# EXPERIMENT NO-4

**Aim: -** To implement Round Robin - RR process scheduling algorithm

**Lab objective:** Describe Process & process management using CPU scheduling Algorithm

**Theory:**

The name of this algorithm comes from the round-robin principle, where each person gets an equal share of something in turn. This is the preemptive version of first come first serve scheduling. The Algorithm focuses on Time Sharing. In this algorithm, every process gets executed in a cyclic way. A certain time slice is defined in the system which is called time quantum. Each process present in the ready queue is assigned the CPU for that time quantum, if the execution of the process is completed during that time then the process will terminate else the process will go back to the ready queue and waits for the next turn to complete the execution. All the jobs get a fare allocation of CPU. The higher the time quantum, the higher the response time in the system. The lower the time quantum, the higher the context switching overhead in the system

**Code:**

def findWaitingTime(processes, n, bt, wt, quantum):

    rem\_bt = [0] \* n

    for i in range(n):

        rem\_bt[i] = bt[i]

    t = 0

    while(1):

        done = True

        for i in range(n):

            if (rem\_bt[i] > 0) :

                done = False

                if (rem\_bt[i] > quantum) :

                    t += quantum

                    rem\_bt[i] -= quantum

                else:

                    t = t + rem\_bt[i]

                    wt[i] = t - bt[i]

                    rem\_bt[i] = 0

        if (done == True):

            break

def findTurnAroundTime(processes, n, bt, wt, tat):

    for i in range(n):

        tat[i] = bt[i] + wt[i]

def findavgTime(processes, n, bt, quantum):

    wt = [0] \* n

    tat = [0] \* n

    findWaitingTime(processes, n, bt,

                        wt, quantum)

    findTurnAroundTime(processes, n, bt,

                                wt, tat)

    print("Processes Burst Time  Waiting",

                    "Time Turn-Around Time")

    total\_wt = 0

    total\_tat = 0

    for i in range(n):

        total\_wt = total\_wt + wt[i]

        total\_tat = total\_tat + tat[i]

        print(" ", i + 1, "\t\t", bt[i],

            "\t\t", wt[i], "\t\t", tat[i])

    print("\nAverage waiting time = %.5f "%(total\_wt /n) )

    print("Average turn around time = %.5f "% (total\_tat / n))

proc = [1, 2, 3]

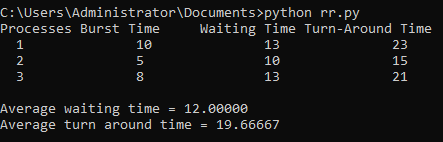
n = 3

burst\_time = [10, 5, 8]

quantum = 2;

findavgTime(proc, n, burst\_time, quantum)

**Output:**



**Lab Outcome:**

CPU Scheduling is a process of determining which process will own CPU for execution while another process is on hold. The successful implementation of RR scheduling algorithm helps to understand that every process gets executed in a cyclic way.

**Conclusion:**

* It can be actually implementable in the system because it is not depending on the burst time.
* A certain time slice is defined in the system which is called time quantum.
* if the execution of the process is completed during that time then the process will terminate else the process will go back to the ready queue.