

Department of Computer Science and Engineering

**Course Code :** CSE -334

**Course Title :** Operating System Lab.

**Report :** 06.

**Report Name :** Implementation of Round-Robin Algorithm.

**Performance Date :** 05-10-2015.

**Submission Date :** 12-10-2015.

**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>

int main()

{

clrscr();

int p[50],b[50],i,n,t;

printf("Input:\n\n");

printf("Enter number of Process and Time Quante : ");

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

printf("\n\n");

printf("Enter Process and Burst Time : ");

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

{

printf("P");

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

}

int f=0,j;

printf("\n\nOutput : \n\n");

while(1)

{

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

{

if(b[i]!=0)

{f=1;break;}

}

if(f==1)

{

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

{

if(b[i]==0)

continue;

if(b[i]<t)

{

printf("P");

for(j=0;j<b[i];j++)

{printf("p[%d] ",p[i]);}

b[i]=0;

}

else

{b[i]=b[i]-t;

for(j=0;j<t;j++)

{printf("p[%d] ",p[i]);}

}

}

}

else

{ break;}

f=0;

}

getch();

return 0;

}

**Input:** Enter number of Process and Time Quant: 3 2

Enter Process and Burst Time :

P1 14

P2 3

P3 3

**Output:**

p[1] p[1] p[2] p[2] p[3] p[3] p[1] p[1] Pp[2] Pp[3] p[1] p[1] p[1] p[1] p[1] p[1

] p[1] p[1] p[1] p[1]

**Discussion:**

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