**Practical : 6**

**Aim: To implement RR (Round Robin) CPU Scheduling Algorithm.**

**Program:**

**#include <stdio.h>**

**int main()**

**{**

**int i,x=-1,k[10],m=0,n,t,s=0;**

**int a[50],temp,b[50],p[10],bur[10],bur1[10],arrtime[10];**

**int wat[10],tur[10],ttur=0,twat=0,j=0,cpu\_util;**

**float awat,atur,throughput=0,totaltime=0;**

**printf("Enter no. of process : ");**

**scanf("%d", &n);**

**for(i=0; i<n; i++)**

**{**

**printf("Burst time for process P%d : ", (i+1));**

**scanf("%d", &bur[i]);**

**bur1[i] = bur[i];**

**}**

**for(i=0; i<n; i++)**

**{**

**printf("Enter arrival time for process P%d : ", (i+1));**

**scanf("%d", &arrtime[i]);**

**}**

**printf("Enter the time slice (in ms) : ");**

**scanf("%d", &t);**

**for(i=0; i<n; i++)**

**{**

**b[i] = bur[i] / t;**

**if((bur[i]%t) != 0)**

**b[i] += 1;**

**m += b[i];**

**}**

**printf("\n............Round Robin Scheduling............\n");**

**printf("\nGANTT Chart\n");**

**for(i=0; i<m; i++)**

**{**

**printf("--------");**

**}**

**printf("\n");**

**a[0] = 0;**

**printf("|");**

**while(j < m)**

**{**

**if(x == n-1)**

**{ x = 0; }**

**else**

**x++;**

**if(bur[x] >= t)**

**{**

**bur[x] -= t;**

**a[j+1] = a[j] + t;**

**if(b[x] == 1)**

**{**

**p[s] = x;**

**k[s] = a[j+1];**

**s++;**

**}**

**j++;**

**b[x] -= 1;**

**printf(" P%d\t|", x+1);**

**}**

**else if(bur[x] != 0)**

**{**

**a[j+1] = a[j] + bur[x];**

**bur[x] = 0;**

**if(b[x] == 1)**

**{**

**p[s] = x;**

**k[s] = a[j+1];**

**s++;**

**}**

**j++;**

**b[x] -= 1;**

**printf(" P%d\t|",x+1);**

**}**

**}**

**printf("\n");**

**for(i=0;i<m;i++)**

**{**

**printf("--------");**

**}**

**printf("\n");**

**for(j=0; j<=m; j++)**

**{**

**printf("%d\t", a[j]);**

**}**

**for(int i=0; i<n; i++)**

**{**

**for(j=i+1; j<n; j++)**

**{**

**if(p[i] > p[j])**

**{**

**temp = p[i];**

**p[i] = p[j];**

**p[j] = temp;**

**temp = k[i];**

**k[i] = k[j];**

**k[j] = temp;**

**}**

**}**

**}**

**for(i=0; i<n; i++)**

**{**

**wat[i] = k[i] - bur1[i];**

**tur[i] = k[i];**

**}**

**for(i=0; i<n; i++)**

**{**

**ttur += tur[i];**

**twat += wat[i];**

**totaltime += bur1[i];**

**}**

**printf("\n\n");**

**for(i=0; i<60; i++)**

**{**

**printf("-");**

**}**

**printf("\n Time | Process | T-Time= | W-Time= |\n");**

**printf(" | Completed | t(PC)-t(PS) | T.T - T(process time) | \n");**

**for(i=0; i<60; i++)**

**{**

**printf("-");**

**}**

**printf("\n 0 | - |\t - | - |\n");**

**for (i=0; i<n; i++)**

**{**

**printf("\n %2d\t| P%d |\t%5d | \t%5d |", bur1[i], p[i]+1 , tur[i],wat[i]);**

**printf("\n");**

**}**

**for(i=0; i<60; i++)**

**{**

**printf("-");**

**}**

**awat = (float)twat / n;**

**atur = (float)ttur / n;**

**printf("\n\nAverage waiting time given by: total waiting time / no. of processes \nSo here: ");**

**for(i=0; i<n; i++)**

**{**

**printf("%d",wat[i]);**

**for(j=0; j<n-1; j++)**

**{**

**printf("+");**

**break;**

**}**

**}**

**printf("/ %d = %5.2f ms \n",n,awat);**

**printf("\n\nAverage Turn-around time given by: total turn-around time / no. of processes \nSo here: ");**

**for(i=0; i<n; i++)**

**{**

**printf("%d",tur[i]);**

**for(j=0; j<n-1; j++)**

**{**

**printf("+");**

**break;**

**}**

**}**

**printf("/ %d = %5.2f ms \n",n,atur);**

**cpu\_util= ( totaltime/ (totaltime + 0) ) \* 100;**

**printf("\n\nCPU utilization, given by: (cpu busy time)/(cpu busytime+ idle time) \nSo here it is %5.2f / (%5.2f + 0) = %d %%\n\n",totaltime,totaltime, cpu\_util);**

**throughput= n/ ( totaltime);**

**printf("\nThroughput, given by: No. of Processes/ Totaltime,\nSo here it is: %d / %5.2f = %5.2f processes per ms",n,totaltime,throughput);**

**return 1;**

**}**

**Output:**

