

CPU SCHEDULING

Muhammed Faheem Shamsudeen

S5 CSE 37

```
#include<stdio.h>
int readyQueue[10],front=-1,rear=-1;
float turnSum=0,waitingSum=0,completionSum=0,responseSum=0;

int completion(int sysTime,int burst)
{
    int completion = sysTime + burst;
    printf("%d\t\t\t",completion);
    completionSum += completion;
    return completion;
}

int turnAround(int completion,int arrival)
{
    int turnAround = completion - arrival;
    printf("%d\t\t\t",turnAround);
    turnSum += turnAround;
    return turnAround;
}

void waiting(int turnAround,int burst)
{
    int waiting = turnAround - burst;
    printf("%d\t\t\t",waiting);
    waitingSum += waiting;
}

void response(int arrival,int sysTime)
{
    int response = sysTime - arrival;
    printf("%d\n",response);
    responseSum += response;
}

void swap(int *val1, int *val2) //Swap func for sort
{
    int temp = *val1;
```

```

*val1 = *val2;
*val2 = temp;
}

void sort(int n,int mode,int arr[n][4])
{
for(int i=0;i<n;++i)
{
for(int j=i+1;j<n;++j)
{
if(arr[i][mode]>arr[j][mode])
{
swap(&arr[i][0], &arr[j][0]);
swap(&arr[i][1], &arr[j][1]);
swap(&arr[i][2], &arr[j][2]);
swap(&arr[i][3], &arr[j][3]);
}}}
for(int i=0;i<n;++i) //sorting with pid
{
for(int j=i;j<n;++j)
{
if(arr[i][mode]==arr[j][mode] && arr[i][0]>arr[j][0])
{
swap(&arr[i][0], &arr[j][0]);
swap(&arr[i][1], &arr[j][1]);
swap(&arr[i][2], &arr[j][2]);
swap(&arr[i][3], &arr[j][3]);
}}}
}

void printAvg(int n)
{
printf("\nAverage response time is : %f\n",responseSum/n);
printf("Average completion time is : %f\n",completionSum/n);
printf("Average turn around Time is : %f\n",turnSum/n);
printf("Average waiting time is : %f\n",waitingSum/n);
responseSum = completionSum = turnSum = waitingSum = 0 ;
}

void nonPreEmptive(int n,int arr[n][4])
{
int key=0,sysTime=0,flag[n];
for(int i=0;i<n;++i)

```

```

flag[i]=0;

printf("\nPID \t COMPLETION TIME \t TURNAROUND TIME \t WAITING TIME \t\t RESPONSE
TIME\n");

while(key<n)
{
int temp = sysTime;
repeat:for(int i=0;i<n;++i)
{
if(arr[i][1]<=sysTime && flag[i]!=1)
{
printf("%d\t\t",arr[i][0]); //printing pid
int comp = completion(sysTime,arr[i][2]);
int turn = turnAround(comp,arr[i][1]);
waiting(turn,arr[i][2]);
response(arr[i][1],sysTime);
sysTime += arr[i][2];
++key;
flag[i]=1;
goto repeat;
}}
if(temp==sysTime)           //to increment system time if no process is executed
++sysTime;
}
printAvg(n);
}

void preEmptive(int n,int q,int arr[n][4])
{
int remTime[n],remain=n,time=0,flag=0;

for(int i=0;i<n;++i) //storing burst time in remTime
remTime[i] = arr[i][2];

printf("\nPID \t COMPLETION TIME \t TURNAROUND TIME \t WAITING TIME \t\t RESPONSE
TIME\n");
for(int i=0;remain!=0;)
{
if(remTime[i]<=q && remTime[i]>0)
{
time += remTime[i];
remTime[i]=0;

```

```

flag=1;
}
else if(remTime[i]>0)
{
time += q;
remTime[i] -= q;
}
if(remTime[i]==0 && flag==1)
{
remain--;
printf("%d\t\t%d\t\t\t",arr[i][0],time); //printing pid and completion
int turn = turnAround(time,arr[i][1]);
waiting(turn,arr[i][2]);
response(arr[i][1],time);
flag=0;
}
if(i==n-1)
i=0;
else if(arr[i+1][1] <= time)
i++;
else
i=0;
}
printAvg(n);
}

void fcfs(int n,int arr[n][4])
{
printf("\nFIRST COME FIRST SERVE\n");
sort(n,1,arr);
nonPreEmptive(n,arr);
}

void sjf(int n,int arr[n][4])
{
printf("\nSHORTEST JOB FIRST\n");
sort(n,2,arr);
nonPreEmptive(n,arr);
}

void priority(int n,int arr[n][4])
{
printf("\nPRIORITY\n");

```

```

sort(n,3,arr);
nonPreEmptive(n,arr);
}

void roundRobin(int n,int q,int arr[n][4])
{
printf("\\nROUND ROBIN\\n");
sort(n,1,arr);
preEmptive(n,q,arr);
}

int main()
{
int n,q;
printf("Enter the number of processes\\n");
scanf("%d",&n);
int arr[n][4];
for(int i=0;i<n;++i) //input Arrival, burst time and priority
{
printf("Enter the ARRIVAL TIME ,BURST TIME and PRIORITY of process %d\\n",i+1);
scanf("%d%d%d",&arr[i][1],&arr[i][2],&arr[i][3]);
arr[i][0] = i+1; //pid
}
printf("Enter the time Quantum\\n");
scanf("%d",&q);
fcfs(n,arr);
sjf(n,arr);
priority(n,arr);
roundRobin(n,q,arr);
}

```

OUTPUT

Enter the number of processes

4

Enter the ARRIVAL TIME ,BURST TIME and PRIORITY of process 1

0 9 1

Enter the ARRIVAL TIME ,BURST TIME and PRIORITY of process 2

1 5 2

Enter the ARRIVAL TIME ,BURST TIME and PRIORITY of process 3

2 3 3

Enter the ARRIVAL TIME ,BURST TIME and PRIORITY of process 4

3 4 4

Enter the time Quantum

5

FIRST COME FIRST SERVE

PID	COMPLETION TIME	TURNAROUND TIME	WAITING TIME	RESPONSE TIME
1	9	9	0	0
2	14	13	8	8
3	17	15	12	12
4	21	18	14	14

Average response time is : 8.500000

Average completion time is : 15.250000

Average turn around Time is : 13.750000

Average waiting time is : 8.500000

SHORTEST JOB FIRST

PID	COMPLETION TIME	TURNAROUND TIME	WAITING TIME	RESPONSE TIME
1	9	9	0	0
3	12	10	7	7
4	16	13	9	9
2	21	20	15	15

Average response time is : 7.750000

Average completion time is : 14.500000

Average turn around Time is : 13.000000

Average waiting time is : 7.750000

PRIORITY

PID	COMPLETION TIME	TURNAROUND TIME	WAITING TIME	RESPONSE TIME
1	9	9	0	0
2	14	13	8	8
3	17	15	12	12
4	21	18	14	14

Average response time is : 8.500000

Average completion time is : 15.250000

Average turn around Time is : 13.750000

Average waiting time is : 8.500000

ROUND ROBIN

PID	COMPLETION TIME	TURNAROUND TIME	WAITING TIME	RESPONSE TIME
2	10	9	4	9
3	13	11	8	11
4	17	14	10	14
1	21	21	12	21

Average response time is : 13.750000

Average completion time is : 0.000000

Average turn around Time is : 13.750000

Average waiting time is : 8.500000

faheemshams@Faheems-MacBook-Air System software %