|  |  |
| --- | --- |
| **Ex.No:5.b** | **CPU SCHEDULING ALGORITHMS** |
| **ROUND ROBIN SCHEDULING** |

# AIM:

To write a C program for implementation of Round Robin scheduling algorithms.

# ALGORITHM:

Step 1: Inside the structure declare the variables.

Step 2: Declare the variable i,j as integer, totwtime and totttime is equal to zero. Step 3: Get the value of „n‟ assign p and allocate the memory.

Step 4: Inside the for loop get the value of burst time and priority and read the time quantum. Step 5: Assign wtime as zero.

Step 6: Check p[i].pri is greater than p[j].pri .

Step 7: Calculate the total of burst time and waiting time and assign as turnaround time. Step 8: Stop the program.

# PROGRAM:

#include<stdio.h> #include<stdlib.h> struct rr

{

int pno,btime,sbtime,wtime,lst;

}p[10];

int main()

{

int pp=-1,ts,flag,count,ptm=0,i,n,twt=0,totttime=0; printf("\n round robin scheduling ");

printf("enter no of processes:"); scanf("%d",&n);

printf("enter the time slice:"); scanf("%d",&ts); printf("enter the burst time"); for(i=0;i<n;i++)

{

printf("\n process%d\t",i+1); scanf("%d",&p[i].btime);

p[i].wtime=p[i].lst=0; p[i].pno=i+1; p[i].sbtime=p[i].btime;

}

printf("scheduling ...\n"); do

{

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

{

count=p[i].btime; if(count>0)

{

flag=-1; count=(count>=ts)?ts:count; printf("\n process %d",p[i].pno); printf("from%d",ptm); ptm+=count; printf("to%d",ptm);

p[i].btime-=count; if(pp!=i)

{

pp=i;

p[i].wtime+=ptm-p[i].lst-count; p[i].lst=ptm;

}

}