

Department of Computer Science and Engineering

**Course Code :** CSE -334

**Course Title :** Operating System Lab.

**Report :** 07.

**Report Name :** Implementation of Round-Robin Algorithm (preemptive).

**Performance Date :** 19-10-2015.

**Submission Date :** 02-11-2015.

**REMARKS**

**NAME : Md. Jahidul alam**

**ID :** **12-023-1-02-00419**

**DEPT :** **CSE**

**YEAR :** **2015**

**SEM. :** **6th**

**Introduction:** This algorithm is Shortest-Process-Next (SPN). Round-Robin is a non-preemptive discipline in which waiting job (or process) with the smallest estimated run-time-to-completion is run next.

**Objective:** Implementation of Round-Robin algorithm.

**Source Code:**

#include<stdio.h>

void main()

{

int k,j,q,i,n,ts,temp;

int aw,float awt;

int bt[10],wt[10],te[10],rt[10],at[10];j=0; printf("enter number of process :\t");

scanf("%d",&n);

printf("\n enter brust time and arriavl time\n");

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

{

printf("P%d \nbrust time ",i+1);

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

printf("ariavl time : ");

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

te[i]=0; wt[i]=0;

}

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

{

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

{

if(at[i]>at[j])

{

temp=at[i];

at[i]=at[j];

at[j]=temp;

temp=bt[i];

bt[i]=bt[j];

bt[j]=temp;

}

}

}

printf("\n enter time slice\t");

scanf("%d",&ts); q=0; clrscr();

printf("\nprocess :") ;

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

{

printf(" %d",i+1);

}

printf("\nBrust time :");

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

{

printf(" %d",bt[i]); rt[i]=bt[i];

}

printf("\nArrival time :");

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

{

printf(" %d",at[i]);

}

printf("\n Gaint chart \n");

j=0;

while(j<=n){

j++;

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

{

if(rt[i]==0) continue;

if(rt[i]>ts)

{ printf("\n %d\t P%d",q,i+1);

q=q+ts;

rt[i]=rt[i]-ts;

te[i]=te[i]+1; }

else

{

printf("\n %d\t P%d",q,i+1);

wt[i]=q-te[i]\*ts;

q=q+rt[i];

rt[i]=rt[i]-rt[i];

}

}

} //end of while

awt=0;

printf("\n Process Waitnig time");

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

{

wt[i]=wt[i]-at[i];

printf("\n P%d : %d",i+1,wt[i]); awt=awt+wt[i];

}

aw=awt;

printf("\ntotal waiting time %d",aw);

printf("\n Avg wainting time %f ",awt/n);

getch();

}

**Input:**

enter number of process : 3

enter brust time and arriavl time

P1

brust time 1

ariavl time : 7

P2

brust time 2

ariavl time : 2

P3

brust time 4

ariavl time : 0

enter time slice 3

**Output:**

process : 1 2 3

Brust time : 4 2 1

Arrival time : 0 2 7

Gaint chart

0 P1

3 P2

5 P3

6 P1

Process Waitnig time

P1 : 3

P2 : 1

P3 : -2

total waiting time 2

Avg wainting time 0.666667

**Discussion:**

1. There are no problems of compilation.
2. The problem was successfully executed.