**STUDENT NAME:- B.LAKSHMIPATHI ACHARI**

**STUDENT ID:-11802394**

**EMAIL:-lakshmipathichari@gmail.com**

**GITHUB LINK:-**

[2020]

**CASE STUDY[CSE316]**

**OPERATING SYSTEMS (CSE 316)**

**CONTEXT**

1. INTRODUCTION
2. DESCRIPTION
3. ALGORITHM
4. CODE SNIPPET
5. COMPLEXITY
6. BOUNDARY CONDITIONS
7. TEST CASES
8. GITHUB REPOSITORY
9. CONCLUSION

**INTRODUCTION**

The scheduler given to me is nothing but ROUND ROBIN ALGORITHM which schedule the processes arriving system at periodical intervals.Every process is assigned with a fixed time slice t milliseconds.so that the given scheduler is round robin scheduling algorithm.in this I am going to explain about this algorithm and its solution and purposes and code of solution and test with test cases.In this I am going to take the arrival and burst time and time quantum from the user and calculating the total time and waiting time.

**DESCRIPTION**

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.

* It is simple, easy to implement, and starvation-free as all processes get fair share of CPU.
* One of the most commonly used technique in CPU scheduling as a core.
* It is preemptive as processes are assigned CPU only for a fixed slice of time at most.
* The disadvantage of it is more overhead of context switching.

**TIMES IN ROUND ROBIN**

* Completion Time: Time at which process completes its execution.
* Turn Around Time: Time Difference between completion time and arrival time. Turn Around Time = Completion Time – Arrival Time
* Waiting Time(W.T): Time Difference between turn around time and burst time.  
  Waiting Time = Turn Around Time – Burst Time

**ADVANTAGES**

* There is fairness since every process gets equal share of CPU.
* While performing a round-robin scheduling,a particular time quantum is alloted to different jobs
* Each process get a chance to reschedule after a particular quantum time in this scheduling
* The newly created process is added to end of ready queue.

**DISADVANTAGES**

* There is Low throughput.
* Gantt chart seems to come too big (if quantum time is less for scheduling.)
* Time consuming scheduling for small quantums
* There is Larger waiting time and Response time

**ALGORITHM**

**STEP1:-**We first have a queue where the processes are arranged in first come first serve order.

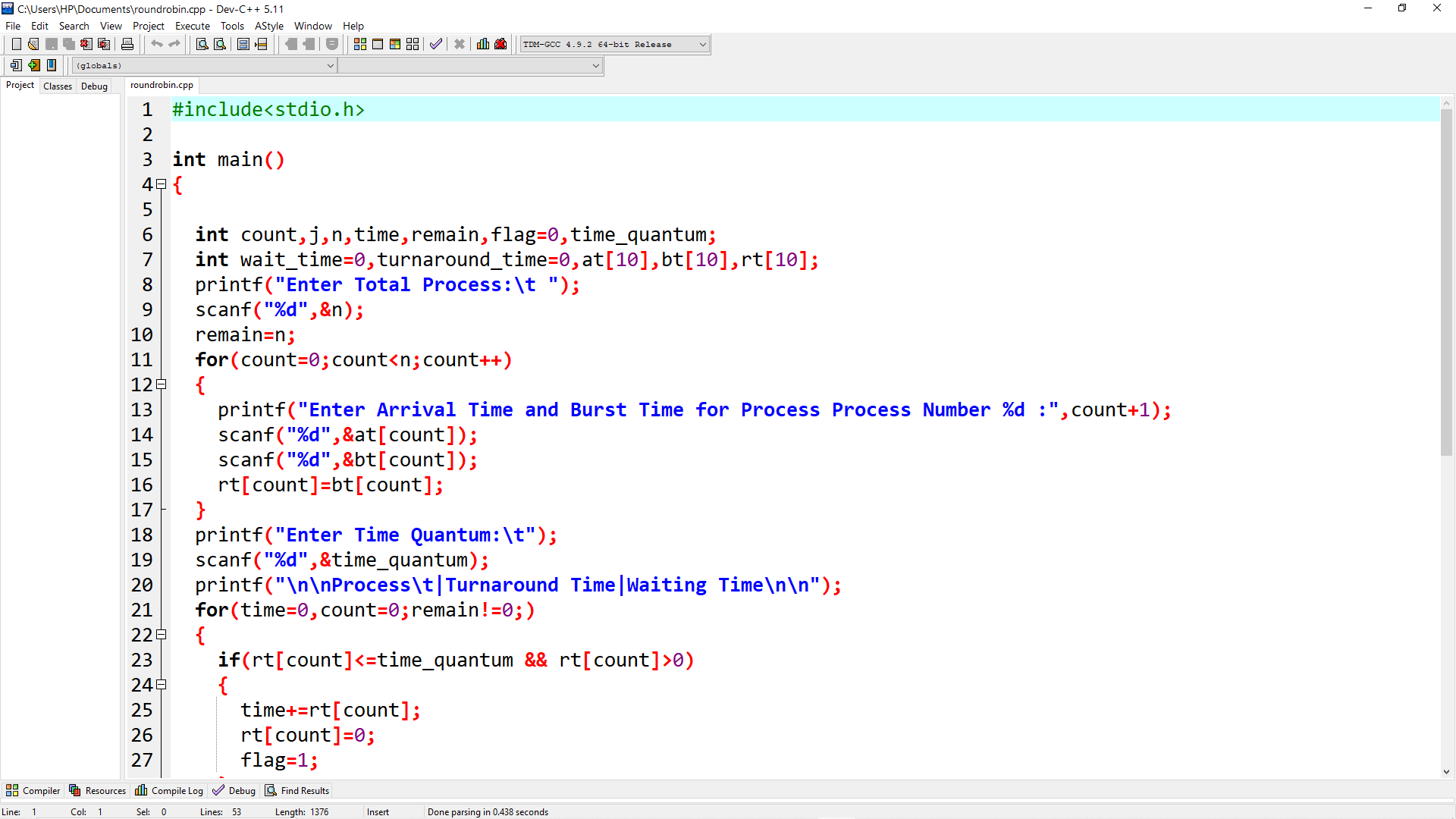
**STEP2:-**A quantum value is allocated to execute each process.

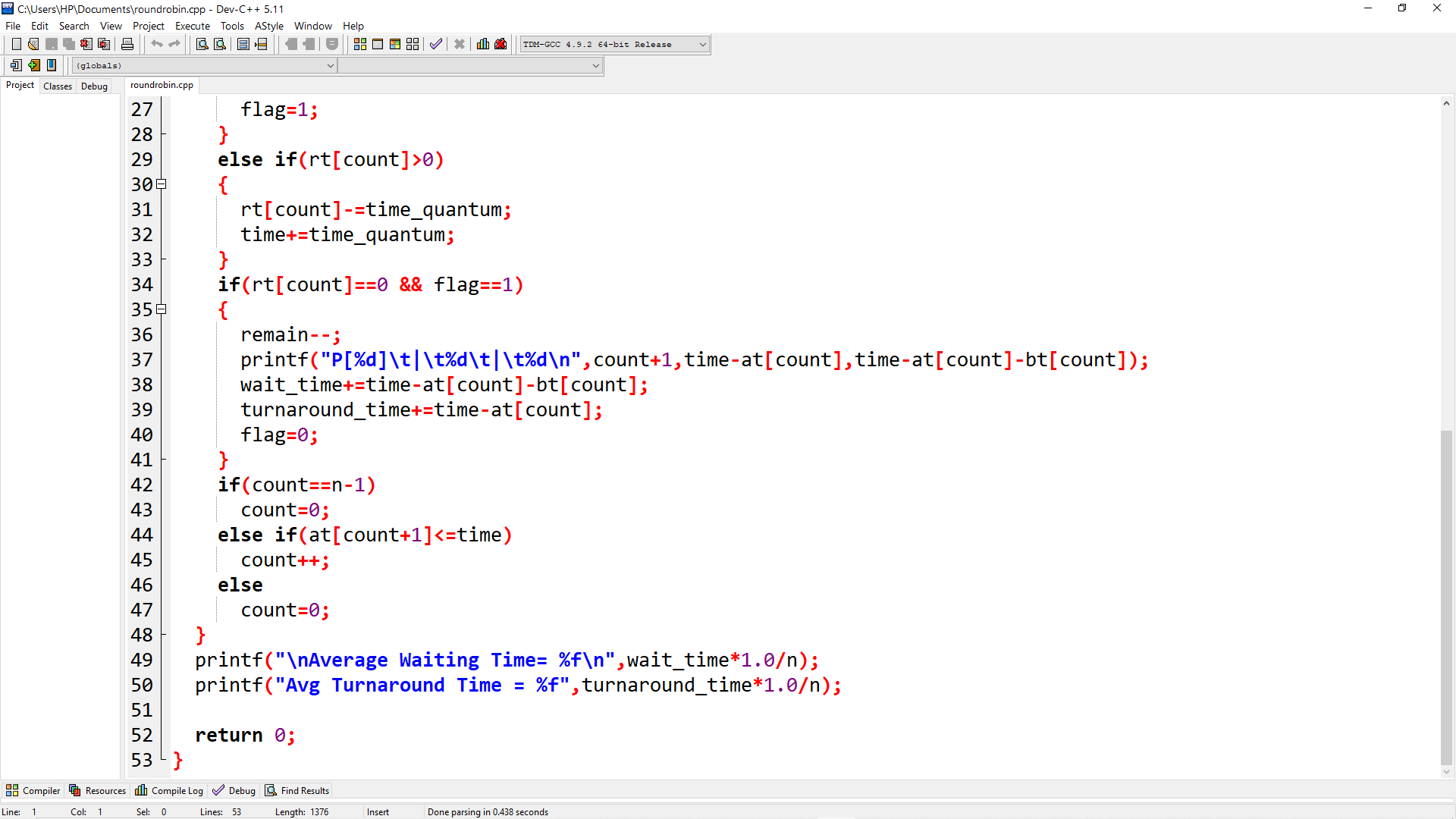
**STEP3:-**The first process is executed until the end of the quantum value. After this, an interrupt is generated and the state is saved.

**STEP4:-**The CPU then moves to the next process and the same method is followed.

**STEP5:-**Same steps are repeated till all the processes are over.

**CODE SNIPPET**

****



**COMPLEXITY**

The basic approach for solving this problem is to apply round robin algorithm with time slice 1. But the time complexity of that approach will be O(ΣAi) i.e. summation of all process’s time, which is quite high.

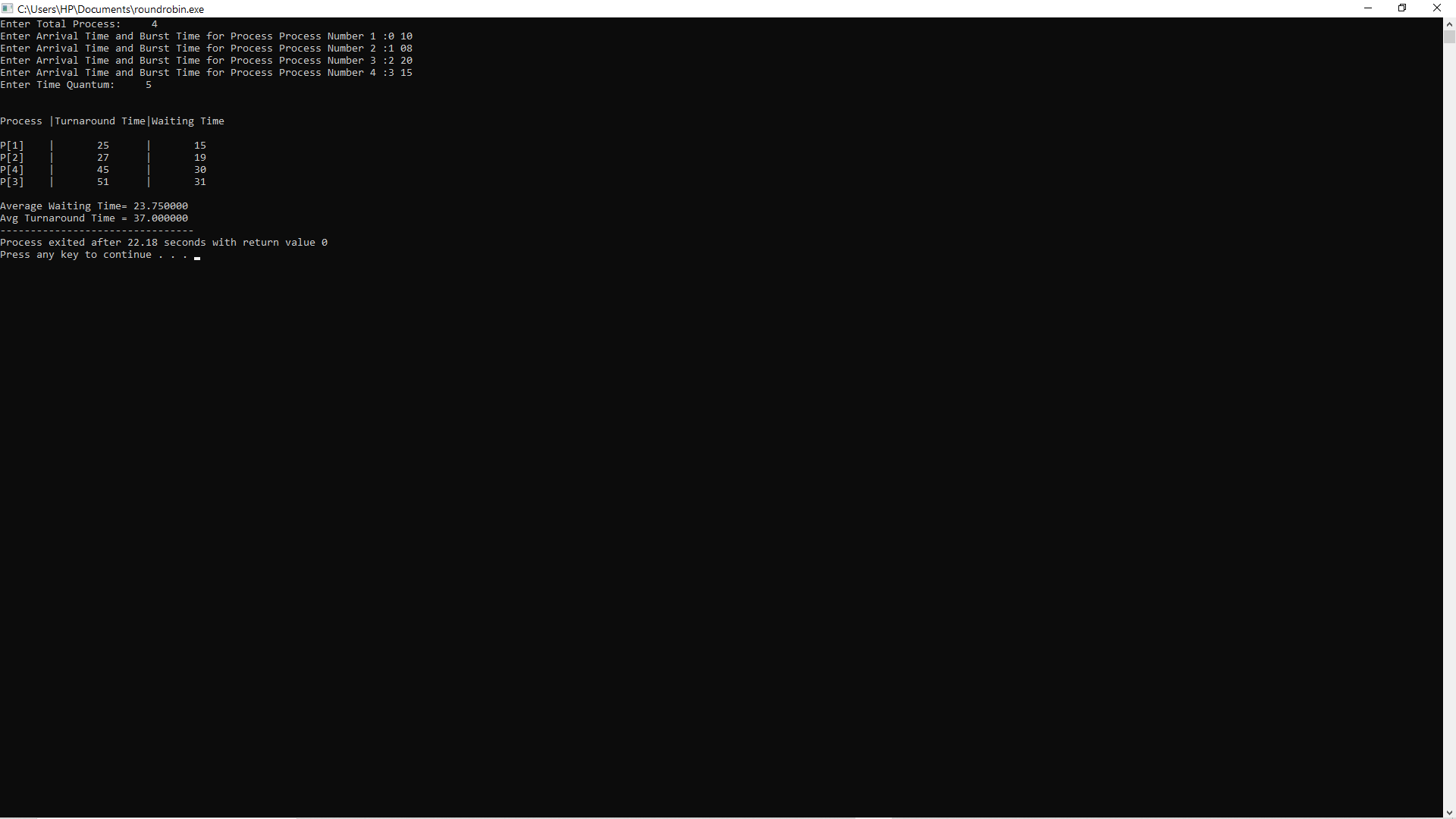
**TEST CASES**

1.Time quantum=5 sec

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Processes | Arrival time | Burst time | Turnaround time | Waiting time |
| P1 | 0 | 10 | 25 | 15 |
| P2 | 1 | 08 | 27 | 19 |
| P3 | 2 | 20 | 51 | 31 |
| P4 | 3 | 15 | 45 | 30 |

Avg waiting time=23.75

Avg turnaround time=37.00

**AFTER EXECUTION**The output is:-

2.Time quantum=15 sec

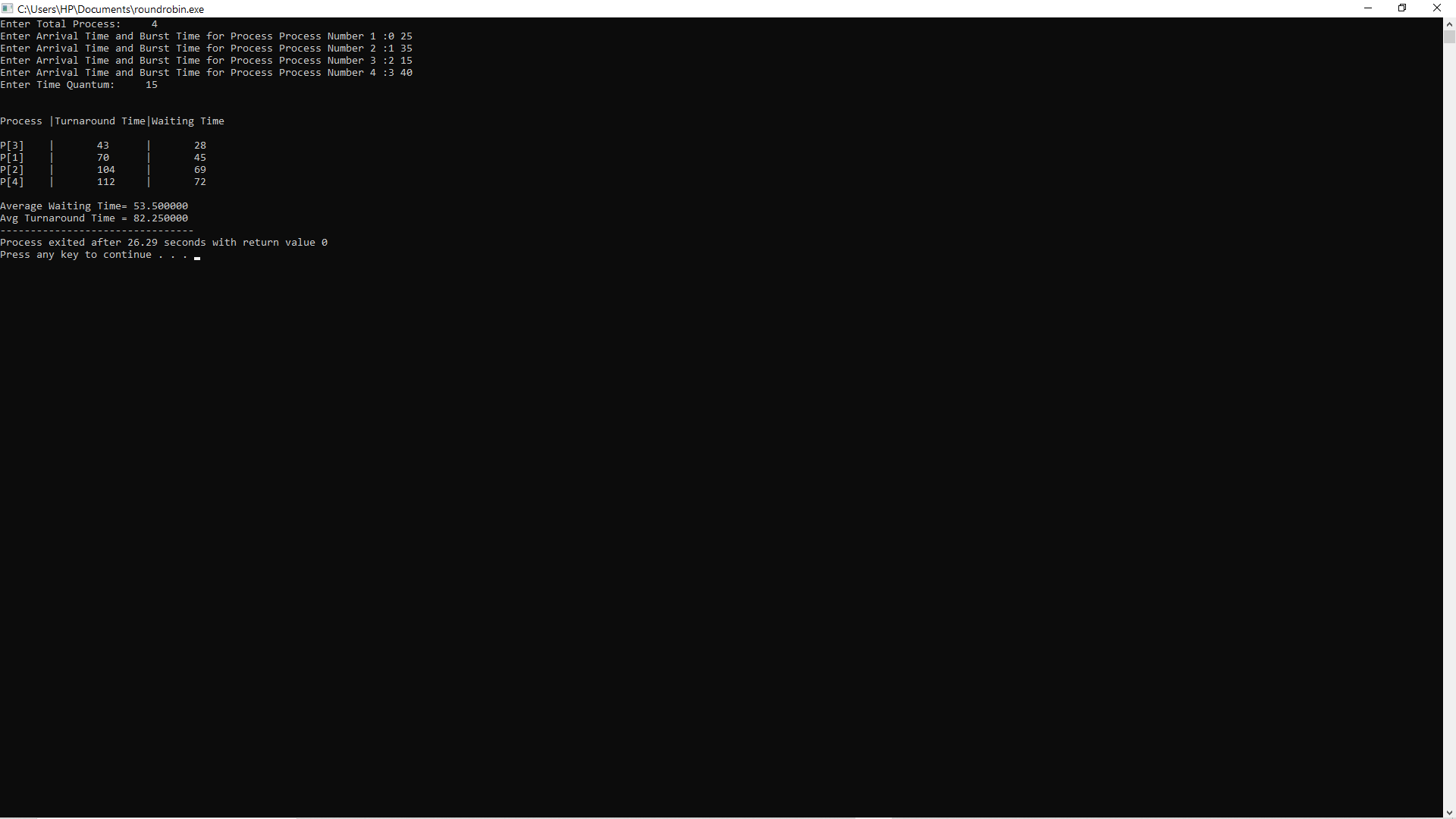
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Processes | Arrival time | Burst time | Turnaround time | Waiting time |
| P1 | 0 | 25 | 70 | 45 |
| P2 | 1 | 35 | 104 | 69 |
| P3 | 2 | 15 | 43 | 28 |
| P4 | 3 | 40 | 112 | 72 |

Avg waiting time=53.50

Avg turnaround time=82.25

**AFTER EXECUTION**

The output is:-



3.Time quantum=12 sec

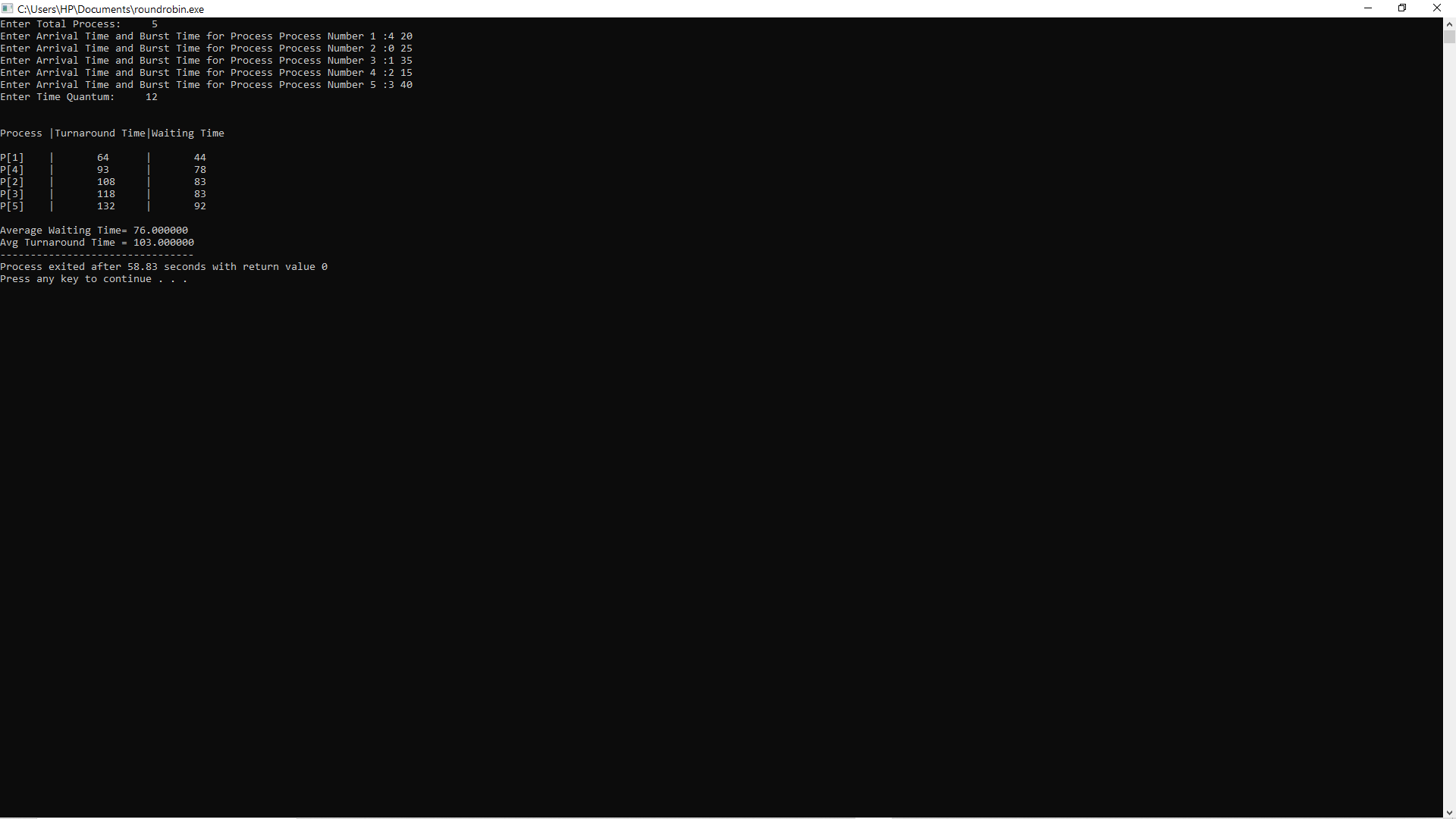
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Processes | Arrival time | Burst time | Turnaround time | Waiting time |
| P1 | 4 | 20 | 64 | 44 |
| P2 | 0 | 25 | 108 | 83 |
| P3 | 1 | 35 | 118 | 83 |
| P4 | 2 | 15 | 93 | 78 |
| P5 | 3 | 40 | 132 | 92 |

Avg waiting time=76.00

Avg turnaround time=103.00

**AFTER EXECUTION**

The output is:-



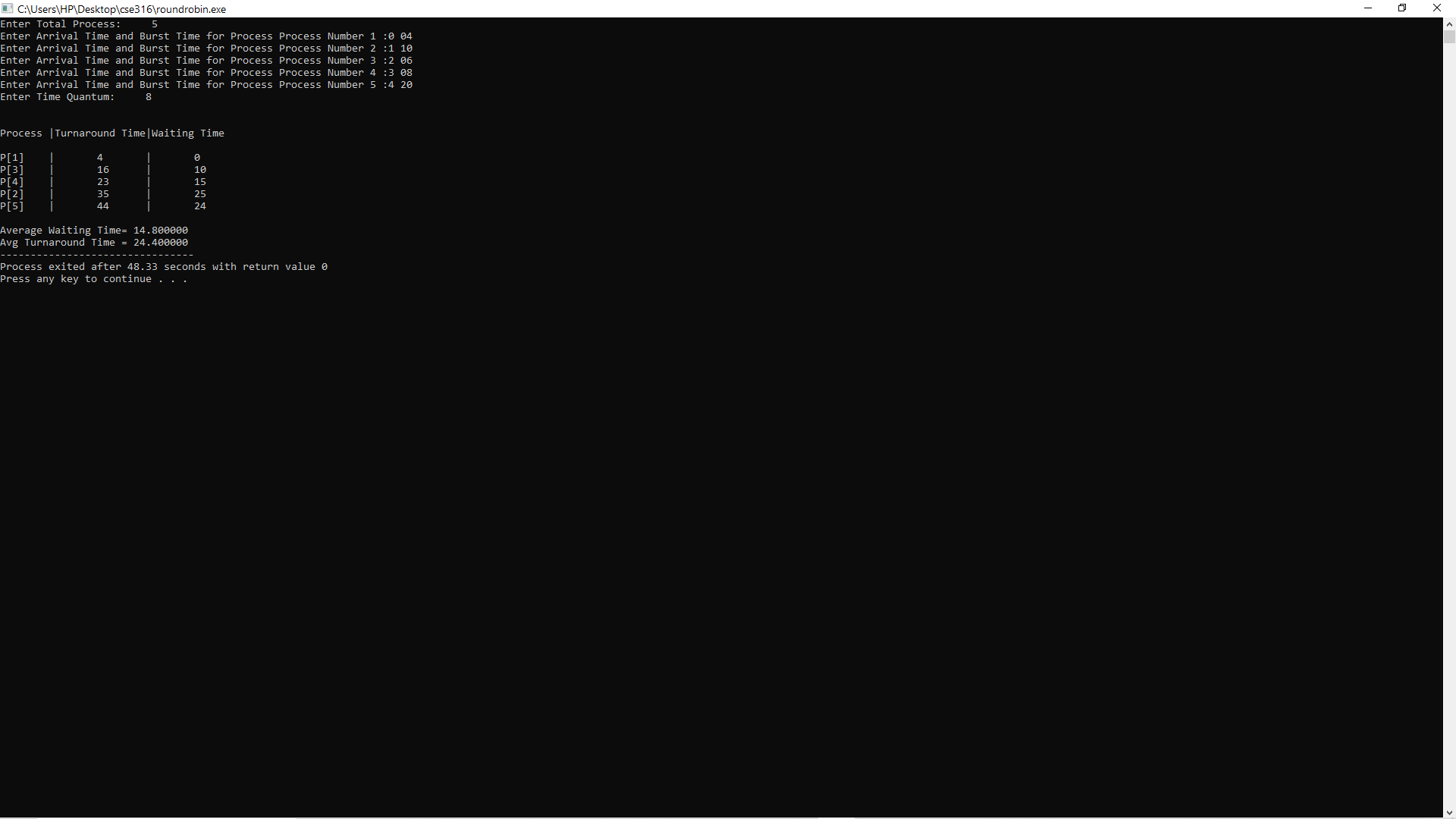
4.Time quantum=8 sec

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Processes | Arrival time | Burst time | Turnaround time | Waiting time |
| P1 | 0 | 04 | 04 | 0 |
| P2 | 1 | 10 | 35 | 25 |
| P3 | 2 | 06 | 16 | 10 |
| P4 | 3 | 08 | 23 | 15 |
| P5 | 4 | 20 | 44 | 24 |

Avg waiting time=14.80

Avg turnaround time=24.40

**AFTER EXECUTION**

The output is:-

5.Time quantum=10sec

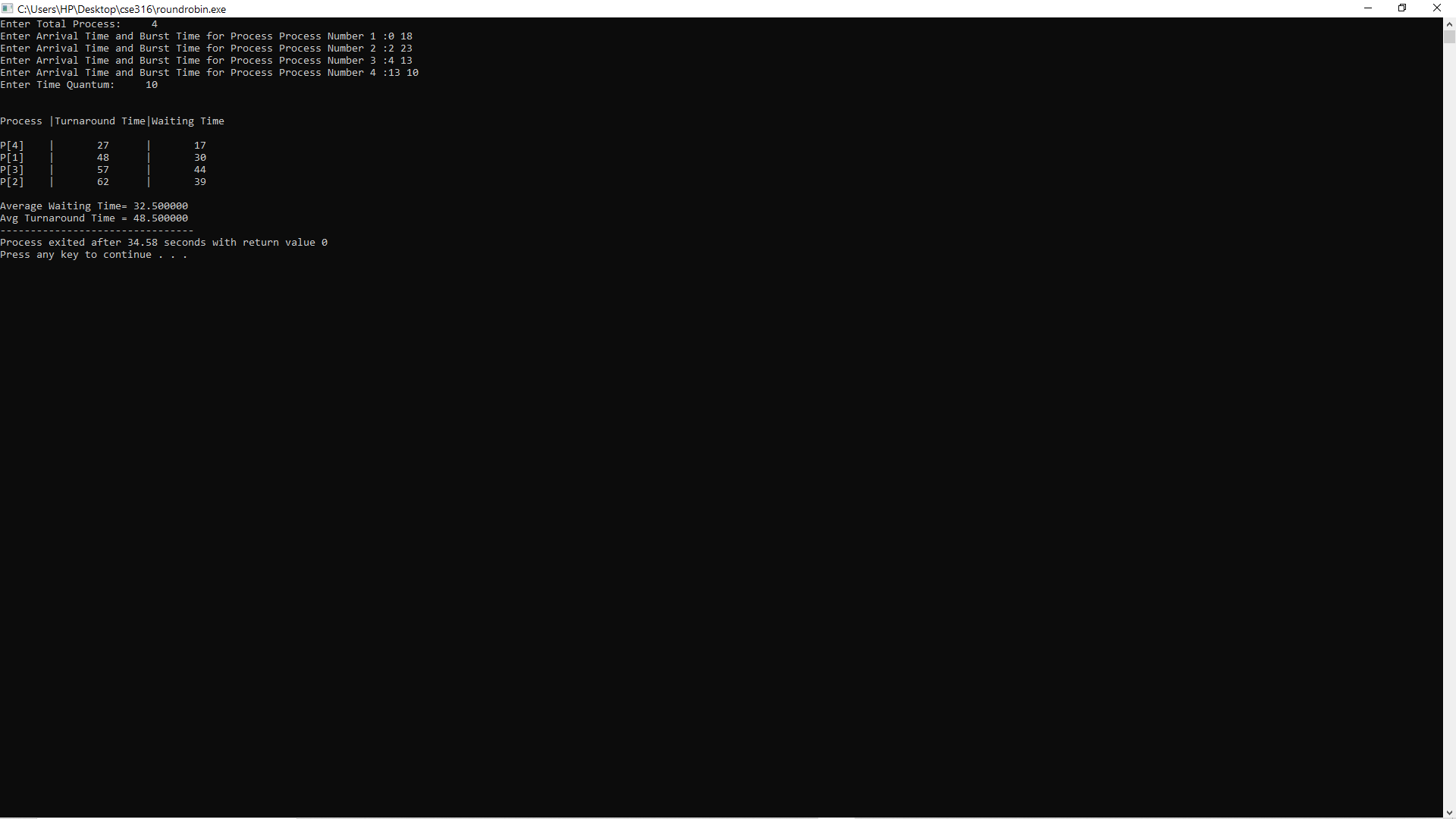
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Processes | Arrival time | Burst time | Turnaround time | Waiting time |
| P1 | 0 | 18 | 48 | 30 |
| P2 | 2 | 23 | 62 | 39 |
| P3 | 4 | 13 | 57 | 44 |
| P4 | 13 | 10 | 27 | 17 |

Avg waiting time=23.75

Avg turnaround time=37.00

**AFTER EXECUTION**

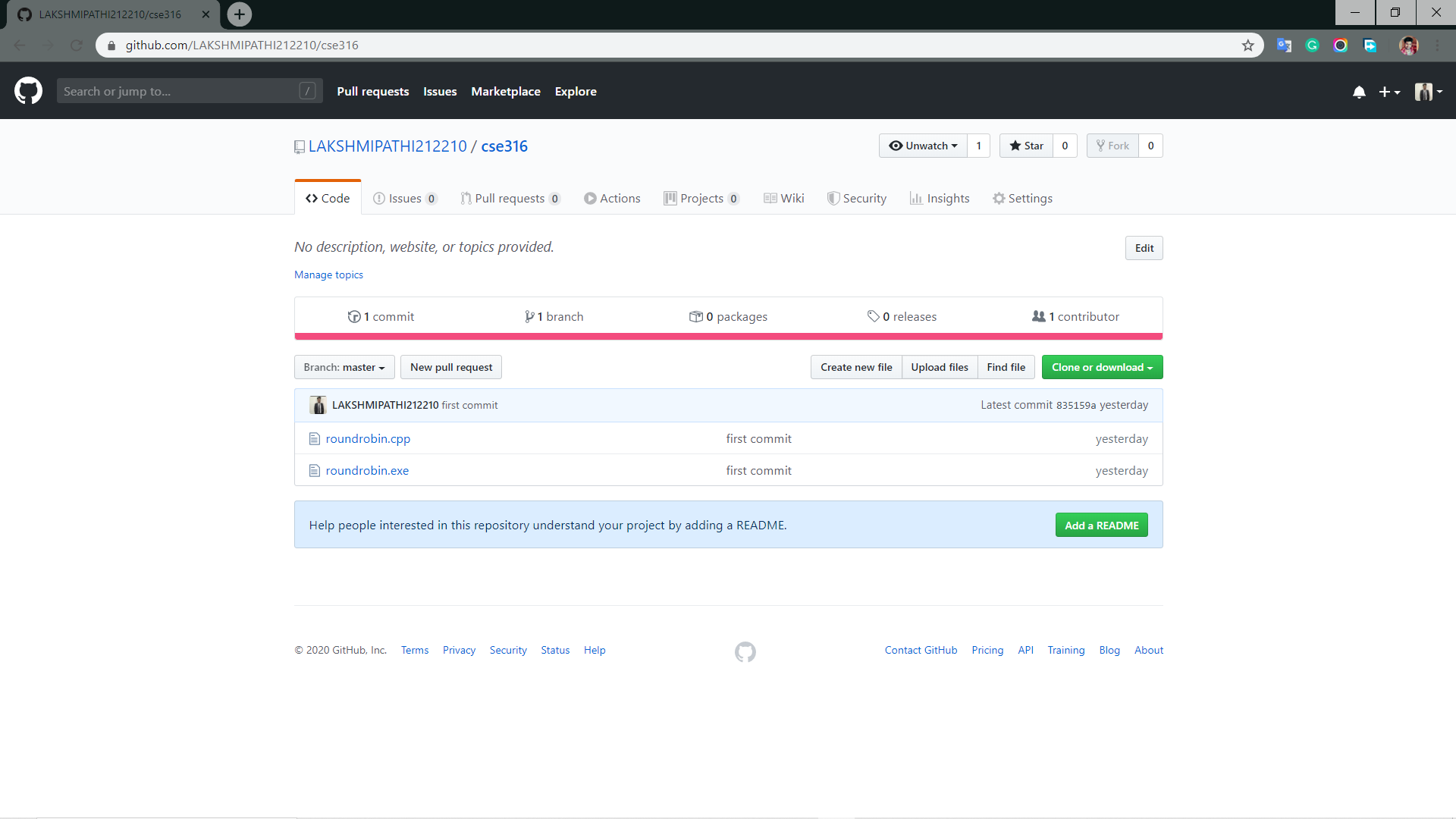
The output is:-



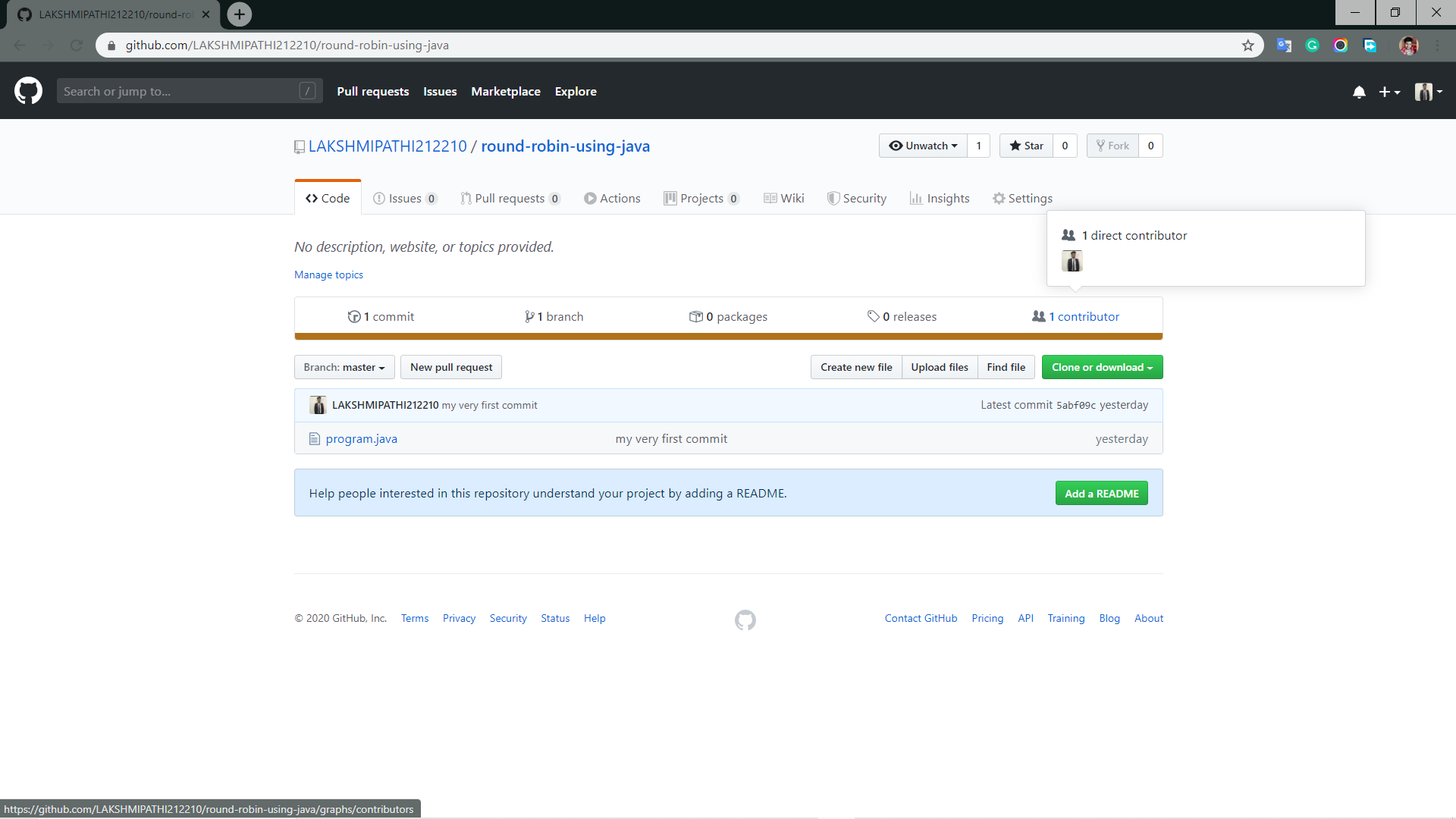
**GITHUB REPOSITORY**

**I have submitted five repositories in github.**

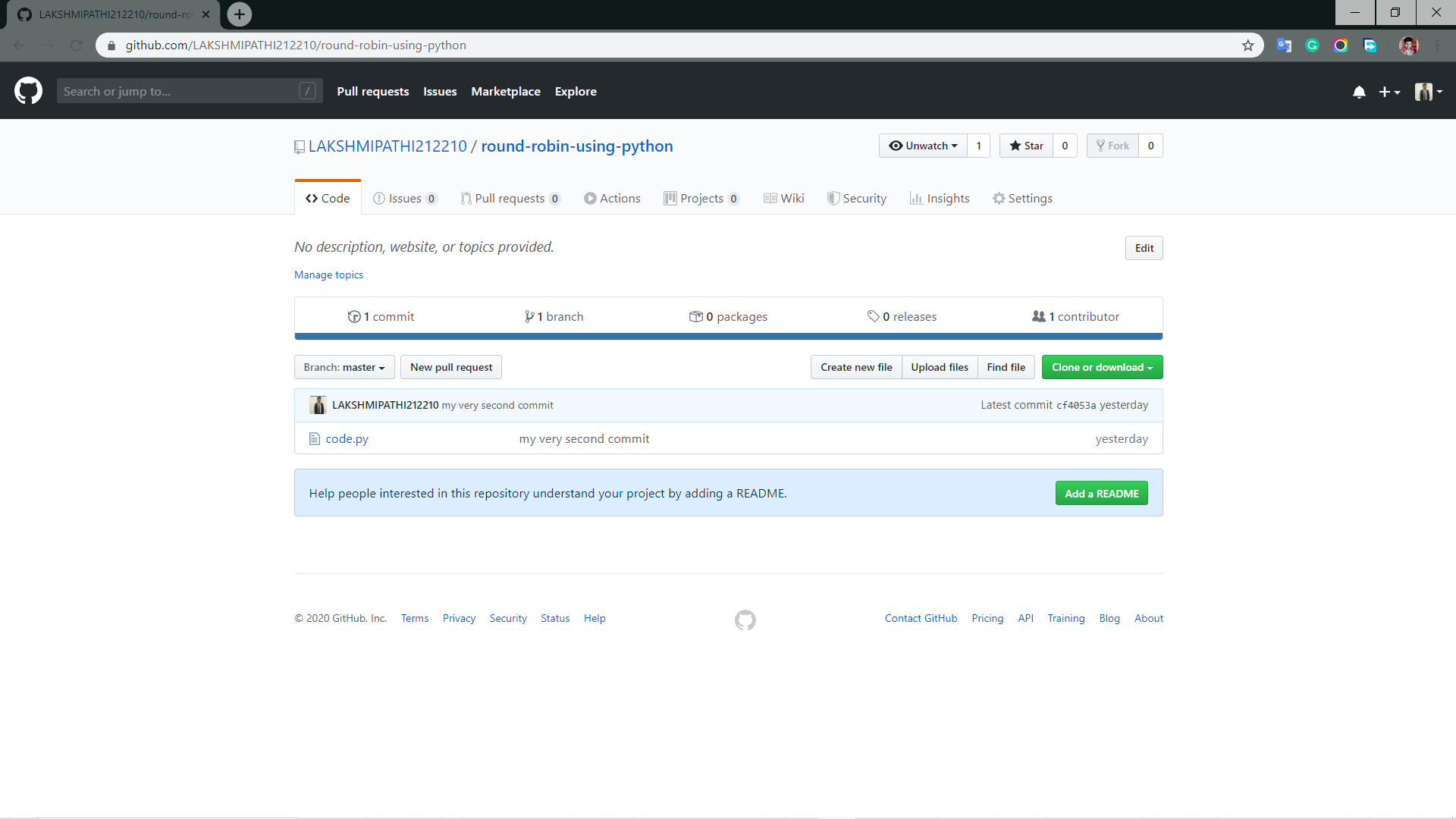
1.First repository



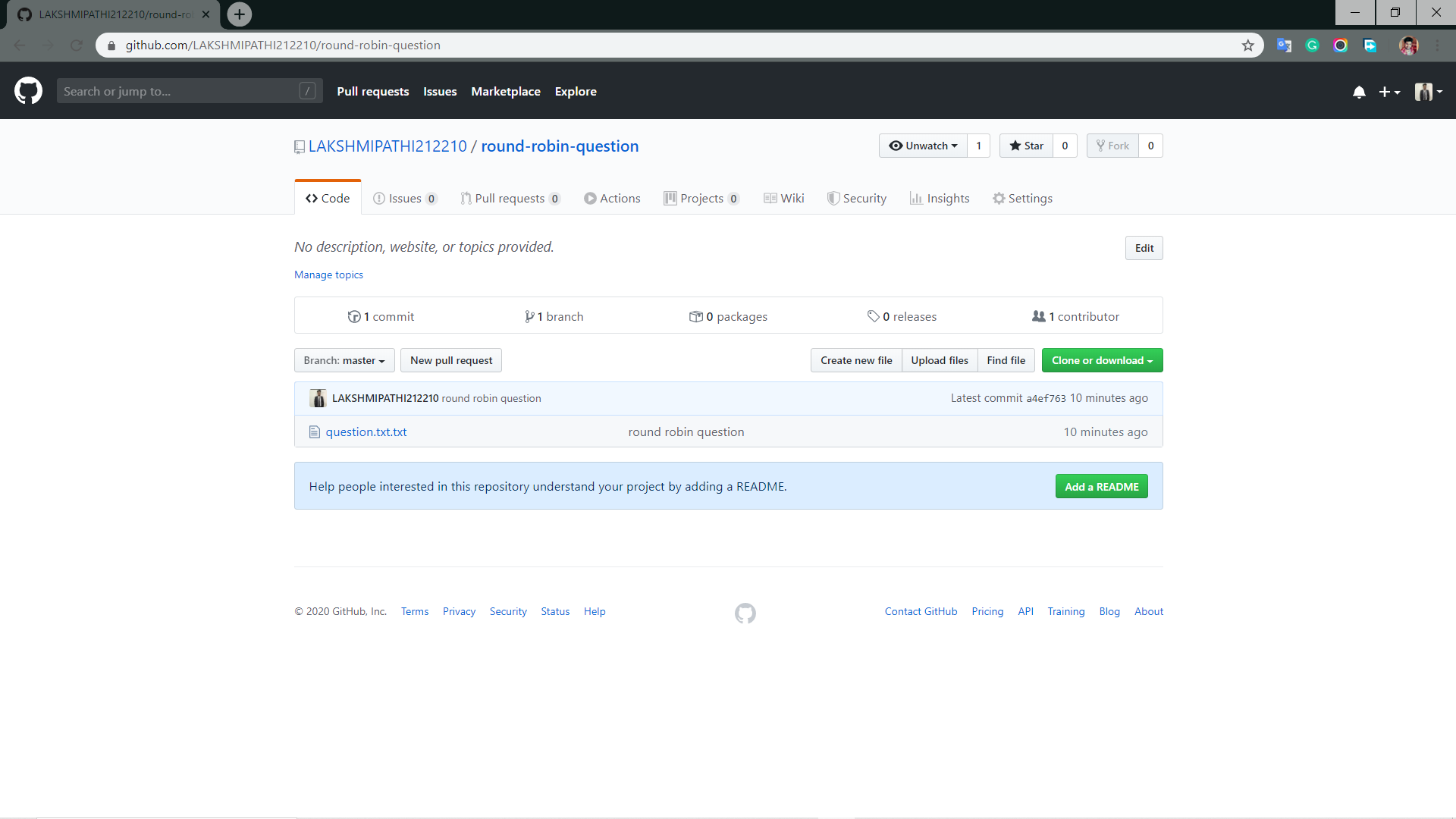
2.Second repository



3.Third repository



4.Fourth repository



5.Fifth repository

**CONCLUSION**

A lot of attempts were developed to find a solution for the high turnaround time, high waiting time and the overhead of extra context switches in Simple Round Robin Algorithm, regardless of the different methodologies used in these attempts; however all of them rely based on the fixed-time quantum. The proposed algorithm called Performance analysis of Round Robin Scheduling using adaptive approach based on smart time slice is designed to solve the drawbacks of Simple Round Robin Scheduling Algorithm in a practical, simple and applicable manner. The above comparisons show that the proposed algorithm provides much better results twice or three times and in some cases perhaps more than other approaches based on fixed time quantum in all scheduling criteria. After mathematical analysis of this algorithm showed through a simulation program which is prepared for this purpose that this algorithm works in a stable manner regardless of the number of the new running processes, taking into consideration the terminated and the new arrival processes. It is recommended to use the shortest burst time concept with smart time slice; because it will give the operating system the ability to adapt to the user behavior and not vice versa, which may lead us to rethink building an intelligent, learnable and adaptable operating system. The Adaptive Round Robin Algorithm is designed to meet all scheduling criteria such as maximum CPU utilization, minimum average waiting time, minimum average turnaround time and fewer contexts switches. According to the result of this algorithm all the CPU scheduling criteria is perfect found to the comparison of Simple Round Robin Scheduling Algorithm. The Adaptive RR algorithm uses fixed time quantum for computing the new.