**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KỸ THUẬT MÁY TÍNH**

**TÀI LIỆU:**

**HƯỚNG DẪN THỰC HÀNH**

**HỆ ĐIỀU HÀNH**

**Nhóm biên soạn:**

- ThS. Phan Đình Duy

- ThS. Phạm Văn Phước

- ThS. Nguyễn Việt Quốc

- KS. Nguyễn Hữu Nhân

- KS. Lê Văn La

- KS. Trần Văn Quang

***Tháng 3 năm 2015***

**NỘI DUNG CÁC BÀI THỰC HÀNH**

**Phần 1: Lập trình trên Linux**

Bài 1: Hướng dẫn cài đặt Ubuntu và các lệnh cơ bản của shell

Bài 2: Cơ bản lập trình shell

**Phần 2: Thực hành hệ điều hành**

Bài 3: Quản lý tiến trình

**Bài 4: Định thời CPU**

Bài 5: Đồng bộ hóa tiến trình, tiểu trình

Bài 6: Quản lý bộ nhớ

**Phần 3: Bài tập lớn**

CÁC PHẦN MỀM THIẾT BỊ SỬ DỤNG TRONG MÔN THỰC HÀNH

- Phần mềm VMware

- Hệ điều hành Ubuntu

**Bài 4:**

**ĐỊNH THỜI CPU**

**Mục tiêu của của buổi thực hành:**

- Sinh viên nắm rõ được các giải thuật : First Come First Served (FCFS), Round Robbin (RR), Shortest Job First (SJF), Priority.

- Chỉ ra được ưu điểm và nhược điểm các giải thuật trên.

- Xây dựng được các chương trình mô phỏng các giải thuật trên.

**I. Một số khái niệm sinh viên cần nắm:**

**1. Khái niệm giờ CPU:**

- CPU là một loại tài nguyên quan trọng của máy tính. Mọi tiến trình muốn hoạt động phải có sự phục vụ của CPU. Thời gian mà CPU phục vụ cho tiến trình hoạt động được gọi là giờ CPU.

- Tại mỗi thời điểm chỉ có một tiến trình được phân phối giờ CPU để hoạt động.

**2. Khái niệm trạng thái của tiến trình:**

Trong chế độ đa chương trình, có ba trạng thái của tiến trình liên quan mật thiết đến giờ CPU bao gồm: Ready, Running và Waiting.

- Sẵn sàng (Ready): Là trạng thái mà tiến trình được phân phối đầy đủ mọi tài nguyên cần thiết và đang chờ giờ CPU.

- Thực hiện (Running): là trạng thái mà tiến trình được phân phối đầy đủ mọi tài nguyên cần thiết và giờ CPU.

- Đợi (Waiting): là trạng thái tiến trình không thực hiện được vì thiếu một vài điều kiện nào đó (đợi dữ liệu vào/ ra, đợi tài nguyên bổ sung...). Khi sự kiện mà nó chờ đợi xuất hiện, tiến trình sẽ quay lại trạng thái sẵn sàng.

- Các tiến trình tuân thủ theo sơ đồ thực hiện sau:

**Bắt đầu**

**Sử dụng CPU**

**Đợi I/O**

**Sử dụng CPU**

**Đợi I/O**

**Sử dụng CPU**

**Kết thúc**

Một tiến trình đang trong trạng thái thực hiện, nó có thể rời khỏi trạng thái bởi một trong ba lý do sau:

- Tiến trình đã hoàn thành công việc, khi đó nó trả lại giờ CPU và chuyển sang chờ xử lý kết thúc.

- Tiến trình tự ngắt: Khi tiến trình chờ đợi một sự kiện nào đó, tiến trình sẽ được chuyển sang trạng thái thực hiện khi có xuất hiện sự kiện nó đang chờ.

- Tiến trình sử dụng hết giờ CPU dành cho nó, khi đó sẽ được chuyển sang trạng thái sẵn sàng.

Việc chuyển tiến trình sang trạng thái sẵn sàng về bản chất là thực hiện việc phân phối lại giờ CPU.

**3. Khái niệm lập lịch cho CPU**

Để điều khiển tiến trình ở nhiều trạng thái khác nhau, hệ thống thường tổ chức các từ trạng thái (thực chất là các khối điều khiển tiến trình) để ghi nhận tình trạng sử dụng tài nguyên và trạng thái tiến trình.

Như vậy lập lịch cho CPU có nghĩa là tổ chức một hàng đợi các tiến trình sẵn sàng để phân phối giờ CPU cho chúng trên độ ưu tiên của các tiến trình; sao cho hiệu suất sử dụng CPU là tối ưu nhất.

Mỗi tiến trình ở trạng thái sẵn sàng sẽ được gắn với một thứ tự ưu tiên. Thứ tự ưu tiên này được xác định dựa ào các yêu tố như: Thời điểm hình thành tiến trình, thời gian thực hiện tiến trình, thời gian kết thúc tiến trình.

**II. BÀI TẬP**

Sinh viên cần chuẩn bị trước ở nhà những công việc sau:

1. Nêu khái niệm, ưu điểm, nhược điểm của các giải thuật: First Come First Served (FCFS), Round Robbin (RR), Shortest Job First (SJF), Shortest Remain Time (SRT).

2. Vẽ sơ đồ khối hoạt động của các giải thuật định thời trên.

3. Chạy thử chương trình mô phỏng giải thuật FCFS sau:

#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 the 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

{

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 Start 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];

}

}

Yêu cầu:

- Thực thi đoạn chương trình trên : Nhập dữ liệu input và xuất các giải trị ouput ra màn hình.

- Vẽ sơ đồ giải thuật và giải thích ý nghĩa đoạn code

- Thêm vào đoạn code trên hàm sắp xếp lại Arrival Time của các process trong trường hợp người nhập nhập vào thời gian đến của các process không theo thứ tự.

- Thêm vào đoạn code trên để tính Average waiting time và Aveage Turn Around time

Hướng dẫn tạo và biên dịch file c trên Linux:

* Để lập trình C trên Linux cần cài đặt thêm thư viện theo lệnh sau:

C:\Users\Mr LA\Desktop\3-13-2015 8-13-12 PM.png

*Sinh viên kiểm tra xem thư viện trên dùng làm gì, có cần cài đặt không?*

* Cách tạo và biên dịch file C trên Linux (Sử dụng chương trình gedit và terminal).
* Tạo file c:

Bạn tạo 1 file tên là FCFS.c bằng cách click chuột phải vào Desktop -> new document -> empty document mở nó ra (mặc định là mở bằng gedit trên Ubuntu) và viết chương trình với nội dung ở trên.

* Biên dịch file c:

Thực hiện lệnh biên dịch như bên dưới:



Nếu biên dịch thành công thì nó không báo gì cả và trên Desktop của các bạn xuất hiện thêm một file là FCFS.

* Để chạy chương trình thực hiện lệnh sau:



* Kết quả sau khi chạy:

4. Viết chương trình mô phỏng giải thuật định thời còn lại với các yêu cầu input và output sau:

**- Với giải thuật SJF, Priority :**

*Input:*

Nhập: Số Process

Nhập: Process Name, Arrival Time & Burst Time (Đối với Priority nhập thêm Priority)

*Output:*

Với mỗi process: Process Name, Arrival time, Execution time, Waiting time, Turn Around time.

Với toàn bộ process: Average Waiting Time, Average Turn Around Time

**- Với giải thuật RR:**

*Input:*

Nhập: Số Process

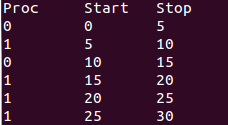
Nhập: Thời gian quantum

Nhập: Tên process & Burst time

Giả sử Arrival time của tất cả process bằng 0, Dispatch latency bằng 0.

*Output:*

* In biểu đồ Gantt với các thông số Process Name, StartCPUTime, StopCPUTime. VD như hình dưới với trường hợp Process 0 và 1 có Burst Time là 10, 20 và quantum Time của RR queue là 5.



* In kết quả Average Waiting Time và Average Turn Around Time.