst Time
P_1	21/16/11/6/1
P_2	3
P_3	6/1
P_4	2

Assume all process arrives at 0 and Quantum = 5.

w.t of $P_2 = 5$
w.t of $P_4 = 13$
w.t of $P_3 = 8+7 = 15$

P_1	P_2	P_3	P_4	P_1	P_3	P_1	P_1	P_1	
0	5	8	13	15	20	21	26	31	32

Gantt chart.

(ii) Once a process is executed for a given time Quantum. period of time, it is preempted and other process executes for a given period of time.

ple. Calculate Average Waiting time for the llowing:-

Process	Burst Time
P_1	21/16/11/6/1
P_2	3
P_3	6/1
P_4	2

Assume all process arrives at 0 and Quantum = 5.

w.t of $P_2 = 5$

w.t of $P_4 = 13$

w.t of $P_3 = 8+7 = 15$

w.t of $P_1 = 0+10+1 = 11$

Average w.t $= \frac{5+13+15+11}{4}$

$= \frac{44}{4}$

10   1

P_1	P_2	P_3	P_4	P_1	P_3	P_1	P

0   5   8   13   15   20   21   2

Example of Round Robin Scheduling

In this example, we will take six processes P1, P2, P3, P4, P5 and P6 whose arrival and burst time are given in the table. The time quantum is 4 units.

Processes	Arrival Time	Burst Time
P1	0	5
P2	1	6
P3	2	3
P5	3	1
P6	4	5
P7	5	4

Now we have to maintain the ready queue and gantt chart in the algorithm again and again as their structures get changed after every scheduling.

Ready Queue

Initially, at time 0, the process P1 will be executed for the 4 units of time quantum. So, in the starting in ready queue, there will be only one process P1.

P1
5

Gantt chart

P1 will be executed for 4 units.

Round robin Gantt Chart 1

Ready Queue

Midwhile, during the execution of process P1, some other processes like P2, P3,P4 and P5 arises for the execution in the ready queue. As we know P1 has not completed since its time is 5 and we have 4 units of time slice. So, 1 unit is still left. So it will be again added at the back in ready queue.

P2	P3	P4	P5	P1
6	3	1	5	1

GANTT chart

Now the gantt chart will be like this.

Round robin Gantt Chart 2

Ready queue

During the execution time of P2, another process P6 arrives in the ready queue. Also, we know that P2 has not completed yet as its 2 units of burst time is still left. So, again we will add P2 in the ready queue at the back.

P3	P4	P5	P1	P6	P2

Ready queue

During the execution time of P2, another process P6 arrives in the ready queue. Also, we know that P2 has not completed yet as its 2 units of burst time is still left. So, again we will add P2 in the ready queue at the back.

P3	P4	P5	P1	P6	P2
3	1	5	1	4	2

GANTT CHART

Now, P3 will be executed for 3 units of time slice as its bursts time is 3 units.

Round robin Gantt Chart 3

Ready queue

Now the process P3 is completed in the time slice of 4 units. So, we will not add P3 in the ready queue and now we will execute the next process P4.

P4	P5	P6	P1	P2
1	5	1	4	2

GANTT CHART

Round robin Gantt Chart 4

Ready queue

The next process in the queue is P5 which has 5 units of bursts time. Again, the process P4 gets completed as it has only 1 unit of bursts time.

P5	P6	P1	P2
5	1	4	2

GANTT chart

Round robin Gantt Chart 5

GANTT chart

Round robin Gantt Chart 5

Ready Queue

Now the process has not completed as 1 unit of burst time is left. So we add it in the ready queue in the back.

P1	P6	P2	P5
1	4	2	1

GANTT chart

Now the process P1 will be executed so that it can complete its execution as its turn has come. As we know it requires only 1 unit of bursts time, so it will get completed.

Round robin Gantt Chart 6

Ready queue

As P1 also get completed. So , we will not add it to the ready queue. Now only three processes are left which are P6, P2 and P5.

P6	P2	P5
4	2	1

GANTT CHART

Now P6 has units of 4 units of bursts time. So it will get executed in 4 units of time slice.

Round robin Gantt Chart 7

Ready queue

Since P6 is executed completely. so, we will not add it to the ready queue. Now only 2 processes left in the ready queue which are P2 and P5.

P2	P5
2	1

GANTT CHART

Now P2 will be executed again and it requires only 2 units of bursts time. So now it will be completed.

Round robin Gantt Chart 8

Ready queue

Only process left in the ready queue is P5 which requires only 1 unit of bursts time. As we know our time slice is of 4 units. So it will get completed in the next burst.

P5
1

GANTT CHART

The process P5 will get executed until it get completed as it is the only process left in the ready queue.

Round robin Gantt Chart 9

Now we will calculate turn around time, completion time and average waiting time which is shown in the below table.

Turn around time= completion time- arrival time

Waiting time= turn around time - burst time

Process	Arrival Time	Burst time	Completion time	Turn around time	Waiting time
P1	0	5	17	17	12
P2	1	6	23	22	16
P3	2	3	11	9	6
P4	3	1	12	9	8
P5	4	5	24	20	15
P6	6	4	21	15	11

Average waiting time= (12+16+6+8+15+11)/6= 76/6 = 12.66 units

Example of Round-Robin Scheduling Algorithm

In the following explained example, we have 4 processes with process ID P1, P2, P3, and P4. The arrival time and burst time of the proceses are given in the following table. **(The Quantum time is 6).**

Process ID	Arriving Time	Burst Time	Completion Time	Turnaround Time	Waiting Time
P1	0	8	25	25	17
P2	1	5	11	10	5
P3	2	10	29	27	17
P4	3	11	34	31	20

Gantt chart

P1	P2	P3	P4	P1	P3	P4

0 6 11 17 23 25 29 34

Steps for Gantt Chart

Step 1. In the above Gantt chart, firstly the Process P1 starts its execution, which has burst time = 8. But each process is executing only for 6 seconds because quantum time is 6. The process P2, P3, and P4 are in the waiting queue.

Step 2. At time=6, the process P1 is added to the end of the queue, and the process P2 starts its execution.

Step 3. At time=11, the process P2, which has the burst time=5, completes its execution in the given time of quantum.

Step 4. After P2, the process P3 starts its execution, which has burst time=10, but again it is also executed for 6 seconds due to the limit of quantum time. At time=17, the process P3 is added to the end of the queue.

Step 5. After P3, the next process P4 starts its execution, the burst time of P4 is 11, but it also executes for only 6 seconds.

Step 6. After P4, the process P1 again starts its execution because P1 Process left its execution due to less quantum time. So, process P1 again starts its execution to complete its execution.

Step 7. At time=29, P3 completes its execution, and P4 starts its execution again. The burst time of P4 is 11, and it is also executed for 6 intervals. At time=34, the process P4 completes its execution.

The waiting time and Turnaround time are calculated with the help of the following formula.

Waiting Time = Turnaround time – Burst Time

Turnaround Time = Completion time – Arrival time

Process Turnaround Time

P1= 25-0 =0

P2= 11-1=10

P3= 29-2 = 27

P4= 34-3=31

Average Turnaround Time= 0+10+27+31/4

Average Turnaround Time= 0+10+27+31/4

= 68/4

=17

Process Waiting Time

P1= 25-8=17

P2=10-5=5

P3=27-10=17

P4=31-11=20

Average Waiting Time=17+5+17+20/4

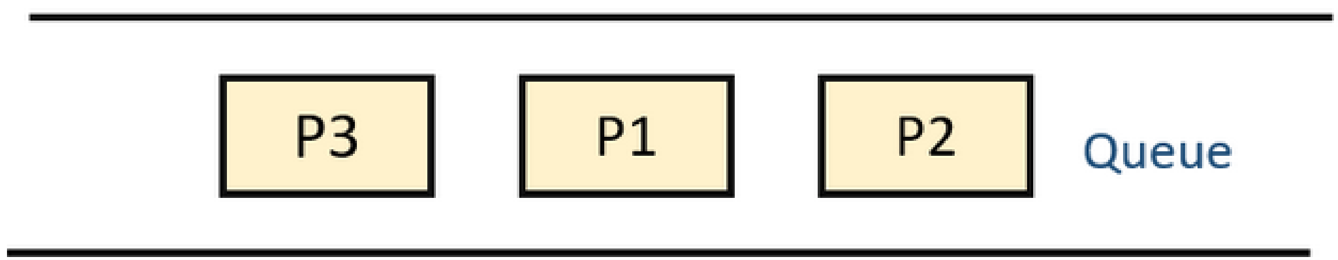
=59/4

=14.75

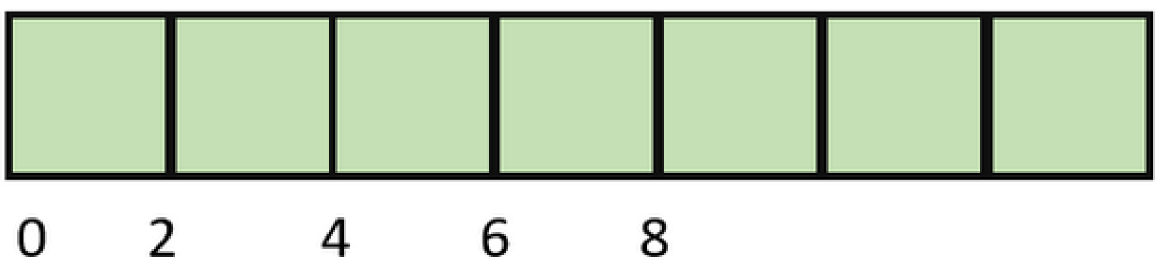
Example of Round-robin Scheduling

Consider this following three processes

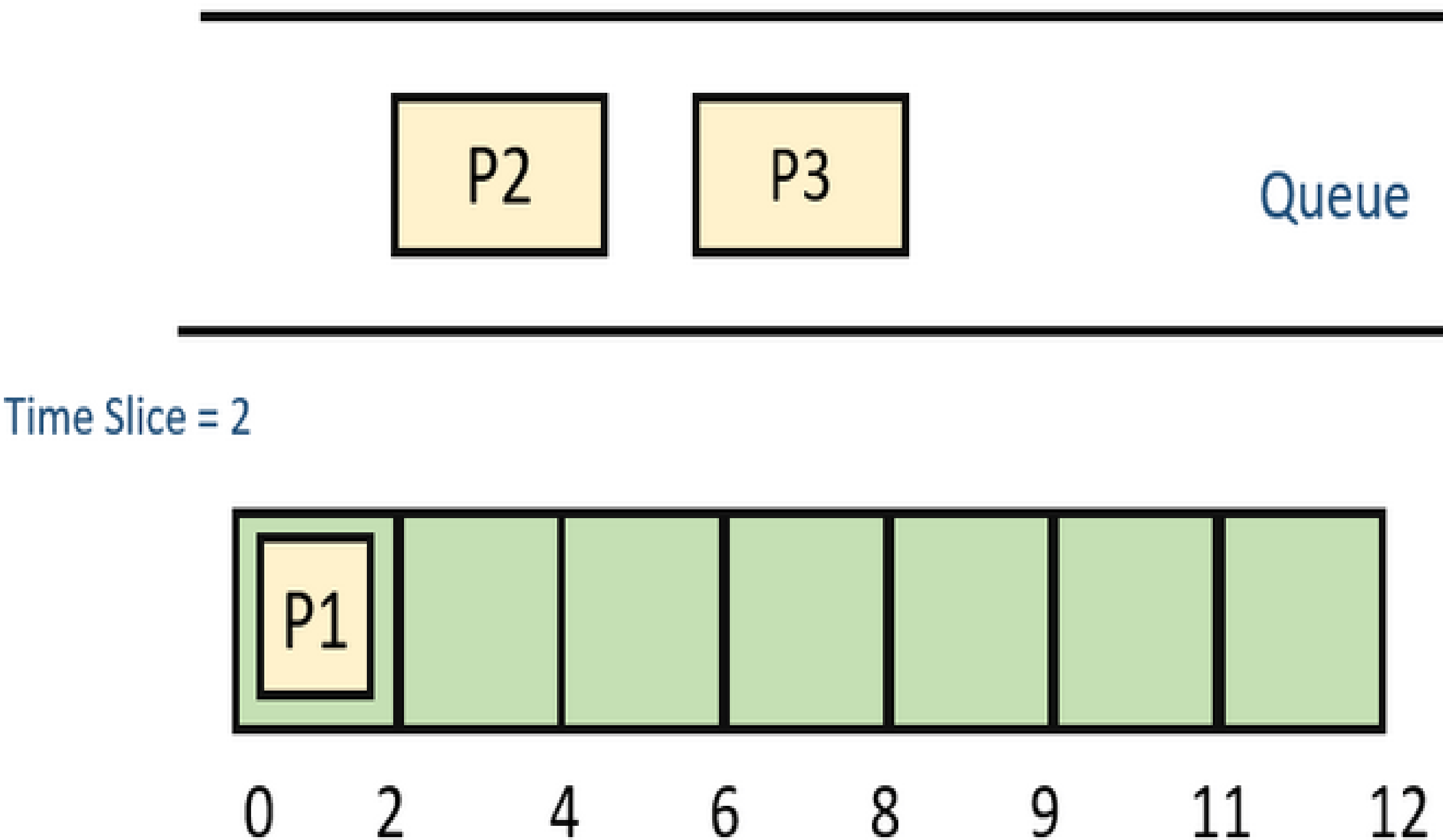
Process Queue	Burst time
P1	4
P2	3
P3	5



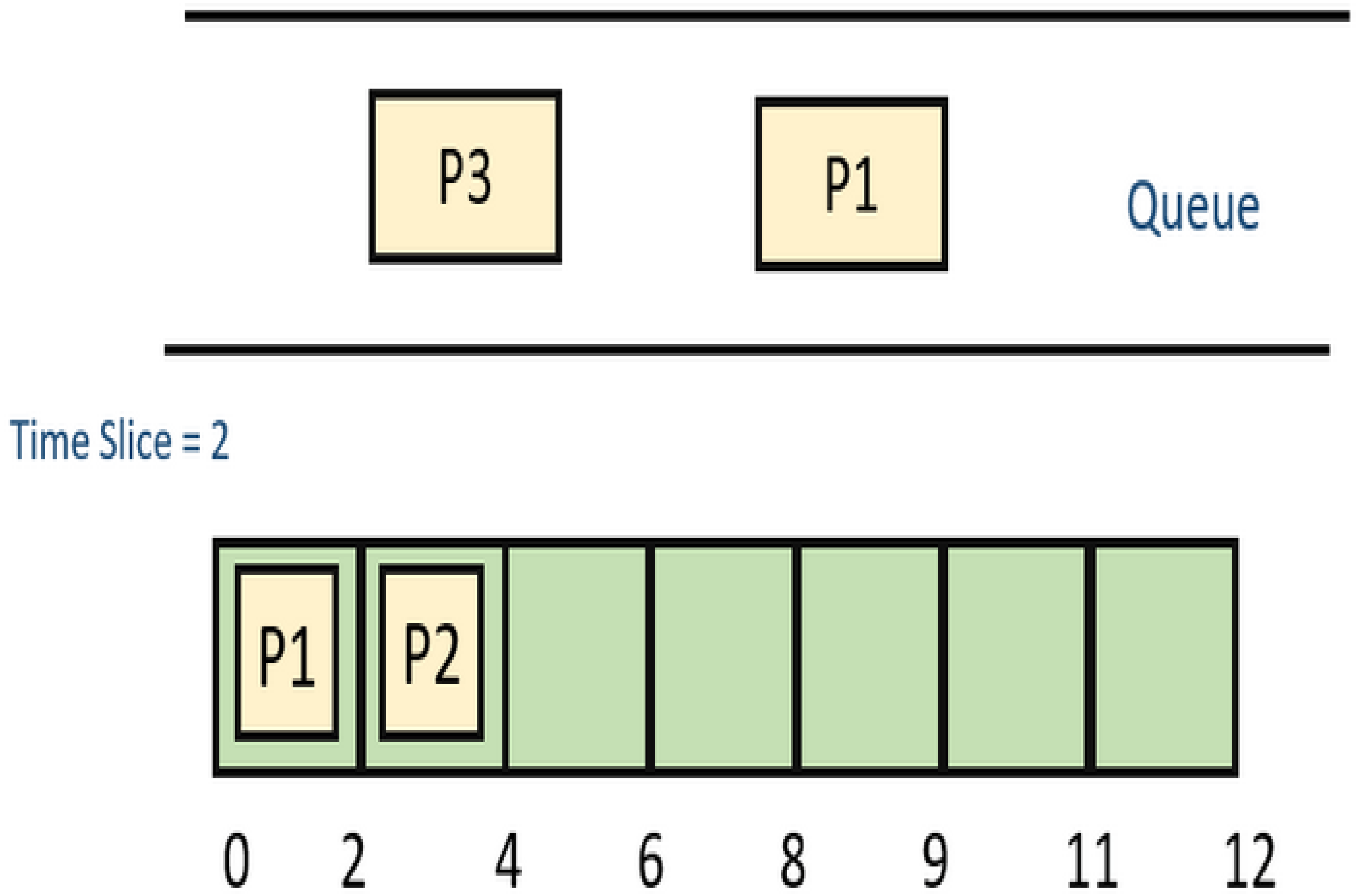
Time Slice = 2



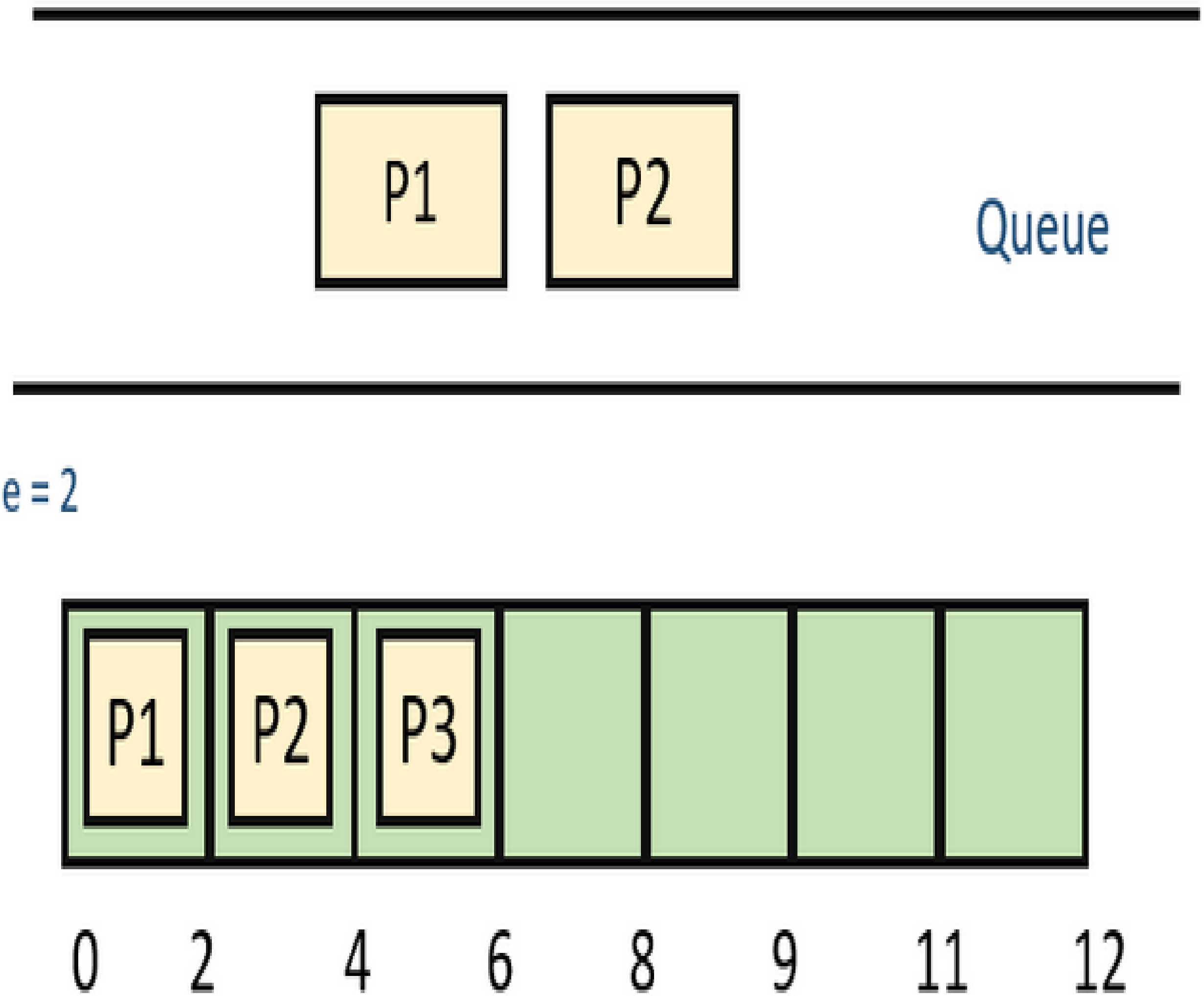
Step 1) The execution begins with process P1, which has burst time 4. Here, every process executes for 2 seconds. P2 and P3 are still in the waiting queue.



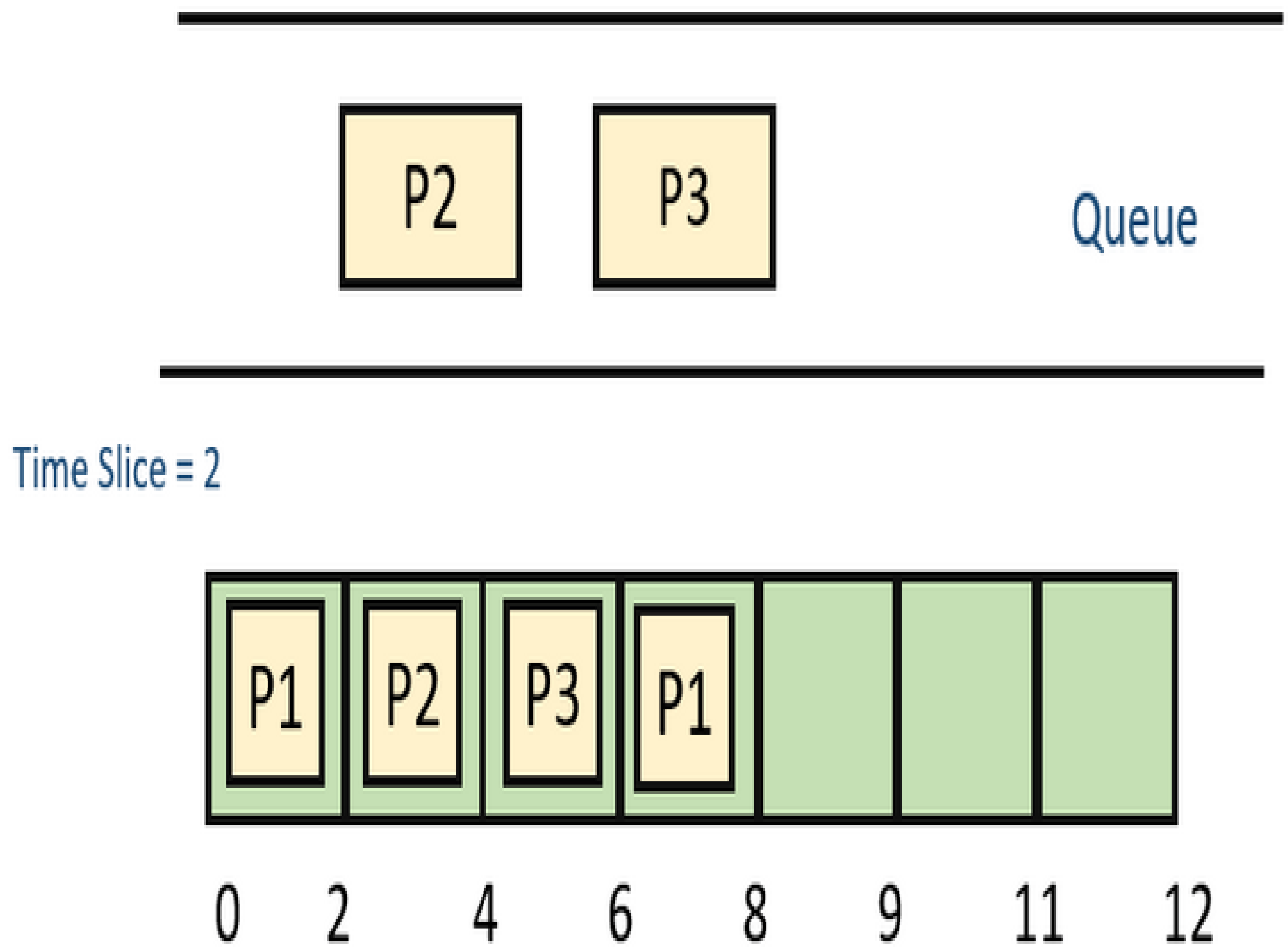
Step 2) At time =2, P1 is added to the end of the Queue and P2 starts executing



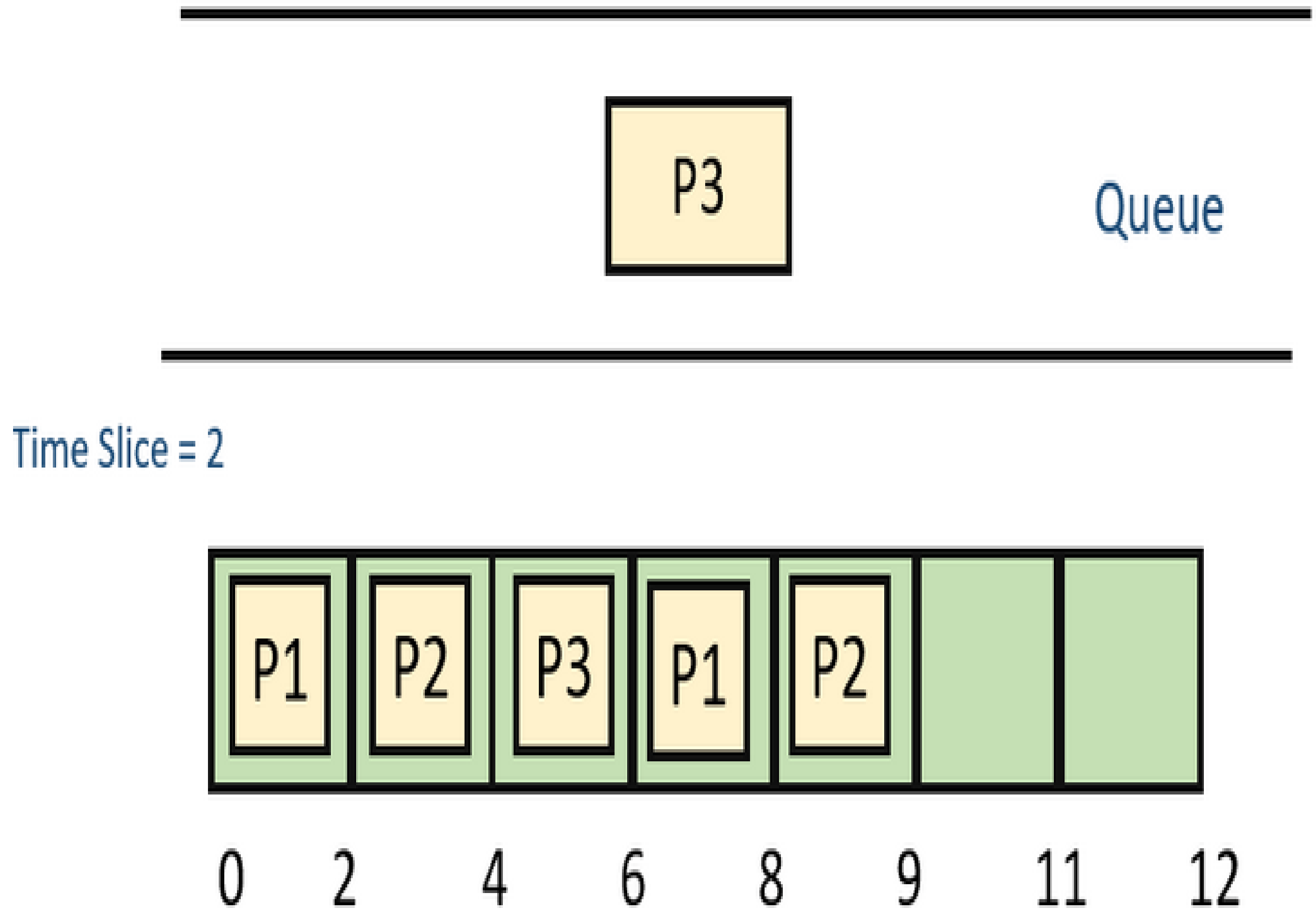
Step 3) At time=4 , P2 is preempted and add at the end of the queue. P3 starts executing.



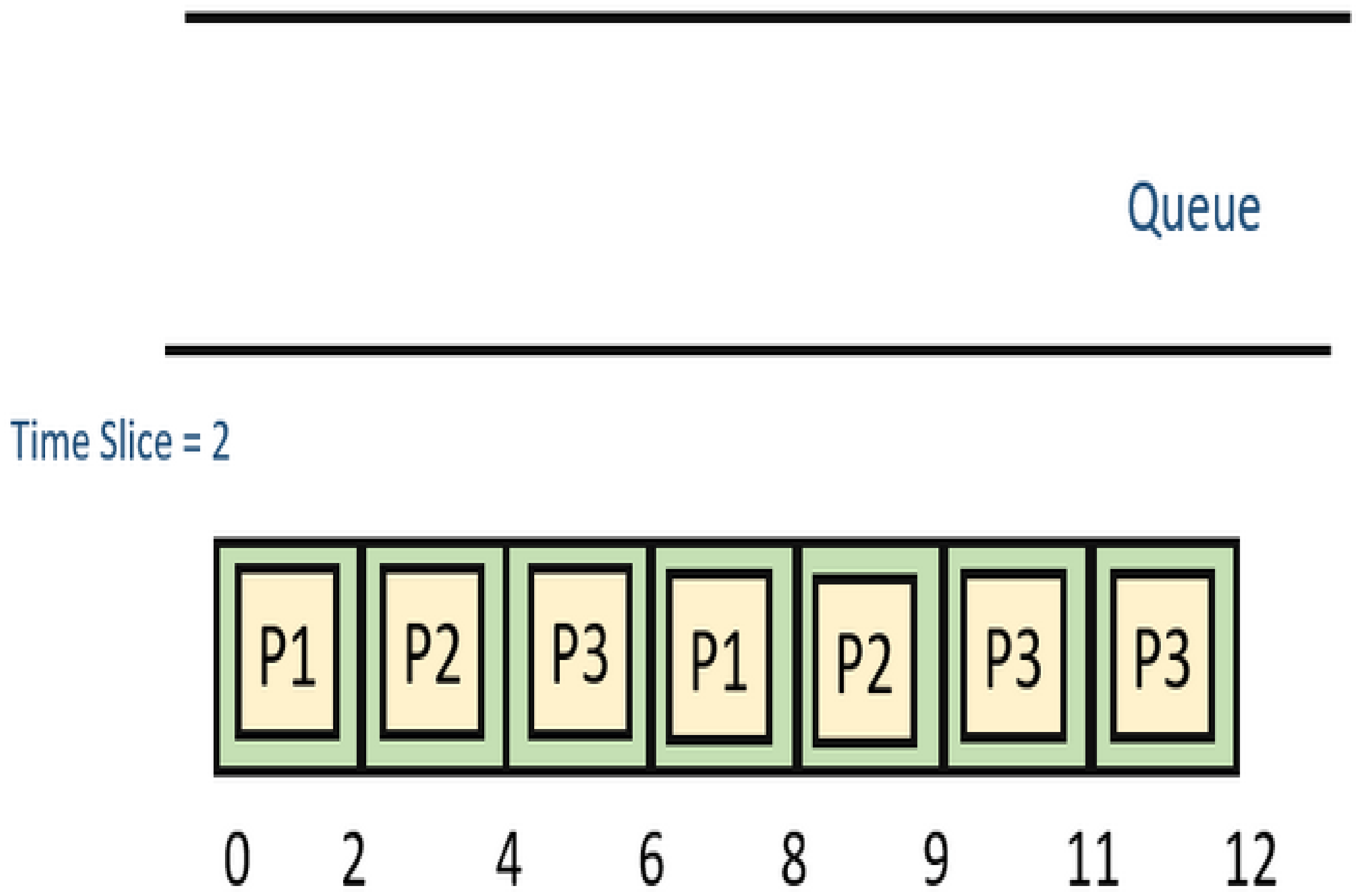
Step 4) At time=6 , P3 is preempted and add at the end of the queue. P1 starts executing.



Step 5) At time=8 , P1 has a burst time of 4. It has completed execution. P2 starts execution



Step 6) P2 has a burst time of 3. It has already executed for 2 interval. At time=9, P2 completes execution. Then, P3 starts execution till it completes.



Step 7) Let's calculate the average waiting time for above example.

```
Wait time
P1= 0+ 4= 4
P2= 2+4= 6
P3= 4+3= 7
```

Preemptive Round Robin Scheduling(Practice Questions)

PRACTICE PROBLEMS BASED ON ROUND ROBIN SCHEDULING-

Problem-01:

Consider the set of 5 processes whose arrival time and burst time are given below-

Process Id	Arrival time	Burst time
P1	0	5
P2	1	3
P3	2	1
P4	3	2
P5	4	3

If the CPU scheduling policy is Round Robin with time quantum = 2 unit, calculate the average waiting time and average turn around time.

Solution-

Gantt Chart-

Ready Queue-

P5, P1, P2, P5, P4, P1, P3, P2, P1

Gantt Chart

Now, we know-

- Turn Around time = Exit time – Arrival time
- Waiting time = Turn Around time – Burst time

Process Id	Exit time	Turn Around time	Waiting time
P1	13	13 – 0 = 13	13 – 5 = 8
P2	12	12 – 1 = 11	11 – 3 = 8
P3	5	5 – 2 = 3	3 – 1 = 2
P4	9	9 – 3 = 6	6 – 2 = 4
P5	14	14 – 4 = 10	10 – 3 = 7

Now,

- Average Turn Around time = (13 + 11 + 3 + 6 + 10) / 5 = 43 / 5 = 8.6 unit
- Average waiting time = (8 + 8 + 2 + 4 + 7) / 5 = 29 / 5 = 5.8 unit

Assignment#05

Questions on Round-Robin CPU Scheduling(Pre-emptive)

Ques1. Calculate Waiting Time and Turn Around Time(TQ=2)

Process Number	Arrival Time	Burst Time	Completion Time	T.A.T	W.T
P1	0	4			
P2	1	5			
P3	2	2			
P4	3	1			
P5	4	6			
P6	5	3			

Questions on Round-Robin CPU Scheduling(Pre-emptive)

Ques2. Calculate Waiting Time and Turn Around Time(TQ=3)

Process Number	Arrival Time	Burst Time	Completion Time	T.A.T	W.T
P1	5	5			
P2	4	6			
P3	3	7			
P4	1	9			
P5	2	2			
P6	6	3			

Round Robin Scheduling

THEORY

Round Robin Scheduling (RR)

In interactive environments, such as time-sharing systems, the primary requirement is to provide reasonably good response time and, in general, to share system resources equitably among all users. Obviously, only preemptive disciplines may be considered in such environments, and one of the most popular is *time slicing*, also known as round robin.

In this scheme, the processor time is divided into *slices* or *quanta* which are allocated to processes waiting in a ready queue. A time quantum is generally from 1 to 100 milliseconds. The ready queue is treated as a circular queue. The RR scheduling attempts to reduce the penalty that short jobs suffer with FCFS by the use of preemption based on clock.

Implementation and Algorithm

AIM :

To write a C program to simulate the CPU scheduling algorithm for round robin

PROBLEM DESCRIPTION:

CPU scheduler will decide which process should be given the CPU for its execution .For this it use different algorithm to choose among the process .one among that algorithm is Round robin algorithm.

In this algorithm we are assigning some time slice .The process is allocated according to the time slice ,if the process service time is less than the time slice then process itself will release the CPU voluntarily .The scheduler will then proceed to the next process in the ready queue .If the CPU burst of the currently running process is longer than time quantum ,the timer will go off and will cause an interrupt to the operating system .A context switch will be executed and the process will be put at the tail of the ready queue.

Algorithm

ALGORITHM:

Step 1: Initialize all the structure elements

Step 2: Receive inputs from the user to fill process id,burst time and arrival time.
Step 3: Calculate the waiting time for all the process id.

i) The waiting time for first instance of a process is calculated as: a[i].waittime=count + a[i].arrivt

ii) The waiting time for the rest of the instances of the process is calculated as:

a) If the time quantum is greater than the remaining burst time then waiting time is calculated as:

a[i].waittime=count + tq

b) Else if the time quantum is greater than the remaining burst time then waiting time is calculated as:

a[i].waittime=count - remaining burst time

Step 4: Calculate the average waiting time and average turnaround time
Step 5: Print the results of the step 4.

Task

Implementation

Implement Round Robin CPU Scheduling Algorithm using C,C++. Also attach the snapshot of Code and output window.

Attach Task Code

Output

```
Enter Total Process:     4
Enter Arrival Time and Burst Time for Process Process Number 1 :0 9
Enter Arrival Time and Burst Time for Process Process Number 2 :1 5
Enter Arrival Time and Burst Time for Process Process Number 3 :2 3
Enter Arrival Time and Burst Time for Process Process Number 4 :3 4
Enter Time Quantum:     5


Process |Turnaround Time|Waiting Time

P[2]    |       9       |       4
P[3]    |       11      |       8
P[4]    |       14      |       10
P[1]    |       21      |       12

Average Waiting Time= 8.500000
Avg Turnaround Time = 13.750000
--------------------------------
Process exited after 45.46 seconds with return value 0
Press any key to continue . . .
```