**LOVELY PROFESSIONAL UNIVERSITY**

**Academic Task-3**

**(Operating System)**

School of Computer Science and Engineering( Faculty of Technology And Sciences )

**Name of the faculty member: Ashu**

**Course Code: CSE 316 Course Title:** Operating System

**Term:**219202

Date of Allotment: 01/03/2020 Date of Submission: 31/03/2020

**Name: Manish Kumar**

**Reg No:** 11805306

**Roll No: 57**

**Section:** K18UW

**Email Address:** manishgupta9507@gmail.com

**GitHub Link:**

**Problem** **1**:

Design a scheduling program to implements a Queue with two levels:

Level 1 : Fixed priority preemptive Scheduling

Level 2 : Round Robin Scheduling

For a Fixed priority preemptive Scheduling (Queue 1), the Priority 0 is highest priority. If one process P1 is scheduled and running, another process P2 with higher priority comes. The New process (high priority) process P2 preempts currently running process P1 and process P1 will go to second level queue. Time for which process will strictly execute must be considered in the multiples of 2..

All the processes in second level queue will complete their execution according to round robin scheduling.

Consider: 1. Queue 2 will be processed after Queue 1 becomes empty.

2. Priority of Queue 2 has lower priority than in Queue 1.

**Description:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::**

enters a Queue with two levels:

Level 1 : Fixed priority preemptive Scheduling

Level 2 : Round Robin Scheduling

For a Fixed priority preemptive Scheduling (Queue 1), the Priority 0 is highest priority. If one

process P1 is scheduled and running, another process P2 with higher priority comes. The new

process (high priority) process P2 preempts currently running process P1 and process P1 will go

to second level queue. Time for which process will strictly execute must be considered in the

multiples of 2.

All the processes in second level queue will complete their execution according to round robin

scheduling.

Consider: 1. Queue 2 will be processed after Queue 1 becomes empty.

2. Priority of Queue 2 has lower priority than in Queue 1

**Algorithm::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::**

#include<stdio.h>

#include<string.h>

#include<conio.h>

int main()

{

char p[10][5],temp[5];

int i,j,pt[10],wt[10],totwt=0,pr[10],temp1,n;

float avgwt;

printf("enter no of processes:");

scanf("%d",&n);

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

{

printf("enter process%d name:",i+1);

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

printf("enter process time:");

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

printf("enter priority:");

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

}

for(i=0;i<n-1;i++)

{

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

{

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

{

temp1=pr[i];

pr[i]=pr[j];

pr[j]=temp1;

temp1=pt[i];

pt[i]=pt[j];

pt[j]=temp1;

strcpy(temp,p[i]);

strcpy(p[i],p[j]);

strcpy(p[j],temp);

}

}

}

wt[0]=0;

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

{

wt[i]=wt[i-1]+wt[i-1];

totwt=totwt+wt[i];

}

avgwt=(float)totwt/n;

printf("p\_name\t p\_time\t priority\t w\_time\n");

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

{

printf(" %s\t %d\t %d\t %d\n" ,p[i],pt[i],pr[i],wt[i]);

}

printf("total waiting time=%d\n avg waiting time=%f",totwt,avgwt);

int ts,pid[10],need[10],wt1[10],tat[10],i1,j1,n2,n1;

int bt[10],flag[10],ttat=0,twt=0;

float awt,atat;

printf("\nEnter the number of Processors \n");

scanf("%d",&n);

n1=n;

printf("\n Enter the Timeslice \n");

scanf("%d",&ts);

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

{

printf("\n Enter the process ID %d",i);

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

printf("\n Enter the Burst Time for the process");

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

need[i]=bt[i];

}

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

{

flag[i]=1;

wt[i]=0;

}

while(n!=0)

{

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

{

if(need[i]>=ts)

{

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

{

if((i!=j)&&(flag[i]==1)&&(need[j]!=0))

wt[j]+=ts;

}

need[i]-=ts;

if(need[i]==0)

{

flag[i]=0;

n--;

}

}

else

{

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

{

if((i!=j)&&(flag[i]==1)&&(need[j]!=0))

wt[j]+=need[i];

}

need[i]=0;

n--;

flag[i]=0;

}

}

}

for(i=1;i<=n1;i++)

{

tat[i]=wt[i]+bt[i];

twt=twt+wt[i];

ttat=ttat+tat[i];

}

awt=(float)twt/n1;

atat=(float)ttat/n1;

printf("\n\n Process \t Process ID \t BurstTime \t Waiting Time \t TurnaroundTime \n ");

for(i=1;i<=n1;i++)

{

printf("\n %5d \t %5d \t\t %5d \t\t %5d \t\t %5d \n", i,pid[i],bt[i],wt[i],tat[i]);

}

printf("\n The average Waiting Time=4.2f",awt);

printf("\n The average Turn around Time=4.2f",atat);

}