NAME: Aryaman Sinha

Roll No.: RK18KKA27

Reg. No.: 11804531

CSE-316

Operating System

Submitted to:

Dr. Baljit Singh Saini

Assignment Question

Q6. Write a program for multilevel queue scheduling algorithm. There must be three queues generated. There must be specific range of priority associated with every queue. Now prompt the user to enter number of processes along with their priority and burst time. Each process must occupy the respective queue with specific priority range according to its priority. Apply Round Robin algorithm with quantum time 4 on queue with highest priority range. Apply priority scheduling algorithm on the queue with medium range of priority and First come first serve algorithm on the queue with lowest range of priority. Each and every queue should get a quantum time of 10 seconds. CPU will keep on shifting between queues after every 10 seconds.

Approach:

I first made three different codes for Round Robin algorithm, Priority Scheduling algorithm, and First Come First Serve algorithm namely round\_robin(),priority() and fcfs().

Then I merged the three and used a Round Robin algorithm with a Quantum time of 10 with the three queues i.e. round\_robin1().

I then set the priority level for each queue:-

Round robin: 1 - 5.

Priority : 5 – 10.

FCFS : 10 – 15.

Code

#include<stdio.h>

int Total=0,t1=0,t2=0,t3=0;

int n,i,at[30],bt[30],pr[30],j=0,k=0,l=0;//substitutes for all the times needed and parameter for lenght of each queue

int arrival\_time1[30],arrival\_time2[30],arrival\_time3[30],priority2[30],process2[30];

int burst\_time1[30],burst\_time2[30],burst\_time3[30];

int total,x,temp[30],counter=0;

float avg\_waiting\_time1=0.0,avg\_turnaround\_time1=0.0;

int p,waiting\_time3[30],turnaround\_time3[30];

float avg\_waiting\_time3=0.0,avg\_turnaround\_time3=0.0;

int pos,q,temp1,sum=0,waiting\_time2[30],turnaround\_time2[30];

float avg\_waiting\_time2=0.0,avg\_turnaround\_time2=0.0;

//round robin algorithm for the first queue

void round\_robin()

{

printf("Time Quantum for Queue1 is 4\n");

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

{

temp[i]=burst\_time1[i];

}

printf("\nProcess ID\tBurst Time\t Turnaround Time\t Waiting Time\n");

x=j;

for(i=0,total=0;x!=0;)

{

if(temp[i]<=4&&temp[i]>0)

{

printf("\nProcess[%d] of Queue1 is running for %d units",i+1,temp[i]);

total=total+temp[i];

temp[i]=0;

counter=1;

}

else if(temp[i]>0)

{

printf("\nProcess[%d] of Queue1 is running for 4 units",i+1);

temp[i]=temp[i]-4;

total=total+4;

}

if(temp[i]==0&&counter==1)

{

x--;

printf("\nProcess[%d]\t%d\t%d\t%d",i+1,burst\_time1[i],total-arrival\_time1[i],total-arrival\_time1[i]-burst\_time1[i]);

avg\_waiting\_time1=avg\_waiting\_time1+total-arrival\_time1[i]-burst\_time1[i];

avg\_turnaround\_time1=avg\_turnaround\_time1+total-arrival\_time1[i];

counter = 0;

}

if(i==j-1)

{

i=0;

}

else if(arrival\_time1[i+1]<=total)

{

i++;

}

else

{

i=0;

}

}

avg\_waiting\_time1=avg\_waiting\_time1/j;

avg\_turnaround\_time1=avg\_turnaround\_time1/j;

printf("\nAverage Waiting Time:%f",avg\_waiting\_time1);

printf("\nAverage Turnaround Time:%f\n",avg\_turnaround\_time1);

}

//priority based scheduling algorithm for queue 2

void priority()

{

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

{

pos=i;

for(q=i+1;q<k;q++)

{

if(priority2[q]<priority2[pos])

{

pos=q;

}

}

temp1=priority2[i];

priority2[i]=priority2[pos];

priority2[pos]=temp1;

temp1=burst\_time2[i];

burst\_time2[i]=burst\_time2[pos];

burst\_time2[pos]=temp1;

temp1=process2[i];

process2[i]=process2[pos];

process2[pos]=temp1;

}

waiting\_time2[0]=0;

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

{

waiting\_time2[i]=0;

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

{

waiting\_time2[i]=waiting\_time2[i]+burst\_time2[j];

}

sum=sum+waiting\_time2[i];

}

avg\_waiting\_time2=sum/k;

sum=0;

printf("\nProcess ID\t\tBurst Time\t Waiting Time\t Turnaround Time\n");

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

{

turnaround\_time2[i]=burst\_time2[i]+waiting\_time2[i];

sum=sum+turnaround\_time2[i];

printf("\nProcess[%d]\t\t%d\t\t %d\t\t %d\n",process2[i],burst\_time2[i],waiting\_time2[i],turnaround\_time2[i]);

}

avg\_turnaround\_time2=sum/k;

printf("\nAverage Waiting Time:\t%f",avg\_waiting\_time2);

printf("\nAverage Turnaround Time:\t%f\n",avg\_turnaround\_time2);

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

{

while(burst\_time2[i]!=0)

{

if(burst\_time2[i]>10)

{

printf("\nProcess[%d] of Queue2 is running for 10 units",i+1);

burst\_time2[i]=burst\_time2[i]-10;

}

else if(burst\_time2[i]<=10)

{

printf("\nProcess[%d] of Queue2 is running for %d units",i+1,burst\_time2[i]);

burst\_time2[i]=0;

}

}

}

}

//first come first serve algorithm for queue 3

void fcfs()

{

waiting\_time3[0] = 0;

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

{

waiting\_time3[i] = 0;

for(p=0;p<l;p++)

{

waiting\_time3[i]=waiting\_time3[i]+burst\_time3[p];

}

}

printf("\nProcess\t\tBurst Time\tWaiting Time\tTurnaround Time\n");

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

{

turnaround\_time3[i]=burst\_time3[i]+waiting\_time3[i];

avg\_waiting\_time3=avg\_waiting\_time3+waiting\_time3[i];

avg\_turnaround\_time3=avg\_turnaround\_time3+turnaround\_time3[i];

printf("\nProcess[%d]\t\t%d\t\t%d\t\t%d\n",i+1,burst\_time3[i],waiting\_time3[i],turnaround\_time3[i]);

}

avg\_waiting\_time3=avg\_waiting\_time3/l;

avg\_turnaround\_time3=avg\_turnaround\_time3/l;

printf("\nAverage Waiting Time=%f",avg\_waiting\_time3);

printf("\nAverage Turnaround Time=%f",avg\_turnaround\_time3);

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

{

while(burst\_time3[i]!=0)

{

if(burst\_time3[i]>10)

{

printf("\nProcess[%d] of Queue3 is running for 10 units",i+1);

burst\_time3[i]=burst\_time3[i]-10;

}

else if(burst\_time3[i]<=10)

{

printf("\nProcess[%d] of Queue2 is running for %d units",i+1,burst\_time3[i]);

burst\_time3[i]=0;

}

}

}

}

//round robin with quantum time 10 for the three queues

void round\_robin1()

{

printf("Time Quantum between the 3 queues is 10\n");

for(i=1;i<Total;i=i+10)

{

if(t1>10)

{

printf("Queue1 is running for 10 units\n");

t1=t1-10;

}

else if(t1<=10&&t1!=0)

{

printf("Queue1 is running for %d units\n",t1);

t1=0;

}

if(t2>10)

{

printf("Queue2 is running for 10 units\n");

t2=t2-10;

}

else if(t2<=10&&t2!=0)

{

printf("Queue2 is running for %d units\n",t2);

t2=0;

}

if(t3>10)

{

printf("Queue3 is running for 10 units\n");

t3=t3-10;

}

else if(t3<=10&&t3!=0)

{

printf("Queue3 is running for %d units\n",t3);

t3=0;

}

}

}

int main()

{

printf("Enter the no. of process you want to enter\n");

scanf("%d",&n);

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

{

printf("Enter details of process[%d]\n",i+1);

printf("Arrival Time:");

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

printf("Burst Time:");

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

printf("Priority(1 to 15):");

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

Total=Total+bt[i];

}

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

{

if(pr[i]>=1&&pr[i]<=5) //round robin queue

{

printf("\n\nProcess[%d] belongs to Queue 1\n",i+1);

arrival\_time1[j]=at[i];

burst\_time1[j]=bt[i];

j++;

t1=t1+bt[i];

}

else if(pr[i]>=6&&pr[i]<=10) //priority queue

{

printf("Process[%d] belongs to Queue 2\n",i+1);

arrival\_time2[k]=at[i];

burst\_time2[k]=bt[i];

priority2[k]=pr[i];

process2[k]=k+1;

k++;

t2=t2+bt[i];

}

else if(pr[i]>=11&&pr[i]<=15) //fcfs queue

{

printf("Process[%d] belongs to Queue 3\n\n\n\n",i+1);

arrival\_time3[l]=at[i];

burst\_time3[l]=bt[i];

l++;

t3=t3+bt[i];

}

}

round\_robin1();

round\_robin();

fcfs();

priority();

return 0;

}

Snapshots and

Test cases

There are two test cases one with 6 inputs and one with 8.

Test Case 1: