*->This code explanation is for Question no. 13, as per given list of Questions provided to us.*

*Code explanation in 15 major points :-*

1. Declarating the queues with maximum 100 processes:-

struct info \*q1[100];

struct info \*q2[100];

struct info \*q3[100];

1. Defining Length of q1, q2, q3 respectively :-

int l=0;

int m=0;

int h=0;

1. Mlq() pushes the processes int0 respective queue according to priorities.

void mlq(struct info p[])

{

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

{

if(p[i].priority>0&&p[i].priority<=10)

{

q1[l]=&p[i];

l++;

}

else if(p[i].priority>10&&p[i].priority<=20)

{

q2[m]=&p[i];

m++;

}

else

{

q3[h]=&p[i];

h++;

}

}

}

1. Sort() sorts the process according arrival time and process id.

void sort(struct info \*q[],int size)

{

struct info temp;

for(int i=0;i<size-1;i++)

{

for(int j=0;j<size-i-1;j++)

{

if(q[j]->at>q[j+1]->at)

{

temp=\*q[j];

\*q[j]=\*q[j+1];

\*q[j+1]=temp;

}

else if(q[j]->at==q[j+1]->at)

{

if(q[j]->p\_id>q[j+1]->p\_id)

{

temp=\*q[j];

\*q[j]=\*q[j+1];

\*q[j+1]=temp;

}

}

}

}

}

1. Orenq() pushes the queue number into queue

void orenq(int x)

{

if(orr==100)

{

printf("over flow\n");

}

else if (orf == -1 && orr == -1)

{

orf++;

orr++;

oready[orr]=x;

}

else

{

orr++;

oready[orr]=x;

}

}

1. Ordq() removes the queue number from queue.

void ordq()

{

if(orr==orf)

{

printf("\n");

}

else

{

orf++;

}

}

1. Hrenq() will add process to hready q

void hrenq(struct info x)

{

if(hrr==100)

{

printf("over flow\n");

}

else if (hrf == -1 && hrr == -1)

{

hrf++;

hrr++;

hready[hrr]=x;

}

else

{

hrr++;

hready[hrr]=x;

}

}

1. Hrdq removes the process from hready q.

void hrdq()

{

if(hrr==hrf)

{

printf("\n");

}

else

{

hrf++;

}

}

1. Mrenq() adds process to mready q.

void mrenq(struct info x)

{

if(mrr==100)

{

printf("over flow\n");

}

else if (mrf == -1 && mrr == -1)

{

mrf++;

mrr++;

mready[mrr]=x;

}

else

{

mrr++;

mready[mrr]=x;

}

}

1. Lrenq() adds process to lready q.

void lrenq(struct info x)

{

if(lrr==100)

{

printf("over flow\n");

}

else if (lrf == -1 && lrr == -1)

{

lrf++;

lrr++;

lready[lrr]=x;

}

else

{

lrr++;

lready[lrr]=x;

}

}

1. Ismempty() will check weather mready q empty or not. (SIMILARLY DONE FOR hready & oready, )

int ismempty()

{

int s=0;

for(int i=mrf;i<=mrr;i++)

{

if(mready[i].bt!=0)

s=1;

}

if(s==0)

{

return 1;

}

else return 0;

}

1. Prior() will return the process index with high priority in mready q.

int prior()

{

int index;

int j=mrf;

while(j<=mrr)

{

if(mready[j].bt!=0)

{

index=j;

break;

}

else

{

j++;

}

}

for(int i=mrf;i<=mrr;i++)

{

if(mready[i].priority>mready[index].priority&&mready[i].bt!=0)

{

index=i;

}

}

return index;

}

1. fcfs() will return the index of the process with least arrival time in lready q

int fcfs()

{

int index;

int lat;

int j=lrf;

while(j<=lrr)

{

if(lready[j].bt!=0)

{

index=j;

break;

}

else

j++;

}

for(int i=lrf;i<=lrr;i++)

{

if(lready[i].at<lready[index].at&&lready[i].bt!=0)

{

index=i;

}

}

return index;

}

void updatel(int x)

{

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

{

if(q1[i]->at==x)

{

lrenq(\*q1[i]);

}

}

}

void updatem(int x)

{

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

{

if(q2[i]->at==x)

{

mrenq(\*q2[i]);

}

}

}

void updateh(int x)

{

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

{

if(q3[i]->at==x)

{

hrenq(\*q3[i]);

}

}

}

1. Minatl returns the index of process with least arrival time in lready q

int minatl() (SIMILARLY DONE FOR mready and hready)

{

int index;

int mint;

int j=lrf;

while(j<=lrr)

{

if(lready[j].bt!=0)

{

index=j;

mint=lready[j].at;

break;

}

else j++;

}

for(int i=lrf;i<lrr;i++)

{

if(lready[i].at<mint&&lready[i].bt!=0)

{

index=i;

mint=lready[i].at;

}

}

return index;

1. WtTat() will Calculate and print the wt and tat of each process.

void WtTat(int chart[],struct info p[],int tt)

{

int pid;

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

{

int l;

pid=p[i].p\_id;

for(int j=tt-1;j>=0;j--)

{

if(chart[j]==pid)

{

l=j+1;

break;

}

}

p[i].tat=l-p[i].at;

p[i].wt=p[i].tat-p[i].bt;

}

printf("\n\n");

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

{

printf("pid:%d\t",p[i].p\_id);

printf("Waiting Time:%d\t",p[i].wt);

printf("Turn Around Time:%d\t",p[i].tat);

printf("\n\n");

}

}

***Code execution image:-***

****

**Thank you…**