Lab4 HỆ ĐIều HÀNH - this

Hệ điều hành (Trường Đại học Công nghệ thông tin, Đại học Quốc gia Thành phố Hồ Chí Minh)

Sinh viên: Võ Minh Cường
MSSV: 19521303
Lớp: IT007.MMCL.L21.2
LAB4 MÔN HỆ ĐIỀU HÀNH

Trả lời lý thuyết

1. **Vẽ sơ đồ giải thuật của các giải thuật lập lịch tiến trình:**
   - FCFS (First Come First Served)

Tiến trình	Thời điểm vảo	Thời gian thực hiện
P1	0	5
P2	1	3
P3	2	2

P1	P2	P3	
0	5	8	10

   - RR (Round Robin)

Tiến trình	Thời điểm vảo	Thời gian thực hiện
P1	0	5
P2	1	3
P3	2	2

P1	P2	P3	P1	
0	4	7	9	10

   - SJF (Shortest Job First)

Tiến trình	Thời điểm vảo	Thời gian thực hiện
P1	0	5
P2	1	3
P3	2	2

P1	P2	P3	P1	
0	1	4	6	10

   - SRT (Shortest Remain Time)

Tiến trình	Thời điểm vảo	Thời gian thực hiện

P1	0	5
P2	1	3
P3	2	2

P1	P2	P3	P1	
0	1	4	6	10

2. Giải thích các thuật ngữ sau

TT	Thuật ngữ	Mô tả
1	Arrival Time	Thời gian đến
2	Burst Time	Thời gian thực hiện
3	Quantum Time	Định mức thời gian
4	Response Time	Thời gian phản hồi
5	Waiting Time	Thời gian chờ đợi
6	Turnaround Time	Thời gian hoàn thành một tiến trình
7	Average Waiting Time	Thời gian chờ đợi trung bình
8	Average Turnaround Time	Thời gian hoàn thành trung bình của một tiến trình

Ví dụ: Soạn thảo và biên dịch giải thuật FCFS bên dưới:

```c
/*##################################
University of Information Technology
IT007 Operating System
Vo Minh Cuong, 19521303
File: fcfs.c
##################################*/

#include<stdio.h>

void main()
{
        int pn[10];
        int arr[10], bur[10], star[10], finish[10], tat[10], wt[10], i, n;
        int totwt = 0, tottat = 0;
        printf("Enter the number of processes:");
        scanf("%d", &n);
        for (i=0; i<n; i++) {
                printf("Enter Process Name, Arrival Time & Burst Time:");
                scanf("%d%d%d",&pn[i],&arr[i],&bur[i]);
        }

        for (i=0; i<n; i++) {
                if (i==0) {
                        star[i]=arr[i];
                        wt[i]=star[i]-arr[i];
                        finish[i]=star[i]+bur[i];
                        tat[i]=finish[i]-arr[i];
                } else {
                                                                1,1           Top
```

```c
                } else {
                        star[i]=finish[i-1];
                        wt[i]=star[i]-arr[i];
                        finish[i]=star[i]+bur[i];
                        tat[i]=finish[i]-arr[i];
                }
        }
        printf("\nPName Arrtime Burtime Star TAT Finish");
        for (i=0; i<n;i++) {
                printf("\n%d\t%6d\t\t%6d\t%6d\t%6d\t%6d",pn[i],arr[i],bur[i],star
[i], tat[i], finish[i]);
                totwt+=wt[i];
                tottat+=tat[i];
        }
}
                                                                41,1          Bot
```

```
PName Arrtime Burtime Star TAT Finish
0               0               2       0       2       2
1               1               3       2       4       5
2               2               4       5       7       9
```

Bài tập

Bài 1: Viết chương trình mô phỏng giải thuật SJF với các yêu cầu sau:

```c
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include<unistd.h>
#include<sys/wait.h>
#include<signal.h>

    main(){
     arrival_time[10], burst_time[10], temp[10];
     time, smallest, n, i, count=0;
     wait_time=0, turnaround_time =0, end;
       average_waiting_time, average_turnaround_time;
printf("Enter Number of processes: ");
scanf("%d", &n);
for(i=0;i<n;i++){
printf("Enter arrival time for process P%d : ", i+1);
```

```c
printf("Enter burst time for process P%d: ", i+1);
scanf("%d", &burst_time[i]);
temp[i] = burst_time[i];
}
burst_time[9]=9999;
for(time = 0; count!= n; time++)
{
       smallest=9;
       for(i=0;i<n;i++){
          if(arrival_time[i] <= time && burst_time[i] < burst_time[smallest] $
          {
               smallest = i;
          }
       }
       burst_time[smallest]--;
       if(burst_time[smallest] ==0)
       {
         count++;
         end = time + 1;
         wait_time = wait_time + end - arrival_time[smallest]-temp[smallest];
         turnaround_time = turnaround_time + end - arrival_time[smallest];
       }
}
```

```
        }
      }
      burst_time[smallest]--;
      if(burst_time[smallest] ==0)
      {
        count++;
        end = time + 1;
        wait_time = wait_time + end - arrival_time[smallest]-temp[smallest];
        turnaround_time = turnaround_time + end - arrival_time[smallest];
      }
}
average_waiting_time = wait_time/n;
average_turnaround_time = turnaround_time/n;
printf("Average waiting time:%f\n", average_waiting_time);
printf("Average turnaround time:%f\n", average_turnaround_time);
      0;
```

```
Enter Number of processes: 4
Enter arrival time for process P1 : 1
Enter burst time for process P1: 4
Enter arrival time for process P2 : 2
Enter burst time for process P2: 4
Enter arrival time for process P3 : 3
Enter burst time for process P3: 5
Enter arrival time for process P4 : 4
Enter burst time for process P4: 8
Average waiting time:4.000000
Average turnaround time:10.000000
```

Bài 2: Viết chương trình mô phỏng giải thuật SRT với các yêu cầu sau:

```
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include<unistd.h>
#include<sys/wait.h>
#include<signal.h>
#define MAX 15
   main(){
     a[10], b[10], x[10], i,j, smallest, count=0,time, n;
       avg=0,tt=0,end;
 printf("Enter the number of process:\n");
 scanf("%d", &n);
 printf("Enter arrival time\n");
 for(i=0;i<n;i++)
 scanf("%d",&a[i]);
 printf("Enter burst time\n");
```

```
 for(i=0;i<n;i++)
 scanf("%d", &b[i]);
 for(i=0;i<n;i++)
 x[i]=b[i];

 b[9]=9999;

 for(time=0;count!=n;time++)
 {
  smallest=9;
  for(i=0;i<n;i++)
  {
```

```
scanf("%d", &b[i]);
for(i=0;i<n;i++)
x[i]=b[i];

b[9]=9999;

for(time=0;count!=n;time++)
{
 smallest=9;
 for(i=0;i<n;i++)
 {
  if(a[i]<=time && b[i]<b[smallest] && b[i]>0 )
  smallest=i;
 }
 b[smallest]--;
 if(b[smallest]==0)
 {
  count++;
  end=time+1;
  avg=avg+end-a[smallest]-x[smallest];
  tt= tt+end-a[smallest];
```

```
 printf("\n\nAverage waiting time =%lf\n", avg/n);
 printf("Average Turnaround time = %lf\n", tt/n);
        0;
}
```

```
Enter the number of process:
4
Enter arrival time
0
1
2
3
Enter burst time
8
4
9
5


Average waiting time =6.500000
Average Turnaround time = 13.000000
```

Bài 3: Viết chương trình mô phỏng giải thuật RR với các yêu cầu sau (giả sử tất cả các tiến trình đều có arrival time là 0):

```c
#include<stdio.h>
#include<stdlib.h>
#include<pthread.h>
#include<unistd.h>
#include<sys/wait.h>
#include<signal.h>
#define MAX 15
    main(){
     i, n, total=0, x, counter=0,time_quantum;
     wait_time=0, turnaround_time=0,arrival_time[10], burst_time[10], temp[10];
       average_wait_time, average_turnaround_time;
 printf("\nEnter total number of processes: ");
 scanf("%d", &n);
 x=n;
 for(i=0;i<n;i++)
 {
```

```c
for(i=0;i<n;i++)
{
 printf("\nEnter details of process[%d]\n", i+1);
 printf("Arrival time: ");
 scanf("%d", &arrival_time[i]);
 printf("Burst time: ");
 scanf("%d", &burst_time[i]);
 temp[i] = burst_time[i];
}

printf("\nEnter time quantum: ");
scanf("%d", &time_quantum);
printf("\nProcess ID\t\tBurst Time\t Turnaround Time\t Waiting Time\n");
for(total=0,i=0;x!=0;)
```

```
if(temp[i]<=time_quantum && temp[i]>0)
{
 total = total + temp[i];
 temp[i]=0;
 counter=1;
}
else if(temp[i]>0)
{
 temp[i] = temp[i] - time_quantum;
 total = total + time_quantum;
}
if(temp[i] == 0 && counter == 1)
{
 x--;
 printf("\nProcess[%d]\t\t%d\t\t %d\t\t\t %d", i+1, burst_time[i],
 total - arrival_time[i], total - arrival_time[i] - burst_time[i]);
 wait_time = wait_time + total - arrival_time[i] - burst_time[i];
 turnaround_time = turnaround_time + total - arrival_time[i];
 counter = 0;
}
if( i == n-1 )
{
 i=0;
```

```
{
 x--;
 printf("\nProcess[%d]\t\t%d\t\t %d\t\t\t %d", i+1, burst_time[i],
 total - arrival_time[i], total - arrival_time[i] - burst_time[i]);
 wait_time = wait_time + total - arrival_time[i] - burst_time[i];
 turnaround_time = turnaround_time + total - arrival_time[i];
 counter = 0;
}
if( i == n-1 )
{
 i=0;
}
else if(arrival_time[i+1]<=total)
{
 i++;
}
else
{
 i=0;
}

average_wait_time = wait_time * 1.0 /n;
average_turnaround_time = turnaround_time *1.0 /n;
```

```
  else if(arrival_time[i+1]<=total)
  {
   i++;
  }
  else
  {
   i=0;
  }
}
 average_wait_time = wait_time * 1.0 /n;
 average_turnaround_time = turnaround_time *1.0 /n;
 printf("\n\nAverage Waiting time: \t%f", average_wait_time);
 printf("\nAverage Turnaround time:\t%f\n", average_turnaround_time);
        0;
}
```