**Round-robin CPU scheduling**

A popular preemptive scheduling technique in computer operating systems is round-robin (RR) CPU scheduling. This algorithm assigns a set time slot or quantum to each process, and the CPU rotates circularly between them. A process is pushed to the back of the queue and gives the next in line the opportunity to run when its time quantum expires.

The round-robin scheduling mechanism distributes the CPU's time evenly among the processes, ensuring fairness. It is especially helpful when there are similar priority levels across the processes since it keeps one process from taking up all of the CPU time for a long time. That might not, however, be the best option where processes have significantly different execution time requirements, as it may lead to unnecessary context switches.

**Advantages of round-robin CPU scheduling:**

* Fairness.
* Simple implementation.

**Disadvantages of round-robin CPU scheduling:**

* Inefficient for certain workload.
* Low turnaround time for certain process.

Here is a pseudo-code for the Round-Robin Algorithm

function findWaitingTime(processes, n, burst time, wt, quantum):

rem\_bt = copy(burst\_time)

t = 0

while true:

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 += rem\_bt[i]

wt[i] = t - burst\_time[i]

rem\_bt[i] = 0

if done:

break

// Main program

processes = [1, 2, 3]

n = size of processes

burst\_time = [10, 5, 8]

quantum = 2

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

function findTurnAroundTime(processes, n, burst\_time, wt, tat):

for i in range(n):

tat[i] = burst\_time[i] + wt[i]

function findavgTime(processes, n, burst\_time, quantum):

wt = new array of size n

tat = new array of size n

total\_wt = 0

total\_tat = 0

findWaitingTime(processes, n, burst\_time, wt, quantum)

findTurnAroundTime(processes, n, burst\_time, wt, tat)

print("Processes\tBurst time\tWaiting time\tTurnaround time")

for i in range(n):

total\_wt += wt[i]

total\_tat += tat[i]

print(processes[i], "\t\t", burst\_time[i], "\t\t", wt[i], "\t\t", tat[i])

avg\_wt = total\_wt / n

avg\_tat = total\_tat / n

print("Average waiting time =", avg\_wt)

print("Average turnaround time =", avg\_tat)

|  |  |
| --- | --- |
| **Algorithms** | **Results** |
| **Turn aroundtime** | 19.6667 |
| **Average waiting time** | 12 |

**Explanation:**

Turnaround Time: This is the total time a process takes from arrival to completion. On average, each process in the given scenario takes approximately 19.67 units of time to finish its execution.

Average Waiting Time: This is the average time a process spends waiting in the ready queue before it gets CPU time. On average, each process waits for about 12 units of time before it can start running on the CPU.