LAB 12

PREEMPTIVE CPU SCHEDULING ALGORITHMS

P.PRIYADHARSHINI

2019103562

26-04-2021

ROUND ROBIN SCHEDULING ALGORITHM:

```
[s2019103562@centos8-linux Mon Apr 26 09:35 AM lab12]$ gcc rr.c -o rr
[s2019103562@centos8-linux Mon Apr 26 09:35 AM lab12]$ ./rr
Enter the number of processes:  4

Enter the arrival time for process 1:   0

Enter the burst time for process 1:     5

Enter the arrival time for process 2:   1

Enter the burst time for process 2:     4

Enter the arrival time for process 3:   2

Enter the burst time for process 3:     2

Enter the arrival time for process 4:   3

Enter the burst time for process 4:     1

Enter the time quantum for the process: 2
Process          ArrivalTime             BurstTime               TAT             Waiting time
Process[3]               2               2               4                               2
Process[4]               3               1               4                               3
Process[2]               1               4               10                              6
Process[1]               0               5               12                              7

AVERAGE WAITING TIME:    4.500000
AVERAGE TURNAROUND TIME:         7.500000
[s2019103562@centos8-linux Mon Apr 26 09:36 AM lab12]$
```

```
[s2019103562@centos8-linux Mon Apr 26 09:36 AM lab12]$ cat rr.c
#include<stdio.h>
int main(){
        int i,n,sum=0,count=0,quant,wt=0,tat=0,y;
        float avg_wt,avg_tat;
        printf("Enter the number of processes:\t");
        scanf("%d",&n);
        int at[n],bt[n],temp[n];
        y=n;
        for(i=0;i<n;i++){
                printf("\nEnter the arrival time for process %d:\t",i+1);
                scanf("%d",&at[i]);
                printf("\nEnter the burst time for process %d:\t",i+1);
                scanf("%d",&bt[i]);
                temp[i]=bt[i];
        }
        printf("\nEnter the time quantum for the process:\t");
        scanf("%d",&quant);
        printf("Process\t\t ArrivalTime\t\t BurstTime\t\t TAT\t\t Waiting time\n");
        for(sum=0,i=0;y!=0;){
                if(temp[i]<=quant && temp[i]>0){
                        sum=sum+temp[i];
                        temp[i]=0;
                        count=1;
                }
                else if(temp[i]>0){
                        temp[i]=temp[i]-quant;
                        sum=sum+quant;
                }
                if(temp[i]==0 && count==1){
                        y--;
                        printf("Process[%d]\t\t %d\t\t %d\t\t %d\t\t\t\t\t %d\n",i+1,at[i],bt[i],sum-at[i],sum-at[i]-bt[i]);
                        wt=wt+sum-at[i]-bt[i];
                        tat=tat+sum-at[i];
                        count=0;
                }
```

```
                }
                if(i==n-1)
                        i=0;
                else if(at[i+1]<=sum)
                        i++;
                else
                        i=0;
        }
        avg_wt=wt*1.0/n;
        avg_tat=tat*1.0/n;
        printf("\nAVERAGE WAITING TIME:\t %f\n",avg_wt);
        printf("AVERAGE TURNAROUND TIME:\t %f\n",avg_tat);
        return 0;
}

[s2019103562@centos8-linux Mon Apr 26 09:39 AM lab12]$
```

SHORTEST REMAINING TIME FIRST ALGORITHM:

```
[s2019103562@centos8-linux Mon Apr 26 09:50 AM lab12]$ vim srtf.c
[s2019103562@centos8-linux Mon Apr 26 09:50 AM lab12]$ gcc srtf.c -o srtf
[s2019103562@centos8-linux Mon Apr 26 09:50 AM lab12]$ ./srtf
Enter the number of processes:  6
Enter the arrival time for process 1:   0

Enter the burst time for process 1:     8
Enter the arrival time for process 2:   1

Enter the burst time for process 2:     4
Enter the arrival time for process 3:   2

Enter the burst time for process 3:     2
Enter the arrival time for process 4:   3

Enter the burst time for process 4:     1
Enter the arrival time for process 5:   4

Enter the burst time for process 5:     3
Enter the arrival time for process 6:   5

Enter the burst time for process 6:     2

AVERAGE WAIITNG TIME:   3.666667
AVERAGE TURNAROUND TIME:        7.000000
[s2019103562@centos8-linux Mon Apr 26 09:52 AM lab12]$
```

```
[s2019103562@centos8-linux Mon Apr 26 10:08 AM lab12]$ cat srtf.c
#include<stdio.h>
int main(){
        int n,count=0,i,j,smallest,time;
        double avg=0,tt=0,end;
        printf("Enter the number of processes:\t");
        scanf("%d",&n);
        int at[10],bt[10],x[10];
        for(i=0;i<n;i++){
                printf("Enter the arrival time for process %d:\t",i+1);
                scanf("%d",&at[i]);
                printf("\nEnter the burst time for process %d:\t",i+1);
                scanf("%d",&bt[i]);
        }
        for(i=0;i<n;i++)
                x[i]=bt[i];
        bt[9]=9999;
        for(time=0;count!=n;time++){
                smallest=9;
                for(i=0;i<n;i++){
                        if(at[i]<=time && bt[i]<bt[smallest] && bt[i]>0){
                                smallest=i;
                                bt[smallest]--;
                                if(bt[smallest]==0){
                                        count++;
                                        end=time+1;
                                        avg=avg+end-at[smallest]-x[smallest];
                                        tt=tt+end-at[smallest];
                                }
                        }
                }
        }
                printf("\nAVERAGE WAIITNG TIME:\t%f",avg/n);
                printf("\nAVERAGE TURNAROUND TIME:\t%f\n",tt/n);
                return 0;
        }

[s2019103562@centos8-linux Mon Apr 26 10:08 AM lab12]$
```

PREEMPTIVE PRIORITY SCHEDULING:

```
[s2019103562@centos8-linux Mon Apr 26 01:18 PM lab12]$ vim pripre.c
[s2019103562@centos8-linux Mon Apr 26 01:19 PM lab12]$ gcc pripre.c -o pripre
[s2019103562@centos8-linux Mon Apr 26 01:19 PM lab12]$ ./pripre
Enter the number of processes:
4
Enter the details for processes[A]:
Enter arrival time:     0

Enter the burst time:   4

Enter priority: 3
Enter the details for processes[B]:
Enter arrival time:     1

Enter the burst time:   2

Enter priority: 2
Enter the details for processes[C]:
Enter arrival time:     2

Enter the burst time:   3

Enter priority: 4
Enter the details for processes[D]:
Enter arrival time:     4

Enter the burst time:   2

Enter priority: 1
```

```
PROCESS          ARRIVAL TIME          BURST TIME          PRIORITY          WAITING TIME          TURNAROUNDTIME

A                        0                   4                 3                   0                     4

C                        2                   3                 4                   2                     5

B                        1                   2                 2                   6                     8

D                        4                   2                 1                   5                     7

 AVERAGE WAITING TIME:  3.250000
 AVERAGE TURNAROUND TIME:       6.000000
[s2019103562@centos8-linux Mon Apr 26 01:20 PM lab12]$
```

```
[s2019103562@centos8-linux Mon Apr 26 01:20 PM lab12]$ cat pripre.c
#include<stdio.h>
struct process{
        char process_name;
        int arrival_time,burst_time,ct,waiting_time,turnaround_time,priority;
        int status;
}process_queue[10];
int limit;
void arrival_time_sorting(){
        struct process temp;
        int i,j;
        for(i=0;i<limit-1;i++){
                for(j=i+1;j<limit;j++){
                        if(process_queue[i].arrival_time>process_queue[j].arrival_time){
                                temp=process_queue[i];
                                process_queue[i]=process_queue[j];
                                process_queue[j]=temp;
                        }
                }
        }
}
int main(){
        int i,time=0,burst_time=0,largest;
        char c;
        float wait_time=0,turnaround_time=0,average_waiting_time,average_turnaround_time;
        printf("Enter the number of processes:\n");
        scanf("%d",&limit);
        for(i=0,c='A';i<limit;c++,i++){
                process_queue[i].process_name=c;
                printf("Enter the details for processes[%c]:\n",process_queue[i].process_name);
                printf("Enter arrival time:\t");
                scanf("%d",&process_queue[i].arrival_time);
                printf("\nEnter the burst time:\t");
                scanf("%d",&process_queue[i].burst_time);
                printf("\nEnter priority:\t");
                scanf("%d",&process_queue[i].priority);
                process_queue[i].status=0;
```

```
                printf("\nEnter the burst time:\t");
                scanf("%d",&process_queue[i].burst_time);
                printf("\nEnter priority:\t");
                scanf("%d",&process_queue[i].priority);
                process_queue[i].status=0;
                burst_time=burst_time+process_queue[i].burst_time;
        }
        arrival_time_sorting();
        process_queue[9].priority=-9999;
        printf("\nPROCESS\t\t ARRIVAL TIME\t\t BURST TIME\t\t PRIORITY\t\t WAITING TIME\t\t TURNAROUNDTIME\n");
        for(time=process_queue[0].arrival_time;time<burst_time;){
                largest=9;
                for(i=0;i<limit;i++){
                        if(process_queue[i].arrival_time<=time && process_queue[i].status!=1 && process_queue[i].priority>process_queue[largest].priority){
                                largest=i;
                        }
                }
                time=time+process_queue[largest].burst_time;
                process_queue[largest].ct=time;
                process_queue[largest].waiting_time=process_queue[largest].ct-process_queue[largest].arrival_time-process_queue[largest].burst_time;
                process_queue[largest].turnaround_time=process_queue[largest].ct-process_queue[largest].arrival_time;
                process_queue[largest].status=1;
                wait_time=wait_time+process_queue[largest].waiting_time;
                turnaround_time=turnaround_time+process_queue[largest].turnaround_time;
                printf("\n%c\t\t\t %d\t\t\t %d\t\t\t %d\t\t\t %d\t\t\t %d\n",process_queue[largest].process_name,process_queue[largest].arrival_time,process_queue[large
st].burst_time,process_queue[largest].priority,process_queue[largest].waiting_time,process_queue[largest].turnaround_time);
        }
        average_waiting_time=wait_time/limit;
        average_turnaround_time=turnaround_time/limit;
        printf("\n AVERAGE WAITING TIME:\t%f",average_waiting_time);
        printf("\n AVERAGE TURNAROUND TIME:\t%f\n",average_turnaround_time);
        return 0;
        }

[s2019103562@centos8-linux Mon Apr 26 06:05 PM lab12]$
```