| S.No: 2 | Exp. Name: ***Implement CPU Scheduling Algorithms*** | Date: 2022-05-06 |
|---|---|---|



**Aim:**

Implementation of the Round Robin cpu scheduling algorithm
(https://gecgudlavalleru.codetantra.com/secure/labs-q.jsp?sNo=4&qId=5bec179564bac110545ba035&bd=AY3RFZHVEQg%3D%3D&lid=5db6d168a183970b79e5cd34&labbd=AMzM2X2N0X2No&expTitle=Implementation%20of%20the%20Round%

**Source Code:**

os4.c

```
#include<stdio.h>
int main()
{
   int i, limit, total =0,x,counter =0,time_quantum;
   int wait_time =0,turnaround_time =0,arrival_time[10],burst_time[10],temp[10];
   float average_wait_time,average_turnaround_time;
   printf("Enter Total Number of Processes: ");
   scanf("%d",&limit);
   x =limit;
   for(i=0;i<limit;i++)
   {
      printf("Enter Details of Process[%d]:",i+1);
      printf(" Arrival Time:\t");
      scanf("%d",&arrival_time[i]);
      printf("Burst Time:\t");
      scanf("%d",&burst_time[i]);
      temp[i]=burst_time[i];
   }
   printf("Enter Time Quantum:\t");
   scanf("%d",&time_quantum);
   printf("Process ID\t\tBurst Time\t Turnaround Time\t Waiting Time");
   for(total = 0,i = 0; x != 0;)
   {
      if(temp[i] <= time_quantum && temp[i]>0)
      {
         total = total + temp[i];
         temp[i] = 0;
         counter = 1;
      }
      else if(temp[i]>0)
      {
         temp[i]= temp[i]-time_quantum;
         total =total+ time_quantum;
      }
      if(temp[i]==0 && counter ==1)
      {
         x--;
         printf("\nProcess[%d]\t\t%d\t\t %d\t\t\t %d", i + 1, burst_time[i], total - ar
rival_time[i], total - arrival_time[i] - burst_time[i]);
         wait_time = wait_time + total - arrival_time[i] - burst_time[i];
         turnaround_time = turnaround_time + total - arrival_time[i];
         counter=0;
         }
         if(i==limit-1)
         {
            i=0;
         }
         else if(arrival_time[i+1]<=total)
         {
            i++;
         }
         else
         {
            i=0;
         }
   }
   average_wait_time=wait_time*1.0/limit;
   average_turnaround_time=turnaround_time*1.0/limit;
   printf("\nAverage Waiting Time:\t%f",average_wait_time);
   printf("\nAvg Turnaround Time:\t%f\n",average_turnaround_time);
   return 0;
}
```

Execution Results - All test cases have succeeded!

| Test Case - 1 |
|---|
| **User Output** |
| Enter Total Number of Processes:  3 |
| Enter Details of Process[1]: Arrival Time:  0 |

**Test Case - 1**

```
Burst Time:      3
Enter Details of Process[2]: Arrival Time:       0
Burst Time:      2
Enter Details of Process[3]: Arrival Time:       1
Burst Time:      3
Enter Time Quantum:      5
```

| Process ID | Burst Time | Turnaround Time | Waiting Time |
|---|---|---|---|
| Process[1] | 3 | 3 | 0 |
| Process[2] | 2 | 5 | 3 |
| Process[3] | 3 | 7 | 4 |

```
Average Waiting Time:    2.333333
Avg Turnaround Time:     5.000000
```