**Experiment No. 09**

**Experiment Name:** Implementation of Round Robin Scheduling algorithm.

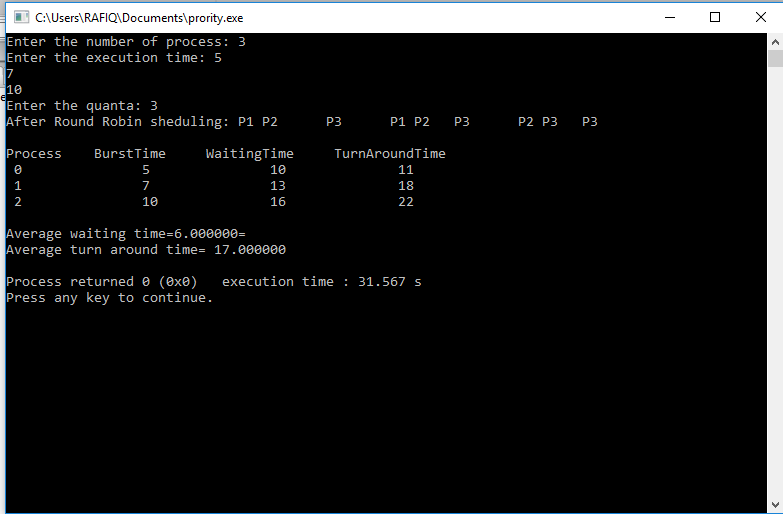
**Aim and Objectives:**

* What is Round Robin Scheduling algorithm.
* How to implementation this algorithm.

**Source Code:**

|  |
| --- |
| #include<stdio.h>  int main()  {  int n,i,k,x=0,s=0,r=0,q=0,a[30],e[30],t[30];  float m,p=0;  printf("Enter the number of process: ");  scanf("%d",&n);  printf("Enter the execution time: ");  for(i=0; i<n; i++)  {  scanf("%d",&a[i]);  e[i]=a[i];  }  printf("Enter the quanta: ");  scanf("%d",&q);  printf("After Round Robin sheduling: ");  for(i=0; i<n; i++)  {  if(x<a[i])  {  x=a[i];  }  }  k=x/q;  while(s<=k)  {  for(i=0; i<n; i++)  {  if(a[i]>0)  {  if(a[i]>q)  {  r=r+q;  a[i]=a[i]-q;  printf("P%d\t",i+1);  }else  {  r=r+a[i];  a[i]=a[i]-q;  printf("P%d ",i+1);  t[i]=r;  }  }  }  s++;  }  printf("\n\nProcess BurstTime WaitingTime TurnAroundTime\n");  for(i=0; i<n; i++)  {  printf(" %d \t\t %d\t\t %d\t\t %d\t\t \n",i,e[i],x,t[i]);  x=x+q;  }  m=x/n;  printf("\nAverage waiting time=%f= ",m);  printf("\nAverage turn around time= ");  for(i=0; i<n; i++)  p=p+t[i];  p=p/n;  printf("%f",p);  printf("\n");  return 0;  } |

**Output:**



**Conclusion:** Round robin is the most widely used process scheduling algorithm. The basic strategy for round robin scheduling is that if there are n process, each of the process will receive 1/n CPU Execution Time. Each process is allotted a time quantum, for which it is executed. The incoming processes are kept in a ready list while another one is executing. If the time quanta allotted for a process is over, then that process is moved to ready and the next process in the ready list is executed for the allotted time quanta