Table of Contents

Introduction

Round Robin CPU Scheduling Algorithm Python Implementation

Round Robin is a CPU scheduling algorithm where each process is assigned a fixed time slot in a cyclic way. Round Robin can be described as a preemptive process scheduling algorithm where each process is given a fix amount of time to execute. This is called a quantum. Once a process is executed for a given time period, it is preempted and other process executes for a given time period.

Tasks/Model/Convention

Task

The task is, given some process along with their arrival times and burst times we are to schedule these processes using the Round Robin CPU Scheduling algorithm and ensure each process gets executed.

Model and Convention

As a team we went with the agile model of development because it's a model that adapts to rapid changes that occur throughout the development process. Team members were tasked with components to work on and deliver so as to speed up the development process. Test cases were taken and the algorithm was used to schedule. Once understanding of this was accomplished python was used to implement this Scheduling and the same cases were tested again.

Round Robin

The program is divided into sections where each section has set of function/models responsible for various activities in the program.

Driver Program

MainProgram Is the driver program of this python implantation of Round Robin Program.

Core Part

Algorithm – This is the heart of the program. It has Round Robin Scheduling implementation.

Process – This is the model which has information such as arrival time, burst time, order and list of running passes.

CPU - It is a model of CPUs and it has list of running slots.

Idle - Simple Model for house keeping the idle of cpu.

KQueue - Our own Queue implementation.

Input/output

Input – This has functions to read input from user and read input from test cases.

Assert Input – This checks inputs and validates to make sure they are correct.

OutputOptions – This is the utility that controls the output options.

Calculation Code

CalculateTime – This will go through the running slots, calculate total time, total wait time, average time and total average time.

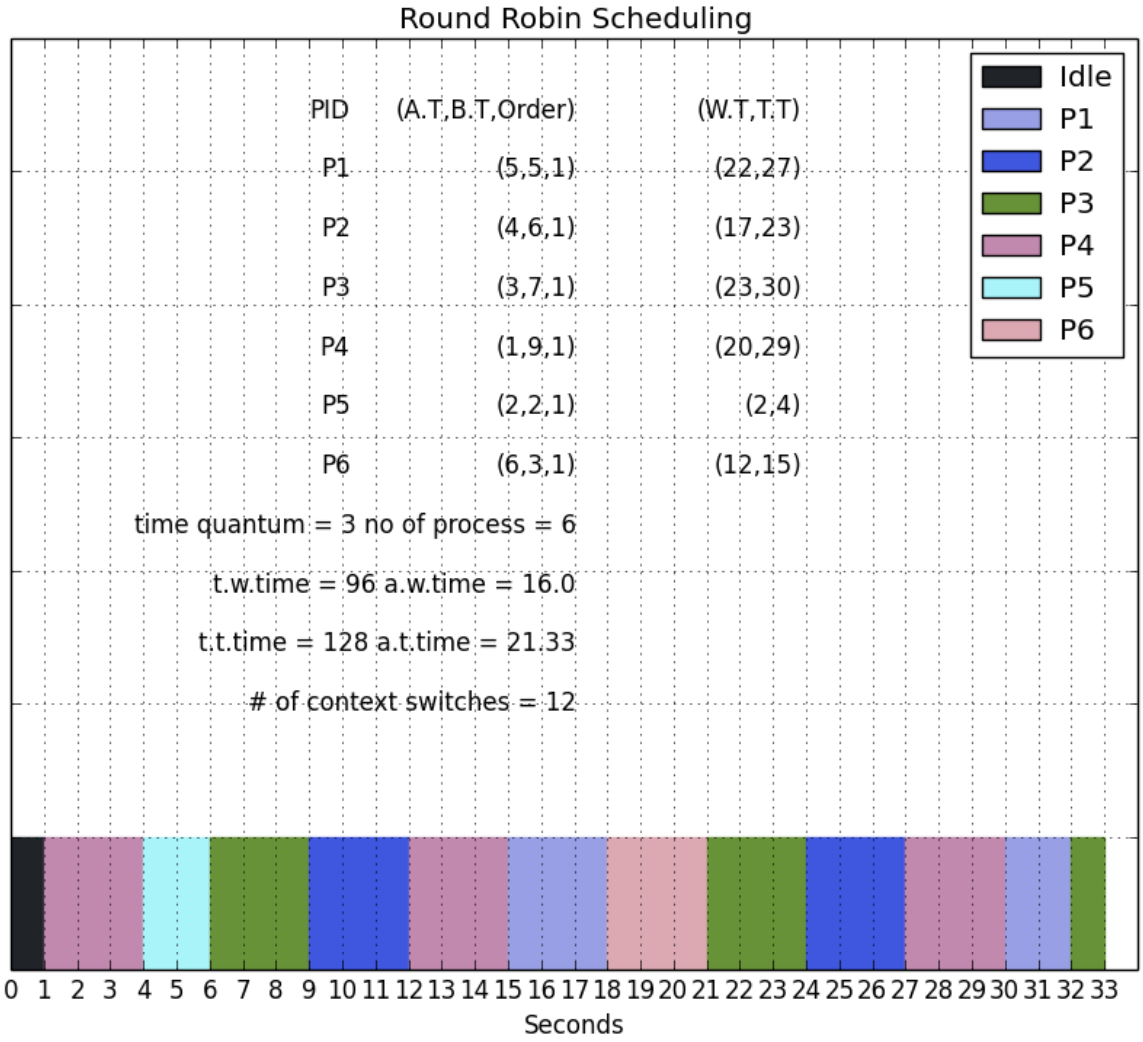
CalculateContextSwitch – This will go through the running slots and count the no of context switches happen on the program

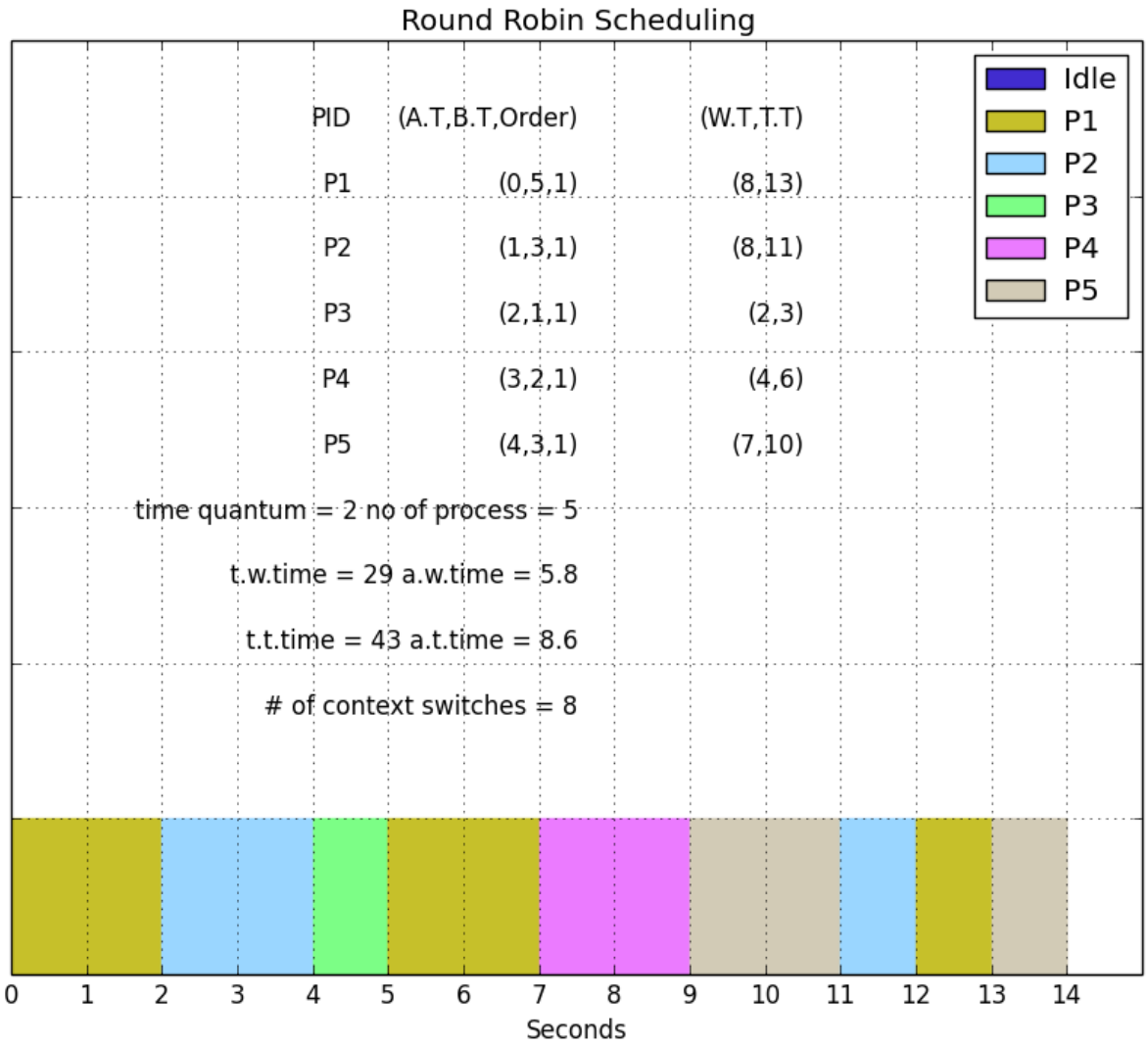
Chart Package and Display

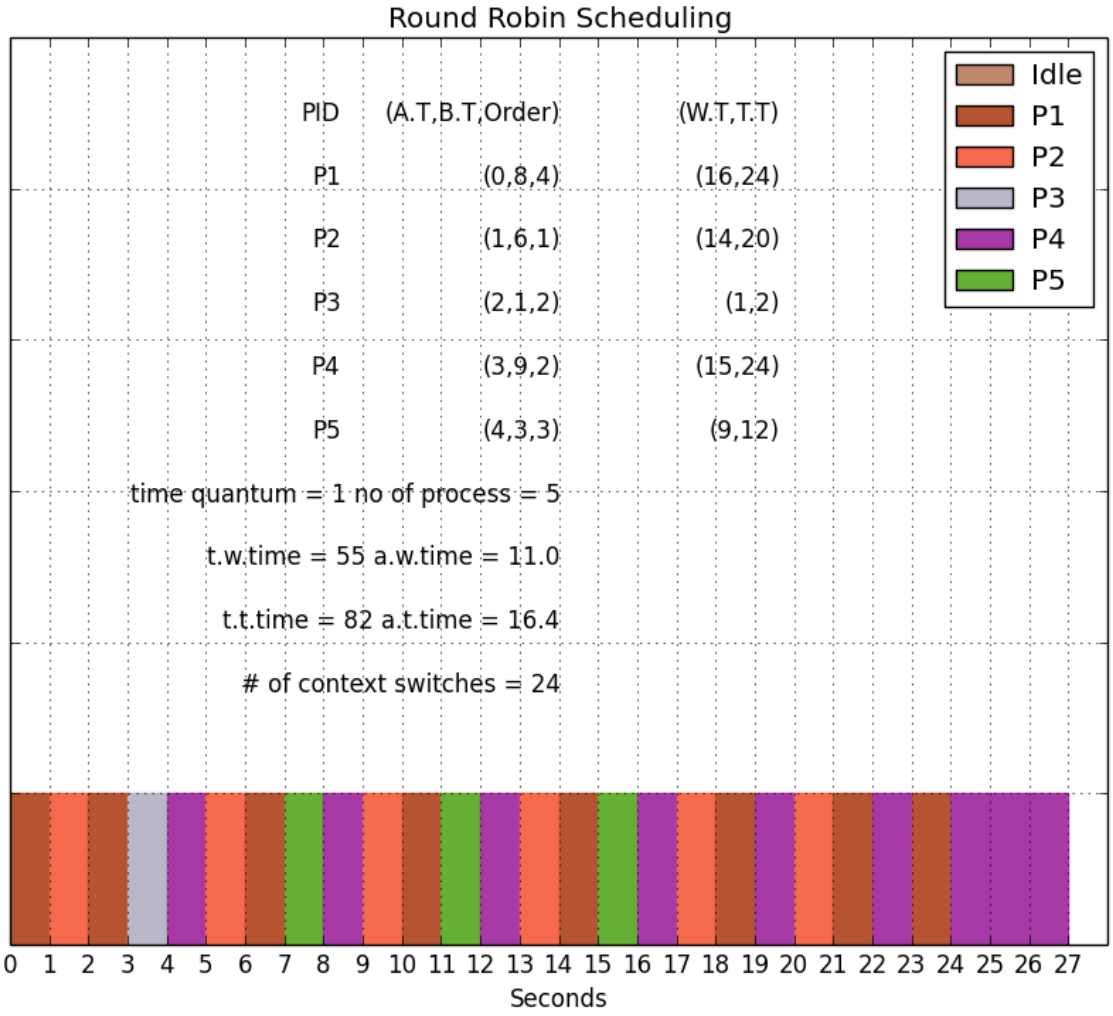
GanttChart - Chart Module to draw the gantt chart

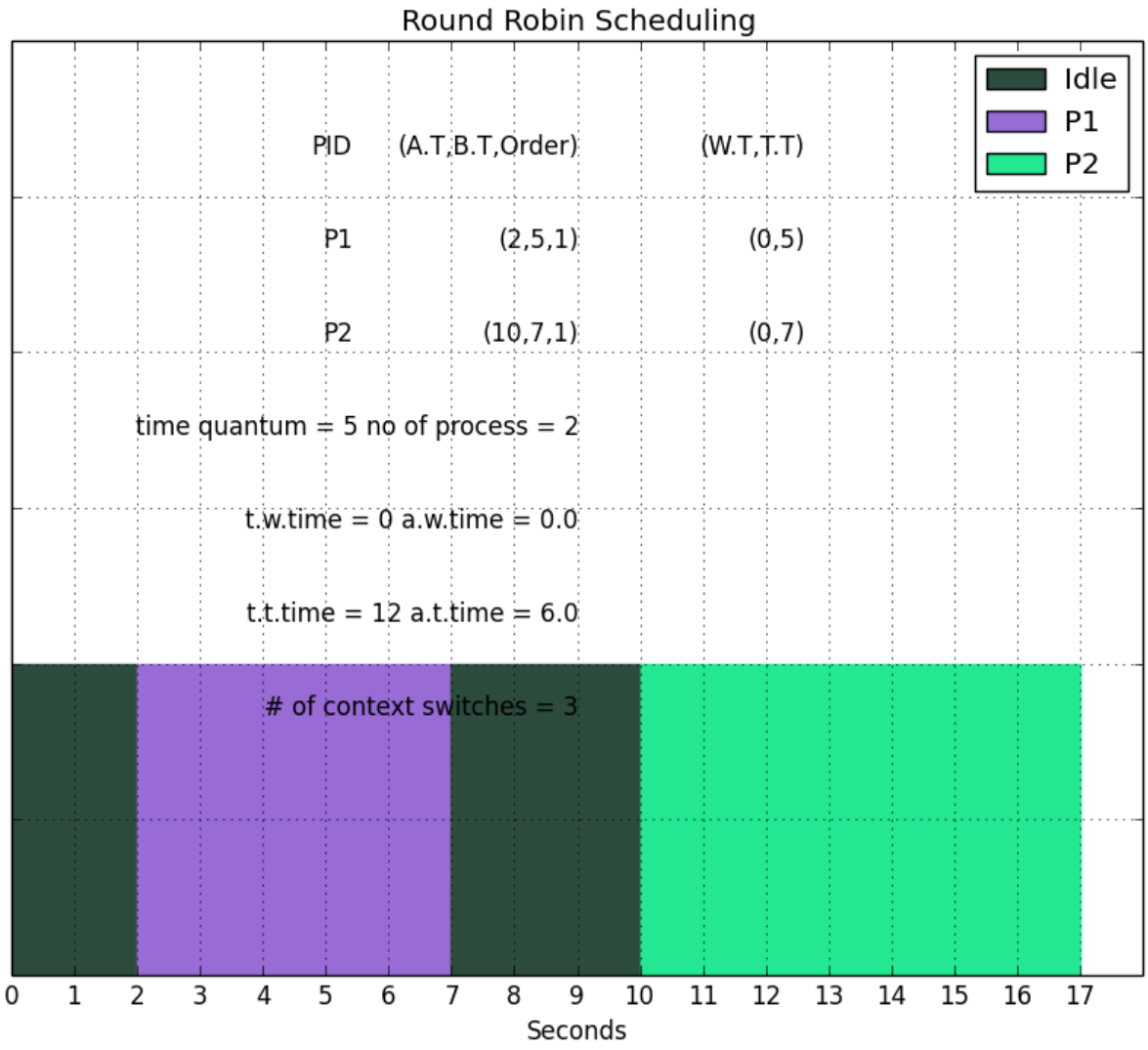
Inputs Used and Corresponding Outputs

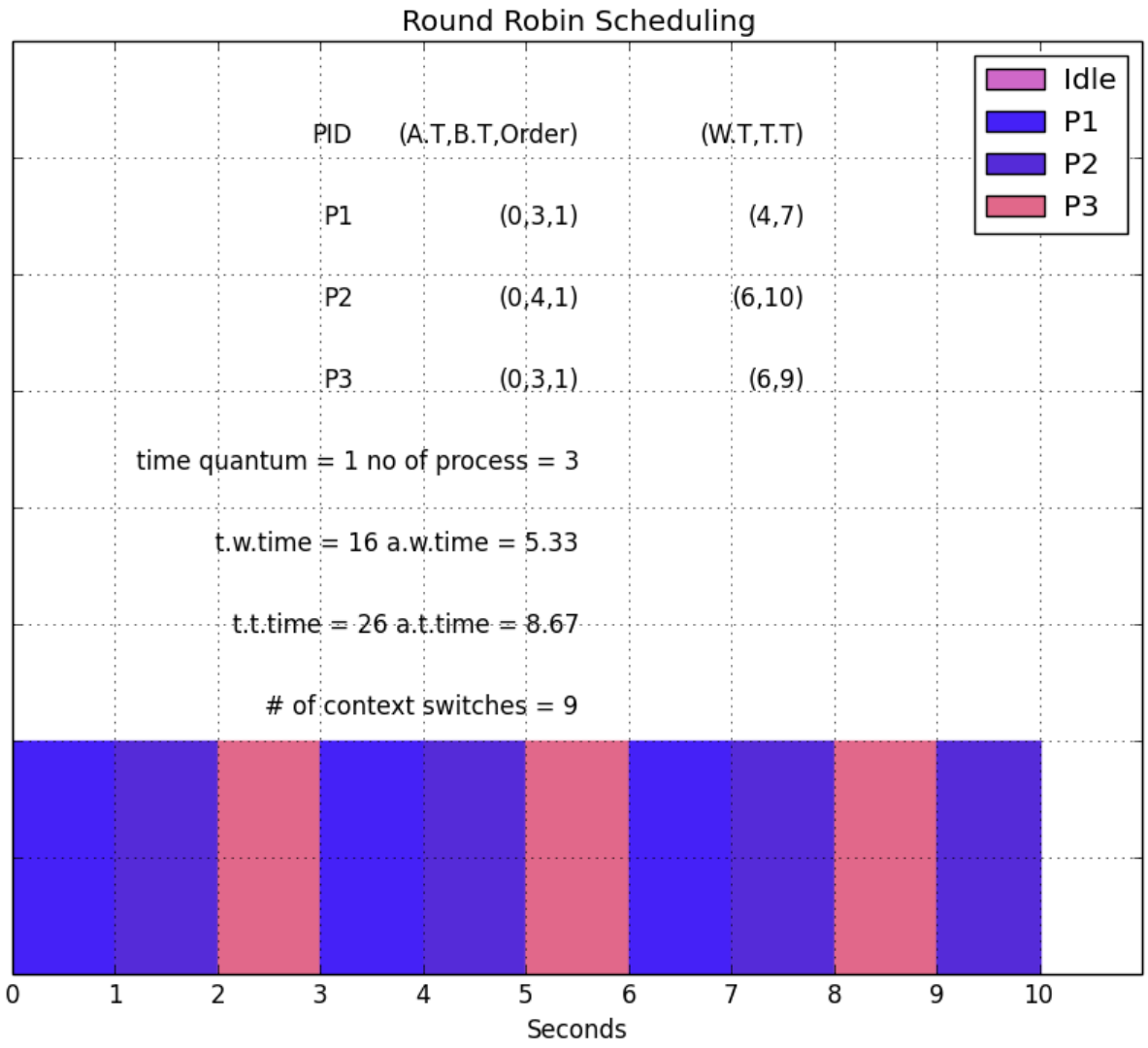
Below are screenshots of the test cases we run and their corresponding outputs

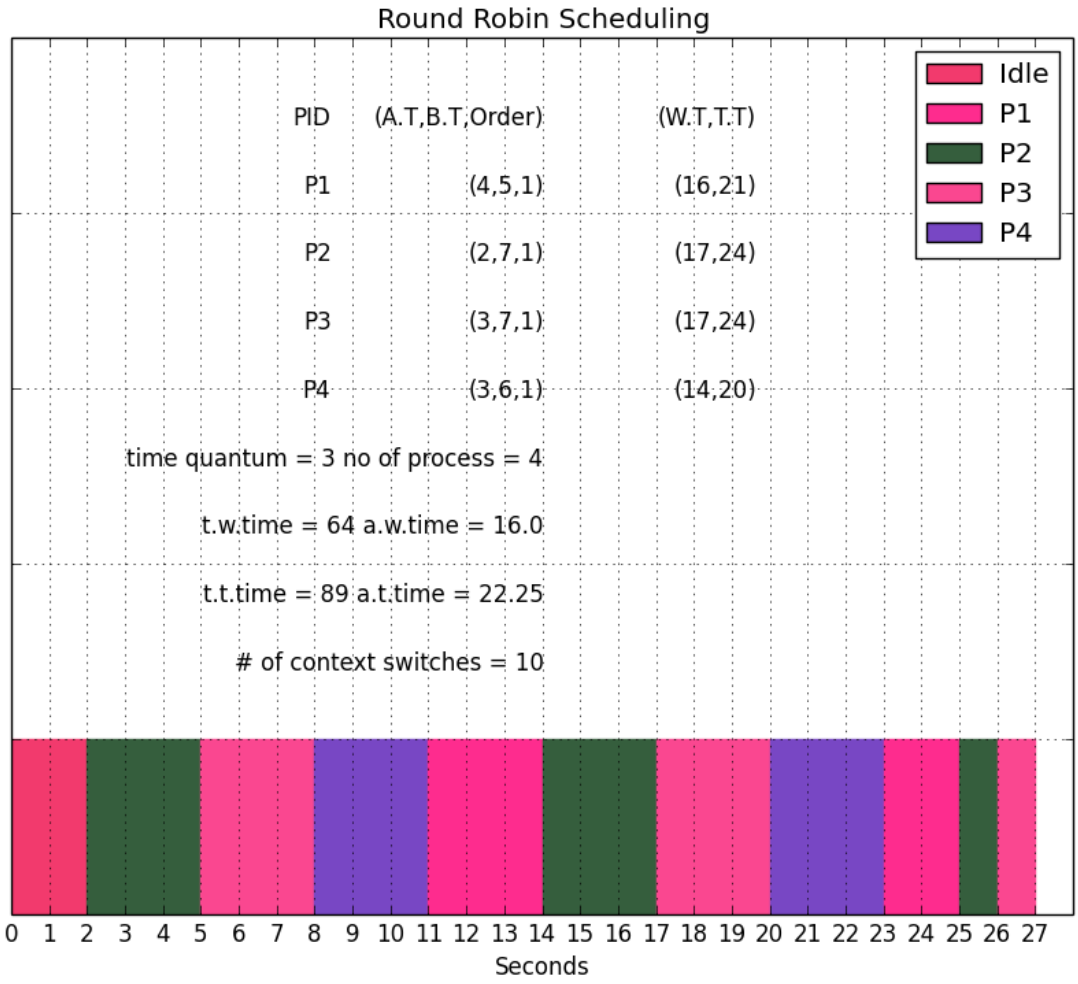

Round Robin Scheduling
PID (A.T,B.T,Order) (W.T,T.T)
P1 (5,5,1) (22,27)
P2 (4,6,1) (17,23)
P3 (3,7,1) (23,30)
P4 (1,9,1) (20,29)
P5 (2,2,1) (2,4)
P6 (6,3,1) (12,15)
time quantum = 3 no of process = 6
t.w.time = 96 a.w.time = 16.0
t.t.time = 128 a.t.time = 21.33
# of context switches = 12
Idle
P1
P2
P3
P4
P5
P6
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
Seconds

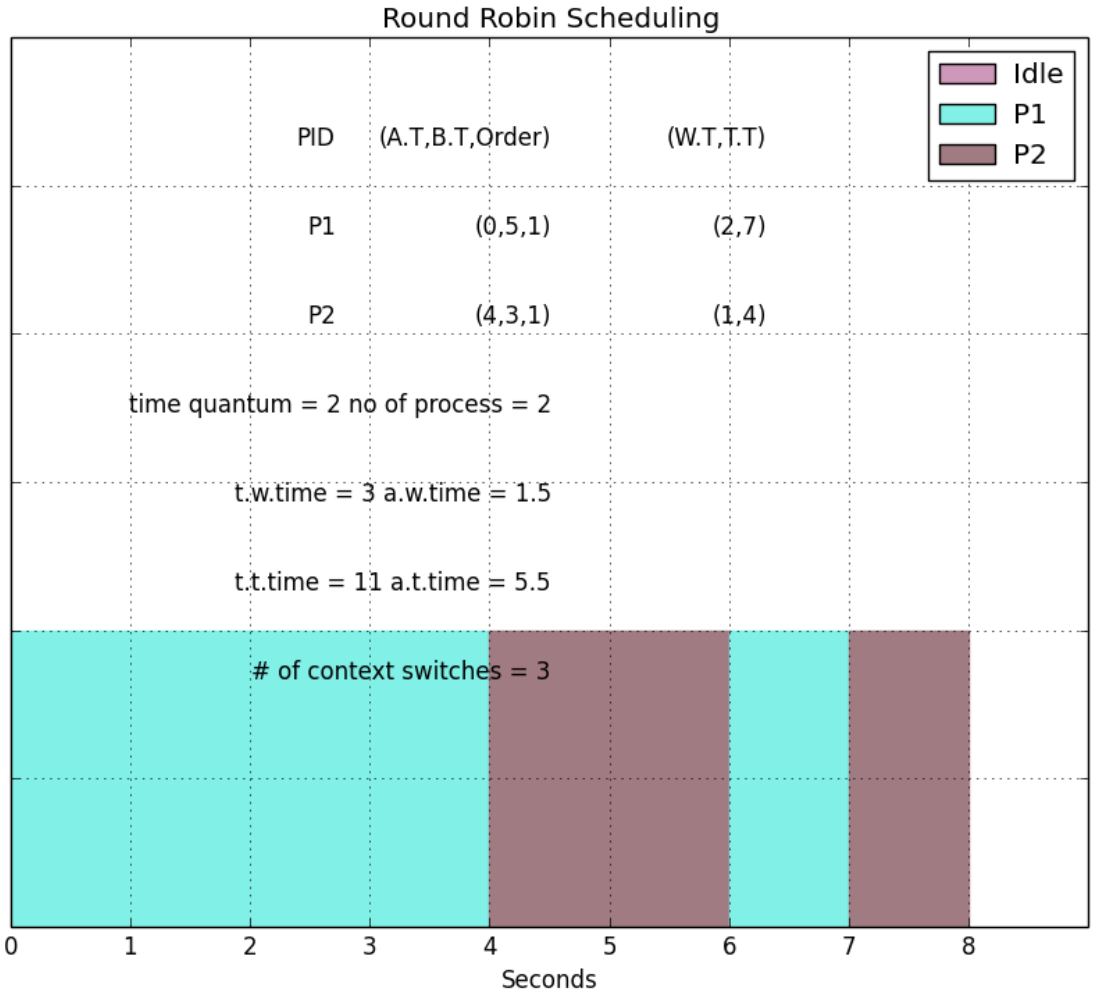

Round Robin Scheduling
PID (A.T,B.T,Order) (W.T,T.T)
P1 (0,5,1) (8,13)
P2 (1,3,1) (8,11)
P3 (2,1,1) (2,3)
P4 (3,2,1) (4,6)
P5 (4,3,1) (7,10)
time quantum = 2 no of process = 5
t.w.time = 29 a.w.time = 5.8
t.t.time = 43 a.t.time = 8.6
# of context switches = 8
Idle
P1
P2
P3
P4
P5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
Seconds


Round Robin Scheduling
PID (A.T,B.T,Order) (W.T,T.T)
P1 (0,8,4) (16,24)
P2 (1,6,1) (14,20)
P3 (2,1,2) (1,2)
P4 (3,9,2) (15,24)
P5 (4,3,3) (9,12)
time quantum = 1 no of process = 5
t.w.time = 55 a.w.time = 11.0
t.t.time = 82 a.t.time = 16.4
# of context switches = 24
Idle
P1
P2
P3
P4
P5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
Seconds


Round Robin Scheduling
Idle
P1
P2
PID (A.T,B.T,Order) (W.T,T.T)
P1 (2,5,1) (0,5)
P2 (10,7,1) (0,7)
time quantum = 5 no of process = 2
t.w.time = 0 a.w.time = 0.0
t.t.time = 12 a.t.time = 6.0
# of context switches = 3
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Seconds


Round Robin Scheduling
Idle
P1
P2
P3
PID (A.T,B.T,Order) (W.T,T.T)
P1 (0,3,1) (4,7)
P2 (0,4,1) (6,10)
P3 (0,3,1) (6,9)
time quantum = 1 no of process = 3
t.w.time = 16 a.w.time = 5.33
t.t.time = 26 a.t.time = 8.67
# of context switches = 9
0 1 2 3 4 5 6 7 8 9 10
Seconds


Round Robin Scheduling
PID (A.T,B.T,Order) (W.T,T.T)
P1 (4,5,1) (16,21)
P2 (2,7,1) (17,24)
P3 (3,7,1) (17,24)
P4 (3,6,1) (14,20)
time quantum = 3 no of process = 4
t.w.time = 64 a.w.time = 16.0
t.t.time = 89 a.t.time = 22.25
# of context switches = 10
Idle
P1
P2
P3
P4
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
Seconds



Conclusion

As a group working on the implementation of this process scheduling algorithm has given us better understanding on preemptive scheduling and how they work to ensure efficient use of the CPU. Using python for this implementation also helped us to understand the python programming concepts better, knowing what to use for what and how to go about certain programming concepts in python. We learnt to efficiently validate our inputs and assert our outputs using python. We also learnt to use chart modules together with math.lib to depict graphs.