

BAN HỌC TẬP CÔNG NGHỆ PHẦN MỀM

TRAINING GIỮA KỲ HỌC KỲ I NĂM HỌC 2022 – 2023



Sharing is learning



 **BAN HỌC TẬP**

Khoa Công nghệ Phần mềm

Trường Đại học Công nghệ Thông tin

Đại học Quốc gia thành phố Hồ Chí Minh

 **CONTACT**


bht.cnpm.uit@gmail.com


fb.com/bhtcnpm


fb.com/groups/bht.cnpm.uit

TRAINING

HỆ ĐIỀU HÀNH

 **Thời gian:** 19h30 thứ 5 ngày 03/11/2022

 **Địa điểm:** Microsoft Teams

 **Trainers:**
Bùi Văn Thi – MTCL2021
Nguyễn Hà Mi – KTPM 2021



Sharing is learning

TRAINING

Hệ Điều Hành



Sharing is learning

Chương I: Tổng quan về hệ điều hành

Chương II: Cấu trúc hệ điều hành

Chương III: Tiến trình

Chương IV: Định thời CPU

TRAINING

Hệ Điều Hành

**Chương I: Tổng quan
về hệ điều hành**

1. Tổng quan cơ bản



Sharing is learning

1. Tổng quan cơ bản

Khái niệm: là chương trình trung gian giữa phần cứng máy tính và người sử dụng, có chức năng điều khiển và phối hợp việc sử dụng phần cứng và cung cấp các dịch vụ cơ bản cho các ứng dụng.

Mục tiêu:

- Giúp người dùng dễ dàng sử dụng hệ thống.
- Quản lý và cấp phát tài nguyên hệ thống một cách hiệu quả.



Sharing is learning

1. Tổng quan cơ bản

Lợi ích:

- Quản lý phần cứng máy tính
- Cung cấp giao diện cho người dùng
- Là nơi để người dùng cài đặt các chương trình ứng dụng
- Kết nối các thiết bị phần cứng với nhau
- Tương tác giữa các chương trình với nhau và với phần cứng



Sharing is learning

1. Tổng quan cơ bản

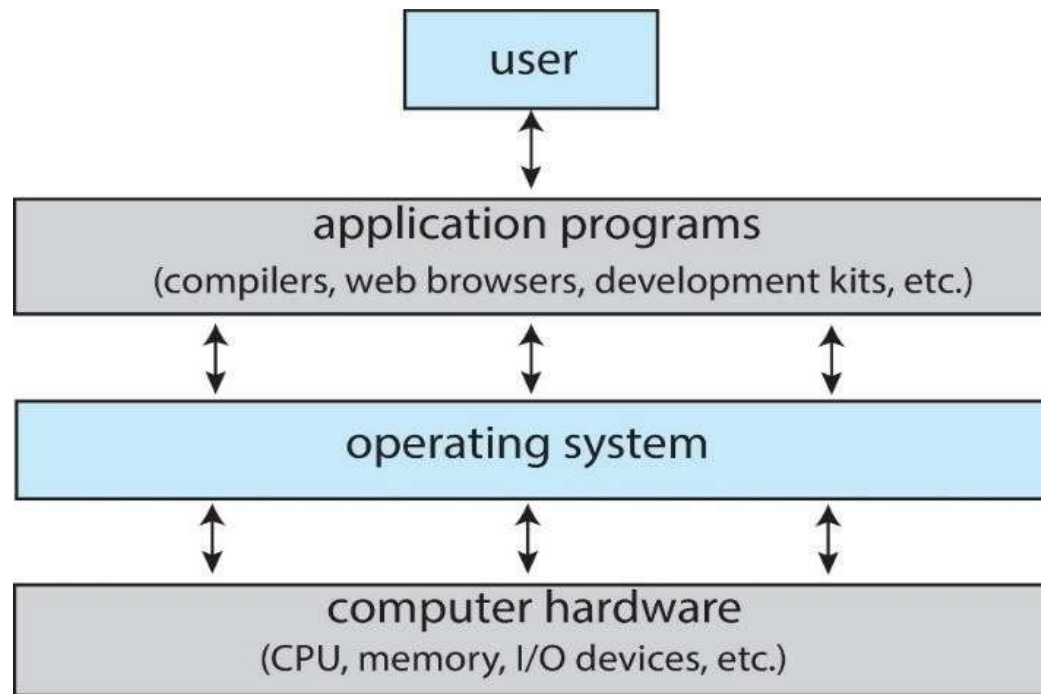
Chức năng:

- Phân chia thời gian xử lý và định thời CPU
- Phối hợp và đồng bộ hoạt động giữa các processes
- Quản lý tài nguyên hệ thống (Thiết bị I/O, bộ nhớ, file chứa dữ liệu,...)
- Kiểm soát truy cập, bảo vệ hệ thống
- Duy trì sự nhất quán của hệ thống, kiểm soát lỗi và phục hồi hệ thống khi có lỗi
- Cung cấp giao diện làm việc cho người dùng



Sharing is learning

1. Tổng quan cơ bản



- ➡ Users(people, machines, other computers)
Sử dụng hệ thống tài nguyên để giải quyết một bài toán tính toán nào đó của user
- ➡ VD: compliers, database systems, video games,...
- ➡ Hệ điều hành: Phân phối tài nguyên điều khiển và phối hợp các hoạt động của các chương trình trong hệ thống
- ➡ Bao gồm các tài nguyên cơ bản của máy tính như CPU, bộ nhớ, các thiết bị I/O



Sharing is learning

1. Tổng quan cơ bản

1. Phát biểu nào dưới đây KHÔNG ĐÚNG về Hệ điều hành?
 - A. Hệ điều hành quản lý các phần cứng máy tính.
 - B. Hệ điều hành trực tiếp điều khiển hoạt động cho từng thiết bị phần cứng.**
 - C. Hệ điều hành hỗ trợ phần mềm giao tiếp phần cứng trên máy tính.
 - D. Hệ điều hành hỗ trợ người dùng điều hành máy tính.
2. Điều gì là ĐÚNG khi một máy tính không có Hệ điều hành?
 - A. Các ứng dụng vẫn chạy bình thường trên máy tính đó.
 - B. CPU vẫn tiếp nhận và thực thi các lệnh từ người dùng.
 - C. Các ứng dụng và lệnh người dùng không thể thực thi trên máy tính.**
 - D. Người dùng vẫn cài đặt phần mềm vào máy tính như bình thường.



Sharing is learning

1. Tổng quan cơ bản

3. Chọn phát biểu **SAI** về hệ điều hành?

A. Hệ điều hành là chương trình trung gian giữa phần cứng máy tính và người sử dụng.

B. Hệ điều hành cung cấp các dịch vụ cơ bản cho các ứng dụng.

☒ C. Hệ điều hành sử dụng hệ thống tài nguyên để giải quyết một bài toán nào đó của người sử dụng.

D. Hệ điều hành có chức năng có chức năng điều khiển và phối hợp việc sử dụng phần cứng.

4. Trong các thành phần của hệ thống máy tính, thành phần nào trực tiếp quản lý các tài nguyên phần cứng:

A. Người dùng

B. Các phần mềm

☒ C. Hệ điều hành

D. Dữ liệu



Sharing is learning

1. Tổng quan cơ bản

5. Trong phân lớp hệ thống máy tính, Hệ điều hành thuộc vị trí nào
- A. Hệ điều hành thuộc lớp cuối cùng, kế trên là lớp phần cứng.
 - B. Hệ điều hành thuộc lớp trên cùng, kế dưới là lớp ứng dụng.
 - ☒ C. Hệ điều hành nằm giữa lớp phần cứng và lớp ứng dụng.
 - D. Hệ điều hành nằm giữa lớp phần cứng và lớp người dùng.
6. Dưới góc độ cơ bản, Hệ điều hành được định nghĩa là:
- A. là một phần mềm chạy trên máy tính
 - ☒ B. là một chương trình quản lý phần cứng máy tính.
 - C. là một chương trình bảo vệ phần cứng máy tính
 - D. là một phần mềm quản lý các phần mềm khác.



Sharing is learning

TRAINING

Hệ Điều Hành

Chương I: Tổng quan
về hệ điều hành

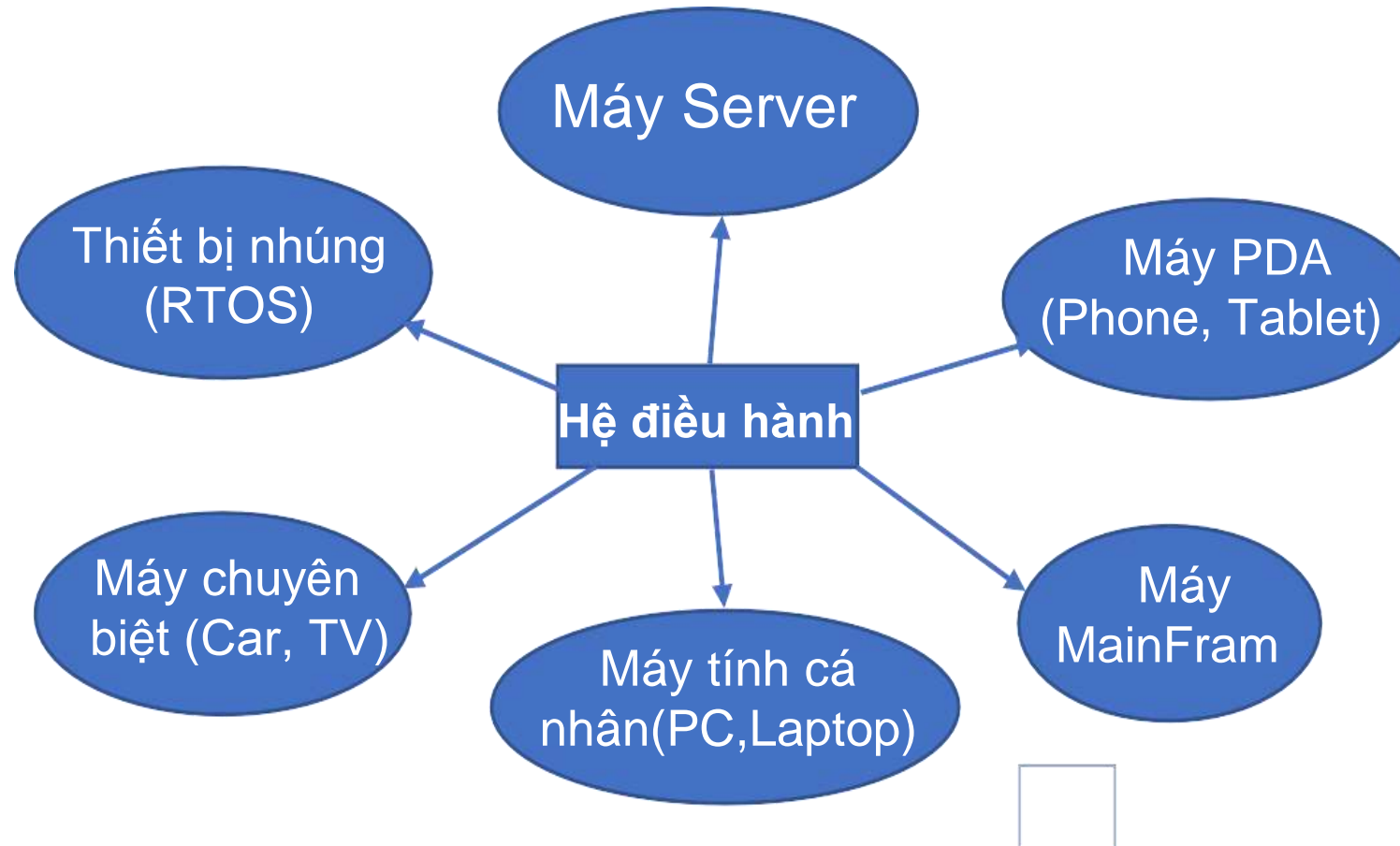
2. Phân loại hệ điều hành



Sharing is learning

2. Phân loại hệ điều hành

Dưới góc độ loại máy tính



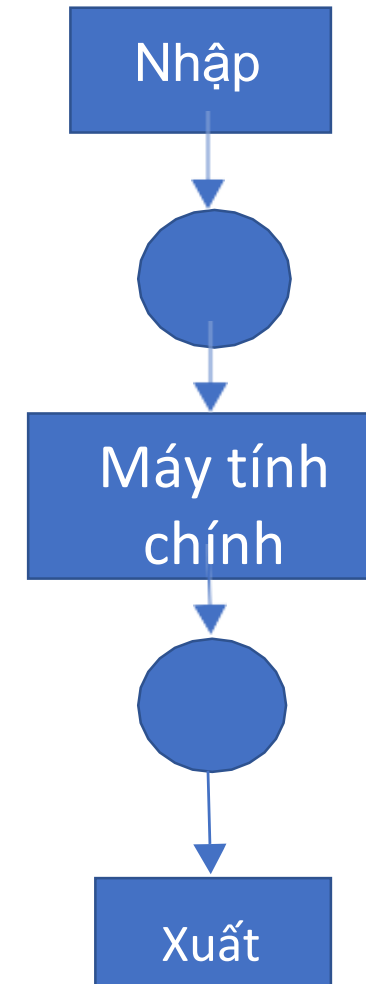
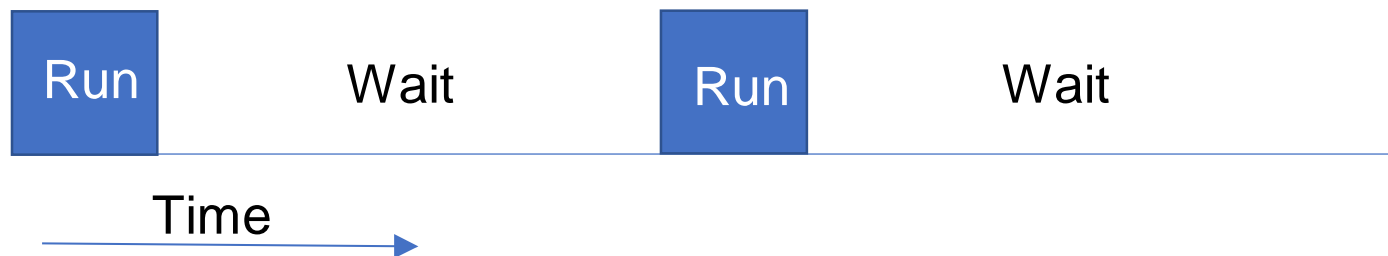
Sharing is learning

2. Phân loại hệ điều hành

Hệ thống xử lý theo chương trình

Hệ thống đơn chương (uniprograming OS)

- Tác vụ được thi hành tuần tự
- Bộ giám sát thường trực
- CPU và các thao tác nhập xuất



Sharing is learning

2. Phân loại hệ điều hành

Hệ thống đa chương

- Nhiều công việc được nạp đồng thời vào bộ nhớ chính
- Khi một tiến trình thực hiện I/O, một tiến trình khác được thực thi



Sharing is learning

2. Phân loại hệ điều hành

Hệ thống đa chương

Yêu cầu đối với hệ điều hành:

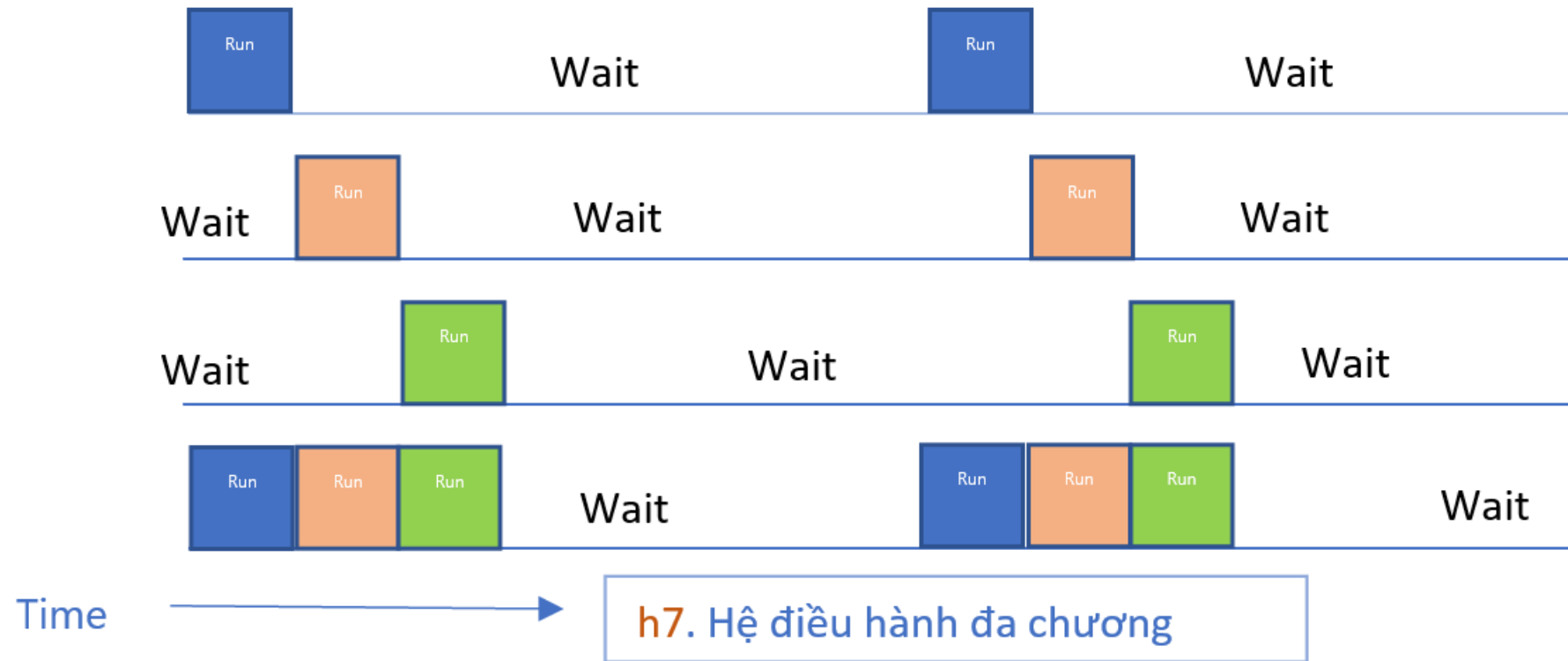
- Định thời công việc
- Quản lý bộ nhớ
- Định thời CPU
- Cấp phát tài nguyên (đĩa, máy in,...)
- Bảo vệ



Sharing is learning

2. Phân loại hệ điều hành

Hệ thống đa chương



Sharing is learning

2. Phân loại hệ điều hành

1. Đây là ưu điểm chính của Hệ thống xử lý đa chương (multiprogramming system)?
 - A. Chương trình khi nạp vào bộ nhớ sẽ được xử lý hoàn thành ngay lập tức.
 - B. Hệ thống chạy được nhiều chương trình cùng lúc.**
 - C. Không cần thiết lập định thời công việc (job scheduling) và quản lý bộ nhớ.
 - D. Tối ưu sử dụng bộ nhớ.
2. Mục đích chính của Hệ thống xử lý đa chương (multiprogramming system) là gì?
 - A. Thực hiện đồng thời nhiều chương trình.
 - B. Tận dụng thời gian nhàn rỗi của CPU.**
 - C. Chia sẻ thời gian giữa các chương trình.
 - D. Tận dụng RAM, ROM khi đọc ghi.



Sharing is learning

2. Phân loại hệ điều hành

Hệ thống chia sẻ thời gian

Mỗi công việc chạy trong khoảng thời gian nhất định

Các yêu cầu: tương tự hệ thống đa chương

- Quản lý các quá trình
 - Đồng bộ giữa các công việc
 - Giao tiếp giữa các công việc
 - Tránh deadlock
- Quản lý hệ thống file, hệ thống lưu trữ



Sharing is learning

2. Phân loại hệ điều hành

Phát biểu nào sau đây KHÔNG ĐÚNG với hệ thống chia sẻ thời gian (time-sharing)?

- A. time-sharing là một hệ thống đa nhiệm (multi-tasking).
- B. time-sharing yêu cầu thời gian chuyển đổi giữa các tác vụ rất ngắn.
- C. time-sharing yêu cầu phải định thời CPU.
- D.** time-sharing yêu cầu hoàn thành xong *nhiệm vụ 1* mới chia sẻ cho *nhiệm vụ 2*.



Sharing is learning

2. Phân loại hệ điều hành

1. Trong hệ thống xử lý đa nhiệm (*multitasking*), việc chuyển đổi giữa các công việc diễn ra:
- A. Sau một khoảng thời gian tùy theo công việc yêu cầu.
 - B. Chuyển đổi khi có công việc khác cần xử lý.
 - C. Luân phiên xoay vòng hoàn thành từng công việc.
 - D. Luân phiên xoay vòng, không đợi công việc hoàn thành.**



Sharing is learning

2. Phân loại hệ điều hành

Hệ thống song song

- Nhiều bộ xử lý cùng chia sẻ một bộ nhớ
- Ưu điểm:
 - Năng suất cao (do nhiều công việc được xử lý đồng thời)
 - Sự hỏng hóc của một bộ xử lý không ảnh hưởng đến toàn bộ hệ thống



Sharing is learning

2. Phân loại hệ điều hành

1. Hệ thống song song được phân loại như thế nào?

A. Đa xử lý đối xứng và bất đối xứng.

B. Đơn chương và đa chương.

C. Client-server và peer-to-peer

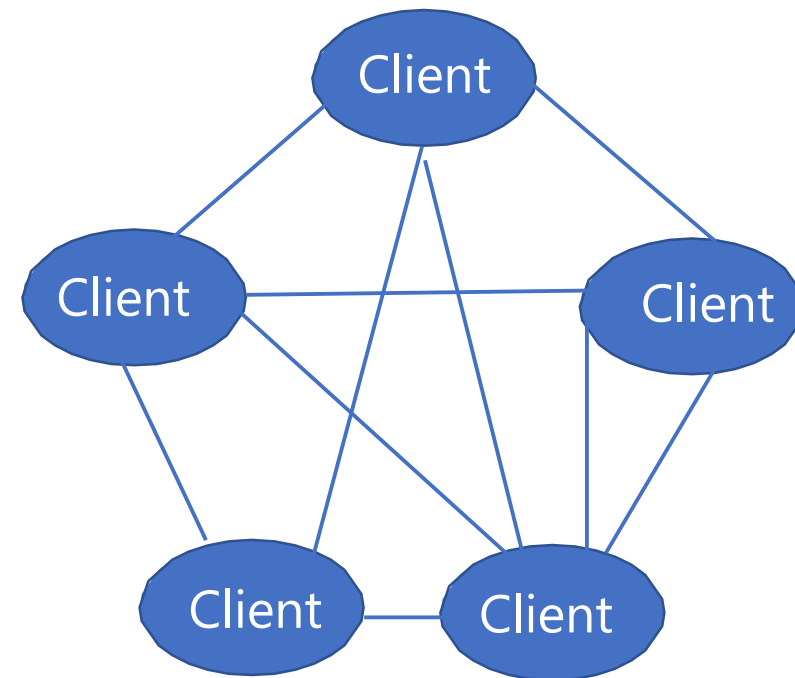
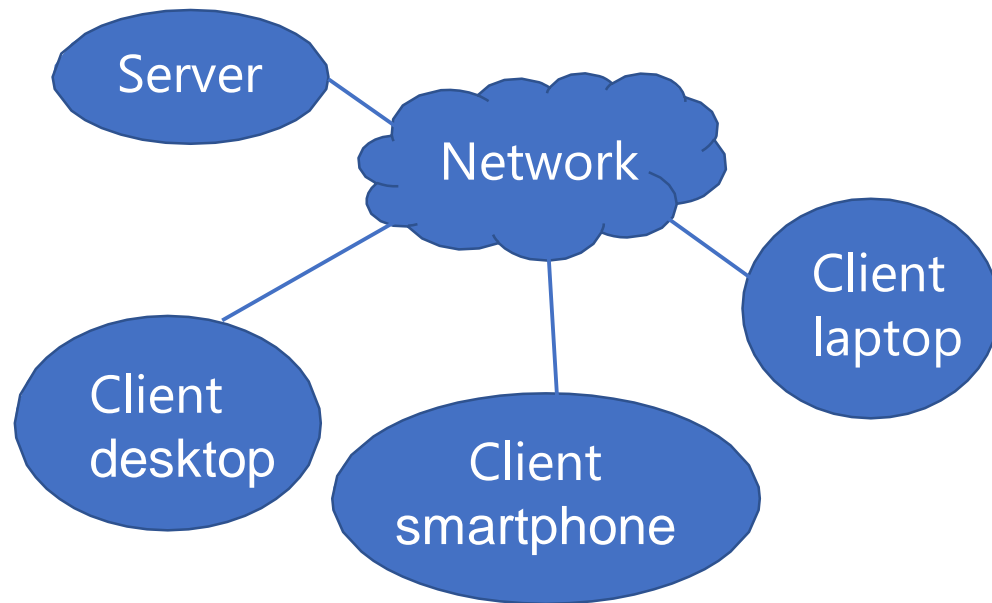
D. Hard real-time và soft real-time.



Sharing is learning

2. Phân loại hệ điều hành

Hệ thống phân tán



Sharing is learning

2. Phân loại hệ điều hành

1. Hệ thống xử lý phân tán có đặc điểm:

- A.** Mỗi bộ xử lý có bộ nhớ riêng.
- B. Các bộ xử lý độc lập không liên hệ nhau.
- C. Một công việc chia đều cho các bộ xử lý.
- D. Dùng chung bộ nhớ kết nối thành mạng.

2. Hệ thống xử lý phân tán được phân loại:

- A. Đồng bộ và bất đồng bộ.
- B.** Peer-to-peer và client-server.
- C. Kết hợp và không kết hợp.
- D. Đối xứng và bất đối xứng.



Sharing is learning

2. Phân loại hệ điều hành

Hệ thống nhúng thời gian thực

- Hệ thống **cho kết quả chính xác trong thời gian nhanh nhất**
- Hard real-time:
 - Yêu cầu thời gian xử lý, **đáp ứng nghiêm ngặt**
 - Giới hạn về độ nhớ
- Soft real-time:
- Các công việc thực hiện theo **độ ưu tiên**

TRAINING

Hệ Điều Hành

Chương II: Cấu trúc hệ điều hành

1. Các thành phần của hệ điều hành



Sharing is learning

1. Các thành phần của hệ điều hành

1. Quản lý tiến trình
2. Quản lý bộ nhớ chính
3. Quản lý file
4. Quản lý hệ thống I/O
5. Quản lý hệ thống lưu trữ thứ cấp
6. Hệ thống bảo vệ
7. Hệ thống thông dịch lệnh

1. Các thành phần của hệ điều hành

Quản lý tiến trình

1. Để hoàn thành công việc, một tiến trình cần:
CPU, Bộ nhớ, File, Thiết bị I/O,..
2. Các nhiệm vụ chính:
 - Tạo và hủy tiến trình
 - Tạm dừng/ thực thi tiếp tiến trình
 - HĐH cung cấp các cơ chế:
 - Đồng bộ hoạt động các tiến trình
 - Giao tiếp giữa các tiến trình
 - Khống chế tắc nghẽn



Sharing is learning

1. Các thành phần của hệ điều hành

Quản lý bộ nhớ chính

1. Bộ nhớ chính là trung tâm của các thao tác, xử lý
2. Hệ điều hành cần quản lý bộ nhớ đã cấp phát cho mỗi tiến trình để tránh xung đột
3. Nhiệm vụ:
 - Quản lý các vùng nhớ trống và đã cấp phát
 - Quyết định sẽ nạp chương trình nào khi có vùng nhớ trống
 - Cấp phát bộ nhớ và thu hồi bộ nhớ



Sharing is learning

1. Các thành phần của hệ điều hành

Quản lý file

1. Hệ thống file:

- File
- Thư mục

2. Dịch vụ chính:

- Tạo và xóa file/ thư mục
- Các thao tác xử lý file/ thư mục
- "Ánh xạ" file/ thư mục vào thiết bị thứ cấp tương ứng
- Sao lưu và phục hồi dữ liệu



Sharing is learning

1. Các thành phần của hệ điều hành

Quản lý hệ thống I/O

1. Che dấu khác biệt của các thiết bị I/O trước người dùng
2. Chức năng:
 - ❖ Cơ chế: buffering, caching, spooling
 - ❖ Cung cấp giao diện chung đến các trình điều khiển thiết bị
 - ❖ Bộ điều khiển các thiết bị phần cứng



Sharing is learning

1. Các thành phần của hệ điều hành

Quản lý hệ thống lưu trữ thứ cấp

1. Lưu trữ bền vững các dữ liệu, chương trình (*vì bộ nhớ chính: kích thước nhỏ & môi trường chứa thông tin không bền vững*)
2. Phương tiện lưu trữ thường là đĩa từ, đĩa quang(HDD,SDD)
3. Nhiệm vụ hệ điều hành trong quản lý đĩa:
 - Quản lý không gian trống trên đĩa
 - Cấp phát không gian lưu trữ
 - Định thời hoạt động cho đĩa



Sharing is learning

1. Các thành phần của hệ điều hành

Hệ thống bảo vệ

1. Nhiệm vụ:

- Cung cấp cơ chế kiểm soát đăng nhập/xuất
- Phân định được sự truy cập tài nguyên hợp pháp và bất hợp pháp
- Phương tiện thi hành các chính sách



Sharing is learning

1. Các thành phần của hệ điều hành

Hệ thống thông dịch lệnh

1. Là giao diện chủ yếu giữa người dùng và OS
2. Khi user login
 - Command line interpreter (shell) chạy, chờ nhận kết quả từ người dùng, thực thi lệnh và trả kết quả về
 - Các lệnh => Bộ điều khiển lệnh => Hệ điều hành



Sharing is learning

1. Các thành phần của hệ điều hành

Hệ thống thông dịch lệnh

- Các lệnh chủ yếu:
 - Tạo, hủy và quản lý tiến trình, hệ thống
 - Kiểm soát I/O
 - Quản lý bộ lưu trữ thứ cấp
 - Quản lý bộ nhớ chính
 - Truy cập hệ thống file và cơ chế bảo mật



Sharing is learning

1. Các thành phần của hệ điều hành

1. Hệ thống thông dịch lệnh là giao diện giữa hệ điều hành và ...?

- A** Người dùng
- B. Các phần mềm
- C. Dữ liệu
- D. Bộ nhớ



Sharing is learning

1. Các thành phần của hệ điều hành

2. *Hệ thống bảo vệ có nhiệm vụ gì?*

- A. Cung cấp cơ chế kiểm soát đăng xuất, nhập xuất
- B. Phân định được sự truy cập tài nguyên hợp pháp và bất hợp pháp
- C. Phương tiện thi hành các chính sách
- ☒ D. Tất cả các phương án trên



Sharing is learning

1. Các thành phần của hệ điều hành

3. Một trong nhiệm vụ của bộ nhớ chính là?

A. Phân định được sự truy cập tài nguyên hợp pháp và bất hợp pháp

B. Quản lý không gian trống trên đĩa

C Quyết định sẽ nạp chương trình nào khi có vùng nhớ trống

D. Tạm dừng/ thực thi tiếp tiến trình



Sharing is learning

2. Các dịch vụ hệ điều hành cung cấp

Kể các dịch vụ hệ điều hành cung cấp?

- Thực thi chương trình
- Thực hiện các thao tác I/O theo yêu cầu của chương trình
- Các thao tác trên hệ thống file
- Trao đổi thông tin giữa các tiến trình qua chia sẻ bộ nhớ và chuyển thông điệp
- Phát hiện lỗi
- Ngoài ra còn có các dịch vụ tăng hiệu suất hệ thống:
 - + Cấp phát tài nguyên
 - + Kế toán
 - + Bảo vệ
 - + An ninh



Sharing is learning

3. Lời gọi hệ thống

- Giao tiếp giữa tiến trình và hệ điều hành
- 3 phương pháp truyền tham số khi sử dụng system call:
 - Qua thanh ghi
 - Qua một vùng nhớ, địa chỉ của vùng nhớ được gửi đến hệ điều hành qua thanh ghi
 - Qua stack



Sharing is learning

3. Lời gọi hệ thống

Khi nào cần sử dụng đến System call (Lời gọi hệ thống)?

- A. Khi một người dùng yêu cầu dịch vụ nào đó từ Kernel của Hệ điều hành.
- B. Khi một chương trình yêu cầu dịch vụ nào đó từ Kernel của Hệ điều hành.**
- C. Khi Hệ điều hành cần trợ giúp từ chương trình.
- D. Khi Hệ điều hành cần trợ giúp từ người dùng.



Sharing is learning

4. Các chương trình hệ thống

Các chương trình hệ thống gồm:

- Quản lý hệ thống file: create, delete, rename, list
- Thông tin trạng thái: date,time,dung lượng bộ nhớ trống
- Soạn thảo file: file editor
- Hỗ trợ ngôn ngữ lập trình: compiler, assembler, interpreter
- Nạp, thực thi, giúp tìm lỗi chương trình: loader, debugger
- Giao tiếp: email,talk, web browser,..



Sharing is learning

5. Cấu trúc hệ thống

Cấu trúc Monolithic-Original UNIX

- UNIX giới hạn về chức năng phần cứng nên Original UNIX cũng có cấu trúc rất giới hạn
- UNIX gồm 2 phần tách rời:
 - Nhân: Cung cấp file system, CPU scheduling,...
 - System program
 - **Đại diện là MS-DOS**

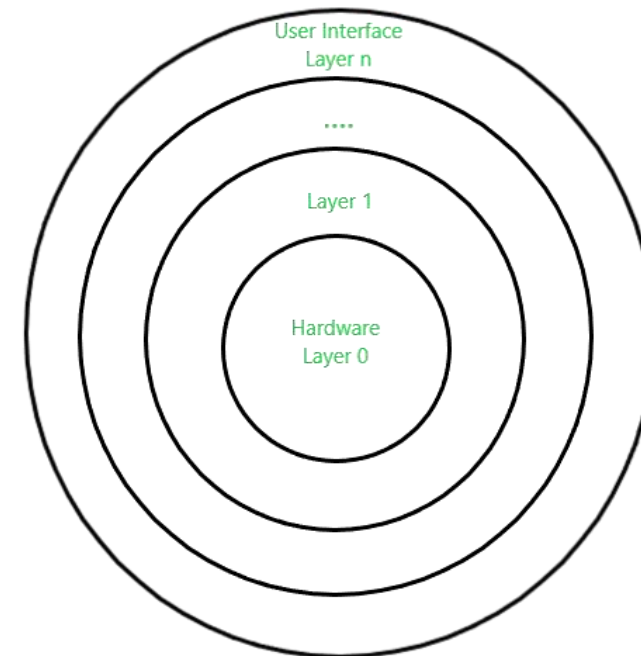


Sharing is learning

5. Cấu trúc hệ thống

Cấu trúc Layered Approach

- Hệ điều hành được chia thành nhiều lớp
- Mỗi lớp tương đương một đối tượng trừu tượng: cấu trúc dữ liệu + thao tác
- Lợi ích của phân lớp:
 - + Gỡ rối
 - + Kiểm tra hệ thống
 - + Thay đổi chức năng
- **Đại diện: THE**

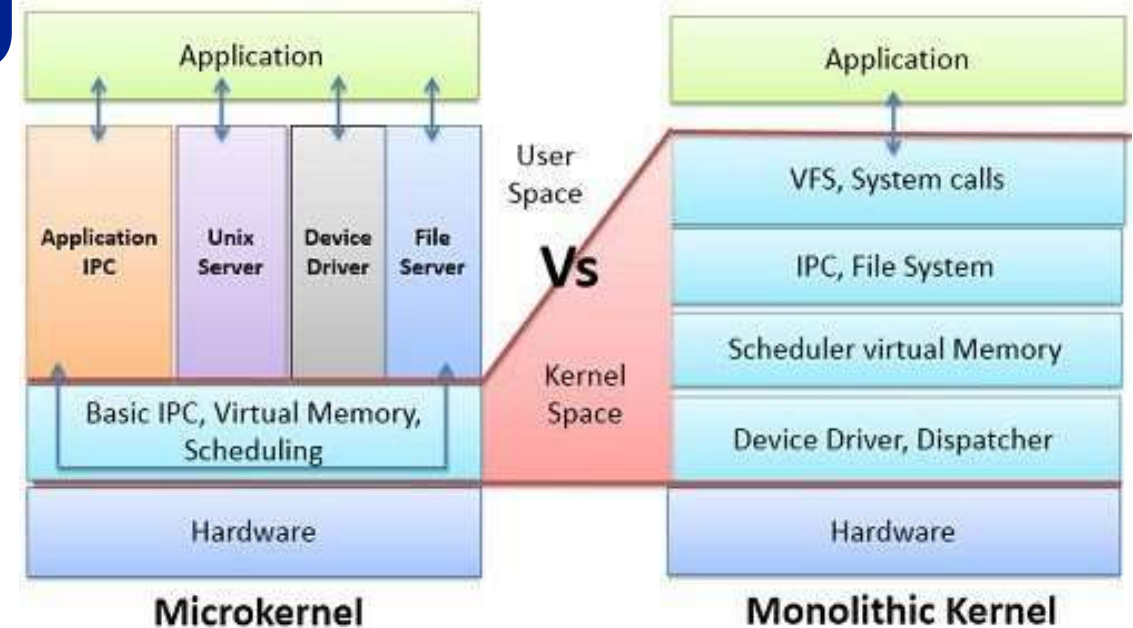


Sharing is learning

5. Cấu trúc hệ thống

Cấu trúc Microkernels

- Phân chia module theo microkernel
- Chuyển một số chức năng của OS từ kernel space sang user space
- Thu gọn kernel => microkernel
- Microkernels có chức năng tối thiểu:
 - Quản lý tiến trình, bộ nhớ
 - Cơ chế giao tiếp giữa các tiến trình



Sharing is learning

5. Cấu trúc hệ thống

Cấu trúc Modules

Nhiều hệ điều hành hiện đại triển khai các loadable kernel modules(LKMs)

- Sử dụng cách tiếp cận hướng đối tượng
- Mỗi core thành phần là tách biệt nhau
- Trao đổi thông qua các interfaces
- Mỗi module như là một phần của nhân

Cấu trúc Modules giống với cấu trúc Layer nhưng phức tạp hơn

VD: Linux, Solaris



Sharing is learning

5. Cấu trúc hệ thống

Cấu trúc Hybrid System

Hầu hết các hệ điều hành hiện đại không theo một cấu trúc thuần túy nào mà lai giữa các cấu trúc với nhau

- Cấu trúc lai là sự kết hợp nhiều cách tiếp cận để giải quyết các nhu cầu về hiệu suất, bảo mật, nhu cầu sử dụng
- Nhân Linux và Solaris theo cấu trúc kết hợp không gian địa chỉ kernel, cấu trúc monolithic và modules
- Nhân Windows hầu như theo cấu trúc liên khối, cộng với cấu trúc vi nhân cho các hệ thống cá nhân khác nhau



Sharing is learning

5. Cấu trúc hệ thống

1. Cho biết tên gọi của kiến trúc Hệ điều hành mà tất cả các modules chức năng của nó được gom hết vào Kernel.
- A. Simple OS.
 - B. Monolithic OS.**
 - C. Layered OS.
 - D. Microkernel OS.



Sharing is learning

5. Cấu trúc hệ thống

2. Cho biết tên gọi của kiến trúc Hệ điều hành mà các modules chức năng của nó được phân chia thành từng lớp giao tiếp với Kernel.

A. Simple OS.

B. Monolithic OS.

☒ C. Layered OS.

D. Microkernel OS.

3. Cho biết tên gọi của kiến trúc Hệ điều hành mà hầu hết các modules chức năng của nó được tách ra ngoài? Kernel chỉ có 2 chức năng chính: quản lý bộ nhớ và liên lạc giữa các tiến trình

A. Simple OS.

B. Monolithic OS.

C. Layered OS.

☒ D. Microkernel OS.

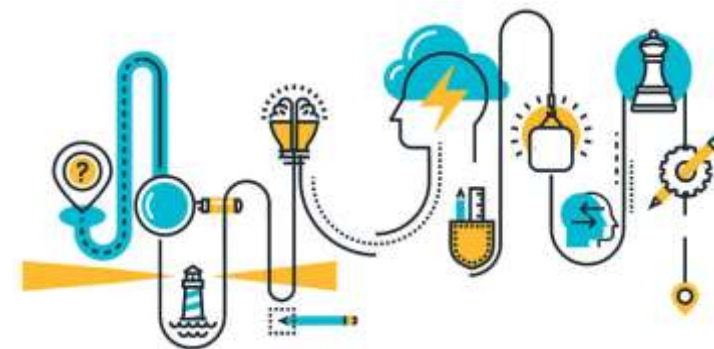


Sharing is learning

TRAINING

Hệ Điều Hành

Chương III: Tiến trình



Sharing is learning



TIẾN TRÌNH

1. Các khái niệm
2. Trạng thái tiến trình
3. PCB (Khối điều khiển tiến trình)
4. Định thời tiến trình
5. Các tác vụ đối với tiến trình



Sharing is learning

TIẾN TRÌNH

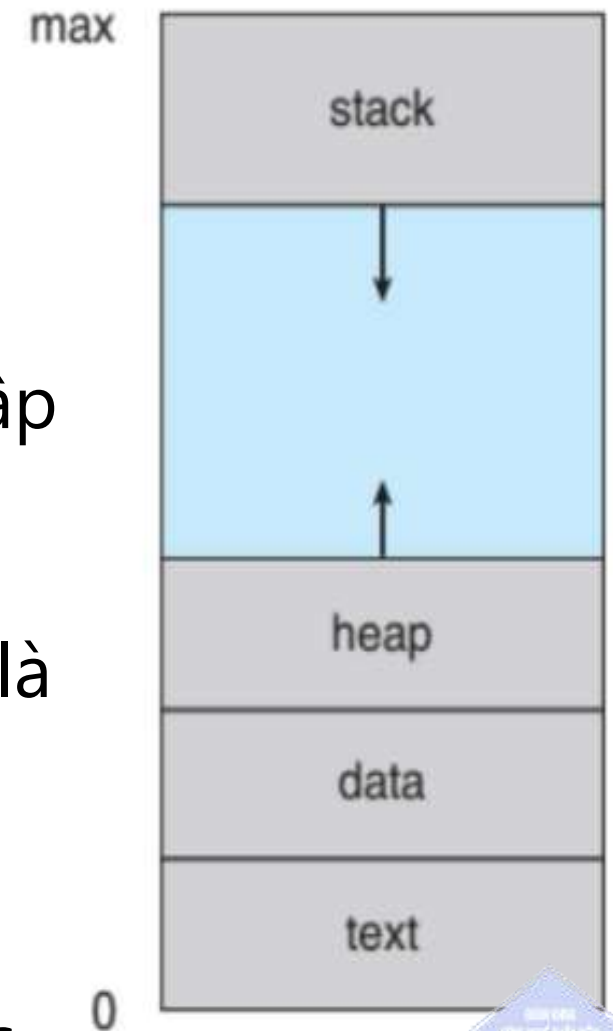
CÁC KHÁI NIỆM CƠ BẢN

Tiến trình là một chương trình đang thực thi

- Chương trình là thực thể bị động lưu trên đĩa (tập tin thực thi)
- Tiến trình là thực thể chủ động
- Khi tập tin thực thi được nạp vào bộ nhớ thì đó là một tiến trình

Một tiến trình bao gồm:

- Text section (program code)
- Data section (chứa global variables)
- Program counter, processor registers
- Heap section (chứa bộ nhớ cấp phát động)
- Stack section (chứa dữ liệu tạm thời)



TIẾN TRÌNH

CÁC KHÁI NIỆM CƠ BẢN

Các bước khởi tạo tiến trình

1

- Cấp phát một định danh duy nhất cho tiến trình

2

- Cấp không gian để nạp tiến trình

3

- Khởi tạo khối dữ liệu PCB cho tiến trình

4

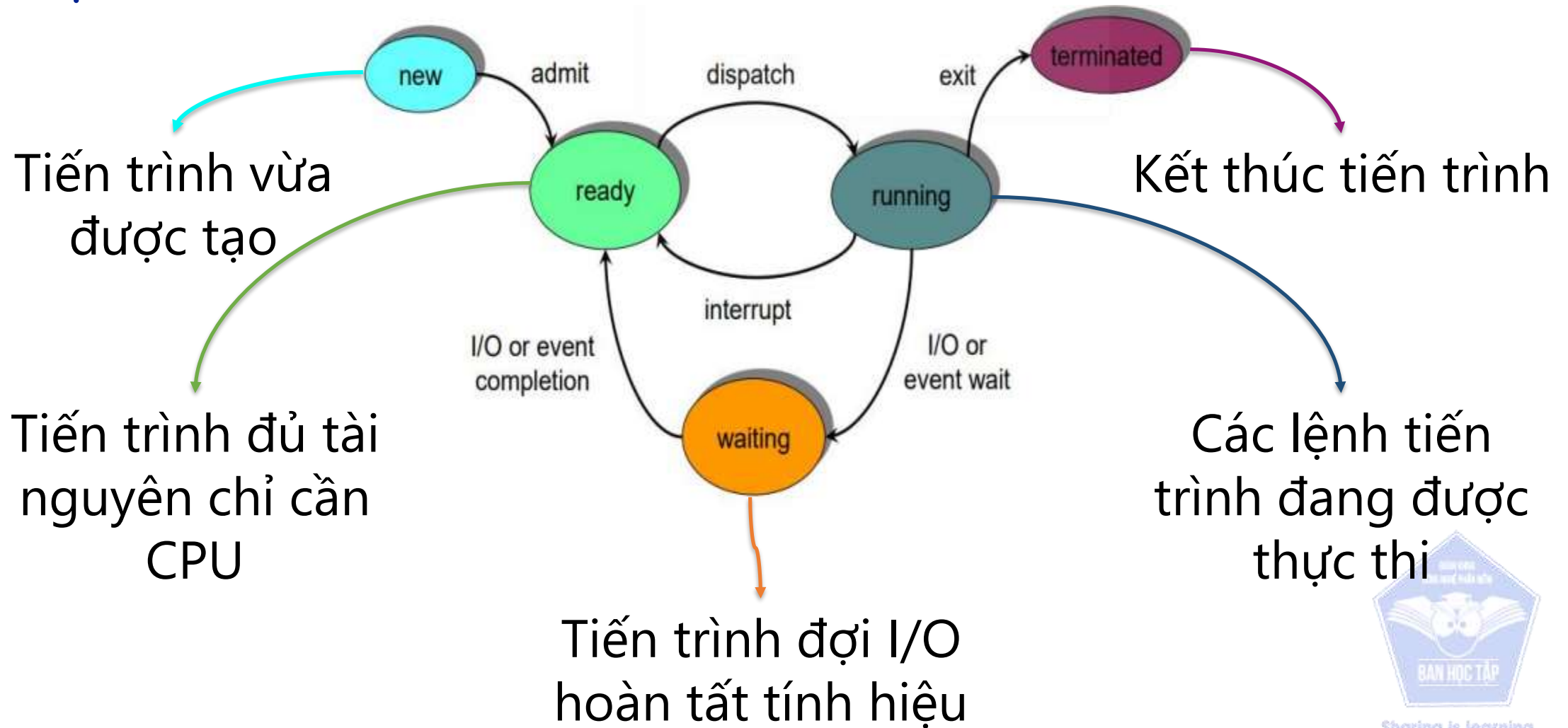
- Thiết lập các mối liên hệ cần thiết



Sharing is learning

TIẾN TRÌNH

TRẠNG THÁI TIẾN TRÌNH



TIẾN TRÌNH

TRẠNG THÁI TIẾN TRÌNH

Ví Dụ

```
#include <stdio.h>
void main() {
    printf(" BHT CONG NGHE PHAN MEM ");
    printf(" Chuc cac ban thi tot ");
    eixt(0);
}
```



Khi chương trình trên chạy thì trải qua những tiến trình nào ?

Hướng dẫn làm

New – Ready – Running – Waiting
– Ready – Running – Wating –
Ready – Running – Terminated



Sharing is learning

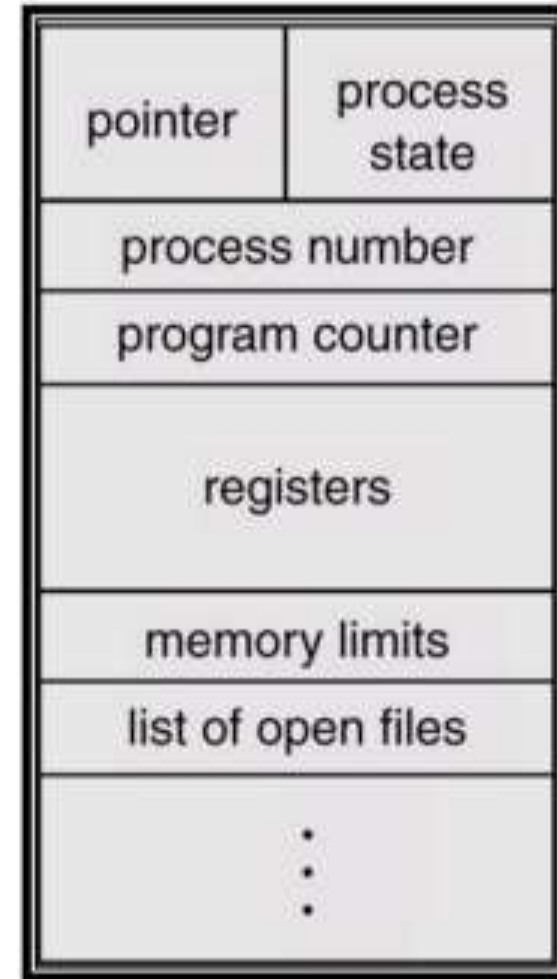
TIẾN TRÌNH

PROCESS CONTROL BLOCK (PCB)

KHỐI ĐIỀU KHIỂN TIẾN TRÌNH

PCB là một trong các cấu trúc dữ liệu quan trọng nhất của hệ điều hành

- Mỗi tiến trình trong hệ thống đều được cấp phát một PCB



Sharing is learning

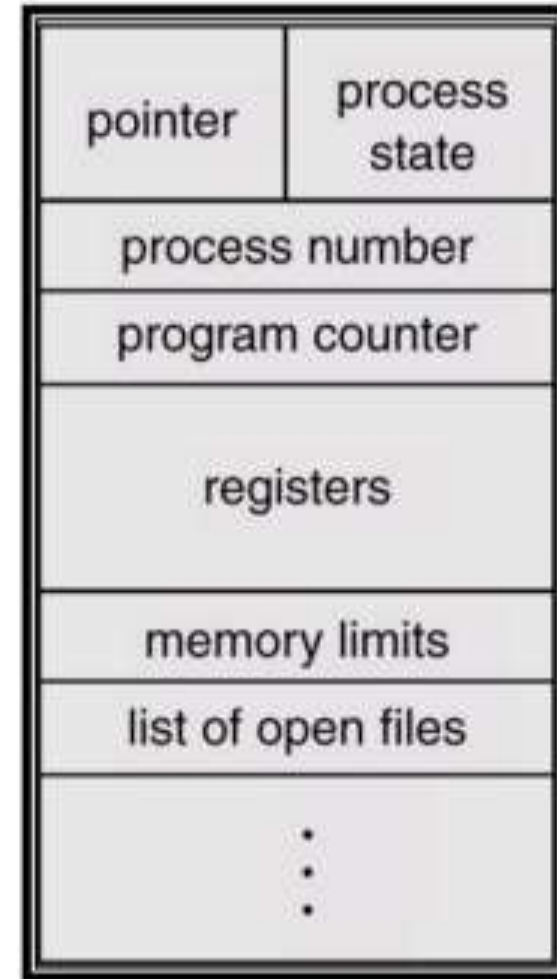
TIẾN TRÌNH

PROCESS CONTROL BLOCK (PCB)

KHOẢNG ĐIỀU KHIỂN TIẾN TRÌNH

PCB gồm:

- Trạng thái tiến trình: new, ready...
- Bộ đếm chương trình
- Các thanh ghi
- Thông tin lập thời biểu CPU: bộ ưu tiên,...
- Thông tin quản lý bộ nhớ
- Lượng CPU, thời gian sử dụng
- Thông tin trạng thái I/O



Sharing is learning

TIẾN TRÌNH

ĐỊNH THỜI TIẾN TRÌNH

Định thời

- Đa chương
 - Nhiều tiến trình chạy tại một thời điểm
 - **Mục tiêu: tận dụng tối đa CPU**
- Chia thời
 - User tương tác với mỗi chương trình đang thực thi
 - **Mục tiêu: tối thiểu thời gian đáp ứng**

Scheduling

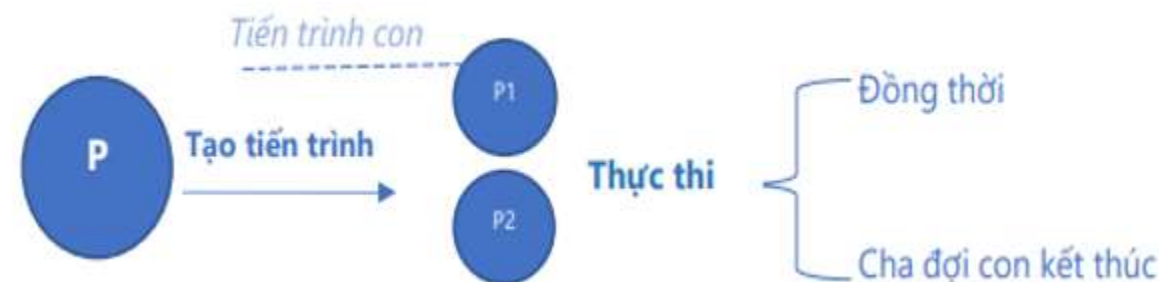
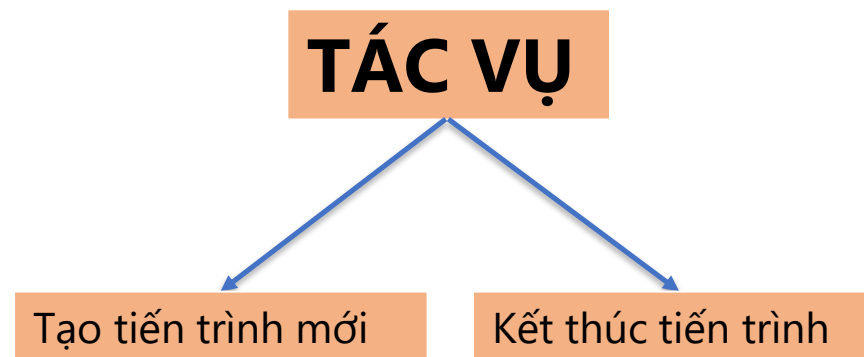
- Long – term
 - Xác định chương trình nào được chấp nhận nạp vào hệ thống để thực thi (new -> ready)
- Medium – term
 - Xác định tiến trình nào được đưa vào và đưa ra khỏi vùng nhớ
- Short – term
 - Xác định tiến trình nào trong ready queue sẽ được chiếm CPU để thực thi kế tiếp

TIỀN TRÌNH

CÁC TÁC VỤ ĐỐI VỚI TIẾN TRÌNH

Tạo tiến trình mới

- Một tiến trình có thể tạo nhiều tiến trình mới thông qua một lời gọi hệ thống create – process (Vd: Hàm fork() trong Unix)
- Tiến trình được tạo là tiến trình con của tiến trình tạo
- Tiến trình con nhận tài nguyên từ hệ điều hành hoặc từ tiến trình cha
- Chia sẻ tài nguyên của tiến trình cha
 - Tiến trình cha và con chia sẻ mọi tài nguyên
 - Tiến trình con chia sẻ một phần tài nguyên của cha
- Trình thực thi
 - Tiến trình cha và con thực thi đồng thời
 - Tiến trình cha đợi đến khi các tiến trình con kết thúc

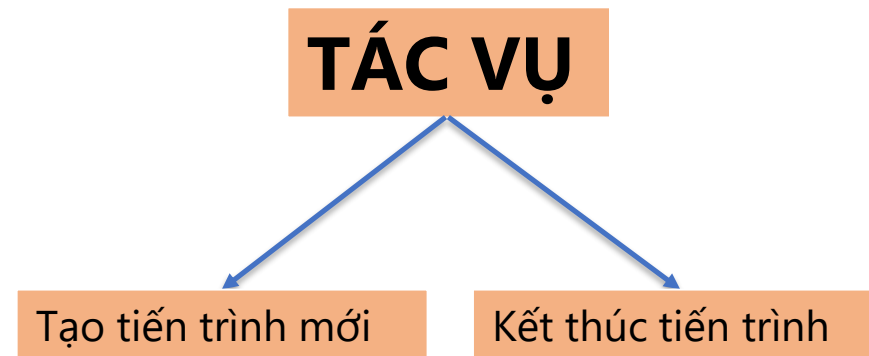


TIẾN TRÌNH

CÁC TÁC VỤ ĐỐI VỚI TIẾN TRÌNH

Kết thúc tiến trình

- Tiến trình tự kết thúc
 - Khi thực thi lệnh cuối cùng và gọi system routine exit
- Tiến trình tự kết thúc do tiến trình khác có quyền kết thúc nó
 - Gọi system routine abort với tham số là pid của tiến trình cần được kết thúc
 - Tiến trình cha kết thúc và kéo theo tất cả tiến trình con của nó đều kết thúc



Sharing is learning

TIẾN TRÌNH

CÁC TÁC VỤ ĐỐI VỚI TIẾN TRÌNH

Hàm fork

- ✓ Fork() tạo ra tiến trình mới bằng cách nhân bản (duplicate) tiến trình gọi hàm này
- ✓ Tiến trình ban đầu gọi là tiến trình cha (parent process)
- ✓ Tiến trình được nhân bản ra được gọi là tiến trình con (child process), là một bản sao giống với tiến trình cha tạo ra nó (kể cả trạng thái thực thi)

Giá trị trả về

{ PID của tiến trình con nếu tạo được tiến trình con
-1 nếu tạo tiến trình con bị lỗi
0 cho tiến trình con



TIẾN TRÌNH

CÁC TÁC VỤ ĐỐI VỚI TIẾN TRÌNH

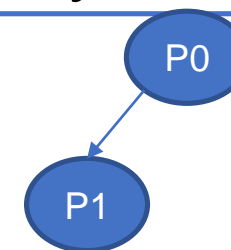
Ví dụ

```
#include <stdio.h>
int main() {
    fork();
    printf("BHT CNPM");
}
```



Cho biết output khi chương trình chạy ?

Hướng dẫn làm
Cây tiến trình



Output: BHT CNPM BHT CNPM



Sharing is learning

TIẾN TRÌNH

CÁC TÁC VỤ ĐỐI VỚI TIẾN TRÌNH

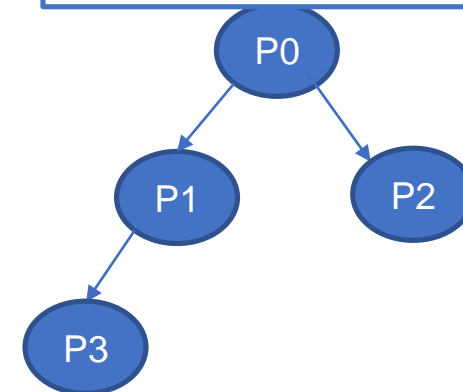
Ví dụ

```
#include <stdio.h>
int main() {
    fork();
    fork();
    printf("BHT CNPM");
}
```



Cho biết output khi chương trình chạy ?

Hướng dẫn làm
Cây tiến trình



Output: BHT CNPM BHT CNPM
BHT CNPM BHT CNPM



Sharing is learning

TIẾN TRÌNH

CÁC TÁC VỤ ĐỐI VỚI TIẾN TRÌNH

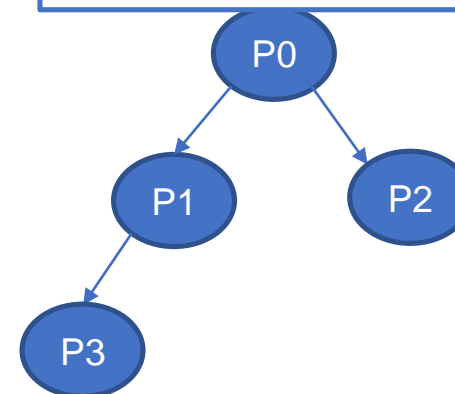
Ví dụ

```
#include <stdio.h>
int main() {
    int a = fork();
    if(a > 0){
        fork();
        printf("Hi");
    }
    else{
        fork();
        printf("Hello");
    }
}
```



Cho biết output khi chương trình chạy ?

Hướng dẫn làm
Cây tiến trình



Output: Hello Hello Hi Hi



Sharing is learning

TIẾN TRÌNH

CÁC TÁC VỤ ĐỐI VỚI TIẾN TRÌNH

Ví dụ

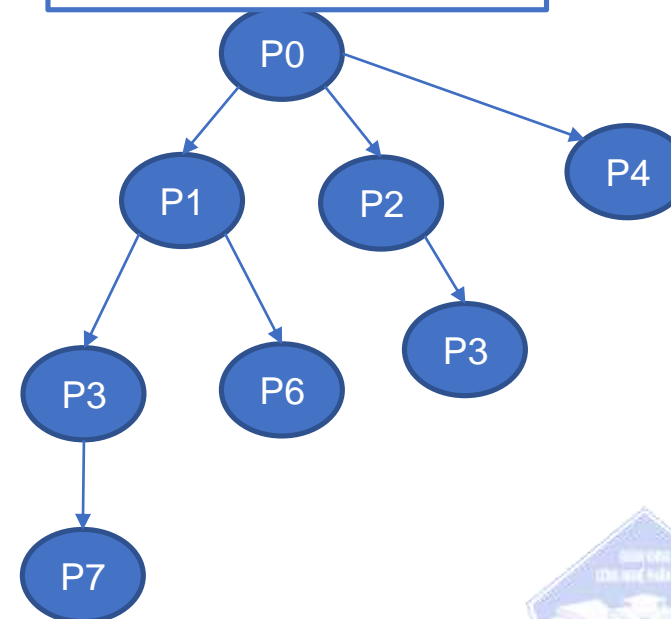
```
#include <stdio.h>
int main() {
    printf("Hello, Alo");
    fork();
    fork();
    printf("Hi");
    fork();
    printf("Alo");
}
```



Có bao nhiêu chữ Hello, Hi, Alo ?

A.1,4,8 B.1,4,6 C. 1,2,9 **D.1,4,9**

Hướng dẫn làm
Cây tiến trình



Sharing is learning

TIẾN TRÌNH

CÁC CÂU HỎI TRẮC NGHIỆM

Câu 1(GK2020-2021): Khi tiến trình được nạp bộ nhớ, stack section của nó **KHÔNG** chứa thành phần nào dưới đây?

- A. Biến cục bộ
- B Địa chỉ trả về
- C. Tham số truyền cho hàm
- ☒ D. Biến toàn cục

Câu 2: Tiến trình ở trạng thái running không thể chuyển sang trạng thái nào dưới đây?

- ☒ A. New
- B ready
- C. waiting
- D. terminated



TIẾN TRÌNH

CÁC CÂU HỎI TRẮC NGHIỆM

Câu 3: Chọn phát biểu sai ?

- A. Một tiến trình có thể tạo nhiều tiến trình mới thông qua một lời gọi hệ thống create – process
- B. Một tiến trình có thể tự kết thúc hoặc bị tiến trình khác kết thúc
- C. Tiến trình con có thể nhận tài nguyên từ hệ điều hành hoặc từ tiến trình cha
- ☒ D. Tiến trình cha và tiến trình con luôn được thực thi đồng thời.



TIẾN TRÌNH

CÁC CÂU HỎI TRẮC NGHIỆM

Câu 4: Cho đoạn mã nguồn sau:

```
#include <stdio.h>
#include <unistd.h>
int main() {
    int i;
    fork();
    for(int i = 0 ;i< =2; i++){
        k = fork();
        if( k > 0 )
            printf("BHTCNPM");
    }
    return 0;
}
```

Khi chạy chương trình thì BHTCNPM in ra bao nhiêu lần ?

A.10

B.12

C.24

D.14

Giải

Trước hết:

$k > 0$ khi là tiến trình cha

$i=0$

$i=1$

$i=2$

Cho rằng các nhánh bên phải là tiến trình cha và tính từ $i = 0$ thì có tổng cộng 14 nhánh tức là có 14 lần xuất hiện BHTCNPM



Sharing is learning

TIẾN TRÌNH

CÁC CÂU HỎI TRẮC NGHIỆM

Câu 5: Cho đoạn mã nguồn sau:

```
#include <stdio.h>
#include <unistd.h>
int main() {
    int x = 1;
    if(fork() == 0) printf("x1= %d", ++x);
    else printf("\n x2= %d", --x);
}
```

Cho biết x1,x2 lần lượt là bao nhiêu ?

- A. 3,1
- B. 3,2
- C. 2,1
- ☒ D. 2,0

Giải

fork() = 0 khi là tiến trình con
Khi fork là con thì if trả về true thì if thực hiện => x1 = 2
Khi fork là cha thì sẽ nhảy vào else => x2 = 0



TRAINING

Hệ Điều Hành

Chương IV: Định thời CPU



Sharing is learning



Sharing is learning

ĐỊNH THỜI CPU

1. Các khái niệm và các loại bộ định thời
2. Các tiêu chuẩn định thời CPU
3. Các yếu tố của giải thuật định thời
4. Các giải thuật định thời

ĐỊNH THỜI CPU

CÁC KHÁI NIỆM VÀ CÁC LOẠI BỘ ĐỊNH THỜI

- **Trong các hệ thống multitasking**

- ✓ Thực thi nhiều chương trình đồng thời làm tăng hiệu suất hệ thống
- ✓ Tại mỗi thời điểm, chỉ có một process được thực thi
- Cần phải giải quyết vấn đề phân chia, lựa chọn process thực thi sao cho được hiệu quả nhất
- Chiến lược định thời CPU



ĐỊNH THỜI CPU

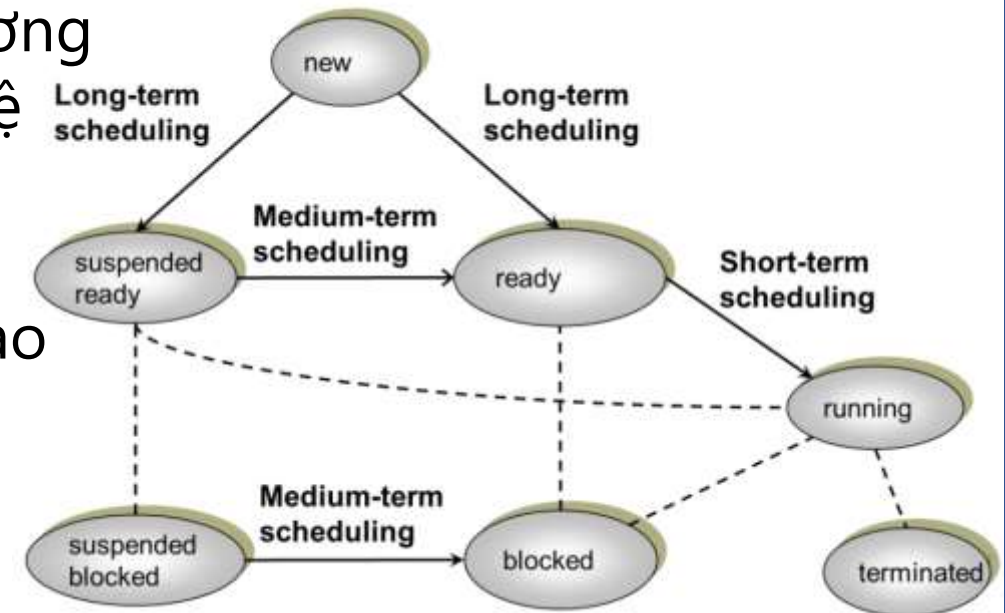
CÁC KHÁI NIỆM VÀ CÁC LOẠI BỘ ĐỊNH THỜI

Các loại bộ định thời

Long-term scheduling: Xác định chương trình nào được chấp nhận nạp vào hệ thống để thực thi

Medium-term scheduling: Process nào được đưa vào (swap in) và đưa ra (swap out) khỏi vùng nhớ chính

Short-term scheduling: Xác định process nào trong ready queue sẽ được chiếm CPU để thực thi kế tiếp



ĐỊNH THỜI CPU

CÁC TIÊU CHUẨN ĐỊNH THỜI CPU

Hướng người dùng

Thời gian đáp ứng: Khoảng thời gian process nhận yêu cầu đến khi yêu cầu đầu tiên được đáp ứng

Thời gian hoàn thành: Khoảng thời gian từ lúc một process được nạp vào hệ thống đến khi process đó kết thúc

Thời gian chờ: Tổng thời gian một process đợi trong ready queue



Cực tiểu



Sharing is learning

ĐỊNH THỜI CPU

CÁC TIÊU CHUẨN ĐỊNH THỜI CPU

Hướng hệ thống

Sử dụng CPU: định thời sao cho CPU càng bận càng tốt

Công bằng: Tất cả process phải được đối xử như nhau

Thông lượng: Số process hoàn tất công việc trong một đơn vị thời gian



Cực đại



Sharing is learning

ĐỊNH THỜI CPU

CÁC YẾU TỐ CỦA GIẢI THUẬT ĐỊNH THỜI

- **Hàm lựa chọn (selection function)**

- ✓ Chọn process nào trong ready queue được thực thi

- **Chế độ quyết định (decision mode)**

- ✓ Chọn thời điểm thực hiện hàm chọn lựa

- ✓ Gồm hai chế độ

- Chế độ không trưng dụng (non-preemptive): Khi running, process sẽ thực thi cho đến khi kết thúc hoặc bị blocked do yêu cầu I/O

- Chế độ trưng dụng (preemptive): Khi running, process có thể bị ngắt nửa chừng chuyển về trạng thái ready



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

- 1. First-Come, First-Served (FCFS)**
- 2. Shortest Job First (SJF)**
- 3. Shortest Remaining Time First (SRTF)**
- 4. Priority Scheduling**
- 5. Round-Robin (RR)**
- 6. Highest Response Ratio Next (HRRN)**
- 7. Multilevel Queue**
- 8. Multilevel Feedback Queue**



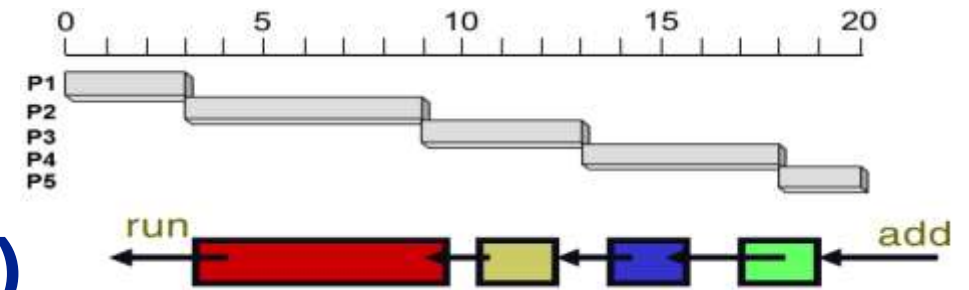
ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

1. First-Come, First-Served (FCFS)

- **Hàm lựa chọn:** Tiến trình nào yêu cầu CPU trước sẽ được cấp phát CPU trước Process sẽ thực thi đến khi kết thúc hoặc bị blocked do I/O

- **Chế độ quyết định:** non-preemptive algorithm
- **Hiện thực:** sử dụng hàng đợi FIFO (FIFO queues)
 - Tiến trình đi vào được thêm vào cuối hàng đợi
 - Tiến trình được lựa chọn để xử lý được lấy từ đầu queues



ĐỊNH THỜI CPU

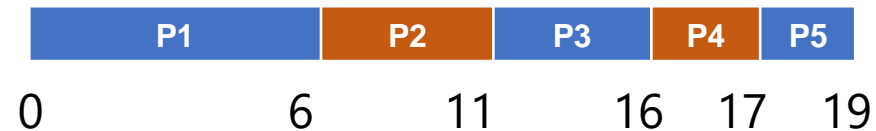
CÁC GIẢI THUẬT ĐỊNH THỜI

1. First-Come, First-Served (FCFS)

Ví dụ: Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào Ready Queues và thời gian cần CPU tương ứng như bảng sau.

Process	Arrival Time	Burst Time
P1	0	6
P2	2	5
P3	5	5
P4	10	1
P5	12	2

Giản đồ Gantt



Thời gian đáp ứng: P1 = 0, P2 = 4, P3 = 6, P4 = 6, P5 = 5

=> Thời gian đáp ứng trung bình: 4.2

Thời gian chờ: P1 = 0, P2 = 4, P3 = 6, P4 = 6, P5 = 5

=> Thời gian chờ trung bình: 4.2

Thời gian hoàn thành: P1 = 6, P2 = 9, P3 = 11, P4 = 7, P5 = 7

=> Thời gian hoàn thành trung bình: 8



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

2. Shortest Job First (SJF)

- **Hàm lựa chọn:**

Lựa chọn Process có thời gian thực thi ngắn nhất trước.

- **Chế độ quyết định:**

- ❖ Non-preemptive

Khi CPU được trao cho quá trình nó không nhường cho đến khi kết thúc chu kỳ xử lý của nó



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

2. Shortest Job First (SJF)

- **Chế độ quyết định:**

- ❖ Preemptive (Shortest-Remaining-Time-First)

Nếu một tiến trình mới được đưa vào có chiều dài sử dụng CPU cho lần tiếp theo nhỏ hơn thời gian còn lại của tiến trình đang xử lý
→ Dừng hoạt động tiến trình hiện hành

- **Ưu điểm:** Thời gian chờ đợi trung bình giảm

- **Nhược điểm:** Process lớn sẽ đói (starvation) khi có nhiều process nhỏ.



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

2. Non – preemptive SJF

Ví dụ: Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào Ready Queues và thời gian cần CPU tương ứng như bảng sau

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	9
P4	10	2
P5	12	5

Giản đồ Gantt:



0 12 14 19 26 35

Thời gian đáp ứng: P1 = 0, P2 = 17, P3 = 21, P4 = 2, P5 = 2
=> Thời gian đáp ứng trung bình: 8.4

Thời gian chờ: P1 = 0, P2 = 17, P3 = 21, P4 = 2, P5 = 2
=> Thời gian chờ trung bình: 8.4

Thời gian hoàn thành: P1 = 12, P2 = 24, P3 = 30, P4 = 4, P5 = 7
=> Thời gian hoàn thành trung bình: 15.4



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

3. Preemptive SJF (SRTF)

Ví dụ: Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào Ready Queues và thời gian cần CPU tương ứng như bảng sau

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	9
P4	10	2
P5	12	5

Giản đồ Gantt:



Thời gian đáp ứng: P1 = 0, P2 = 0, P3 = 4, P4 = 0, P5 = 0

=> Thời gian đáp ứng trung bình: 0.8

Thời gian chờ: P1 = 23, P2 = 0, P3 = 11, P4 = 0, P5 = 0

=> Thời gian chờ trung bình: 6.8

Thời gian hoàn thành: P1 = 35, P2 = 7, P3 = 20, P4 = 2, P5 = 5

=> Thời gian hoàn thành trung bình: 13.8



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

4. Priority Scheduling

- **Hàm lựa chọn:** Mỗi process sẽ có một độ ưu tiên, CPU sẽ được cấp cho process có độ ưu tiên cao nhất
- **Chế độ quyết định**
 - ❖ Non-preemptive và Preemptive
- **Ưu điểm:** Các process quan trọng được thực thi trước
- **Nhược điểm:** Trì hoãn vô hạn định cho các process có độ ưu tiên thấp. Giải pháp là tăng độ ưu tiên theo thời gian



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

4. Non – Preemptive Priority Scheduling

Ví dụ: Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào Ready Queues và thời gian cần CPU tương ứng như bảng sau

Process	Arrival Time	Burst Time	Priority
P1	0	12	2
P2	2	7	1
P3	5	9	4
P4	10	2	5
P5	12	5	3

Giản đồ Gantt:



Thời gian đáp ứng: P1 = 0, P2 = 10, P3 = 19, P4 = 23, P5 = 7

=> Thời gian đáp ứng trung bình: 11.8

Thời gian chờ: P1 = 0, P2 = 10, P3 = 19, P4 = 23, P5 = 7

=> Thời gian đáp ứng trung bình: 11.8

Thời gian hoàn thành: P1 = 12, P2 = 17, P3 = 28, P4 = 25, P5 = 12

=> Thời gian hoàn thành trung bình: 18.8



Sharing is learning

ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

5.Round Robin (RR)

- **Hàm lựa chọn**

- ✓ Mỗi process được cấp cho một định mức thời gian (quantum time – q)
- ✓ Sau khoảng thời gian đó thì process bị đoạt quyền và trở về cuối hàng đợi ready
- ✓ Khi q lớn thì RR sẽ thành FCFS, q quá nhỏ thì tốn chi phí chuyển ngữ cảnh



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

- **Ưu điểm:** Thời gian đáp ứng nhanh
- **Nhược điểm:** Các process dạng CPU-bound (hướng xử lý) vẫn được “ưu tiên” và thời gian chờ đợi thường quá dài
 - Nếu có n process trong hàng đợi ready, quantum time là $q \rightarrow$ mỗi process lấy $1/n$ thời gian CPU theo từng khối có kích thước lớn nhất là q
 - Không có process nào chờ lâu hơn $(n-1)*q$ đơn vị thời gian
 - RR sử dụng giả thiết ngầm là tất cả các process đều có tầm quan trọng ngang nhau
 - Không thể sử dụng RR nếu muốn các process có độ ưu tiên khác nhau



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

5.Round Robin (RR)

Ví dụ: Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào Ready Queues và thời gian cần CPU tương ứng như bảng sau với quantum time = 5

Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	9
P4	10	2
P5	12	5

Giản đồ Gantt:



Thời gian đáp ứng: P1 = 0, P2 = 3, P3 = 5, P4 = 10, P5 = 12

=> Thời gian đáp ứng trung bình: 6

Thời gian chờ: P1 = 23, P2 = 15, P3 = 19, P4 = 10, P5 = 12

=> Thời gian chờ trung bình: 15.8

Thời gian hoàn thành: P1 = 35, P2 = 22, P3 = 28, P4 = 12, P5 = 17

=> Thời gian hoàn thành trung bình: 22.8



Sharing is learning

ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

6. Highest Response Ratio Next

- **Hàm lựa chọn**

- ✓ Chọn process có tỉ lệ phản hồi cao nhất
- ✓ Các process ngắn được ưu tiên hơn

- **Công thức**

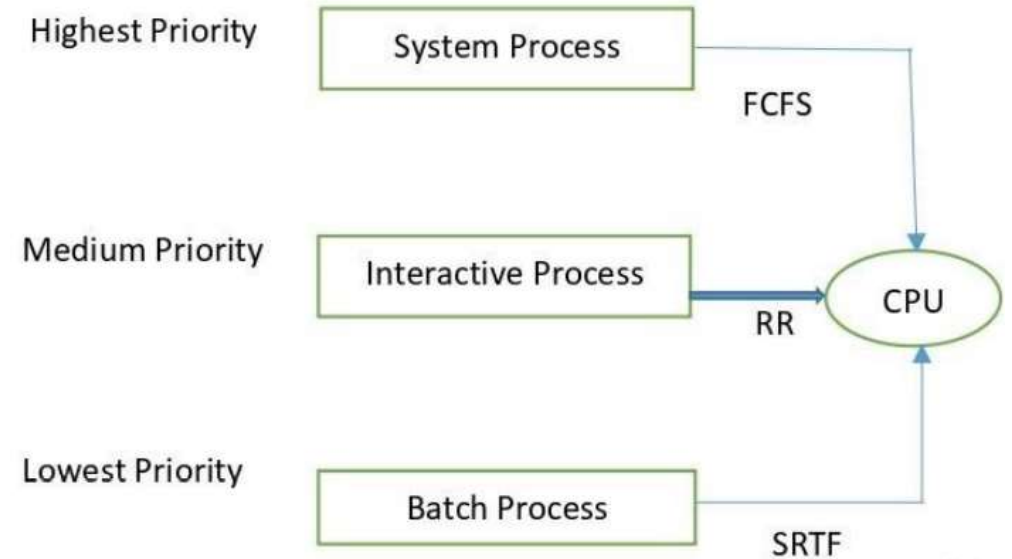
$$RR = \frac{\text{time spent waiting} + \text{expected service time}}{\text{expected service time}}$$



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

7. Multilevel Queue Scheduling



Hàng đợi ready được chia thành các hàng đợi riêng biệt dựa vào các tiêu chuẩn như:

- Đặc điểm và yêu cầu định thời của process
- Foreground và background process,....

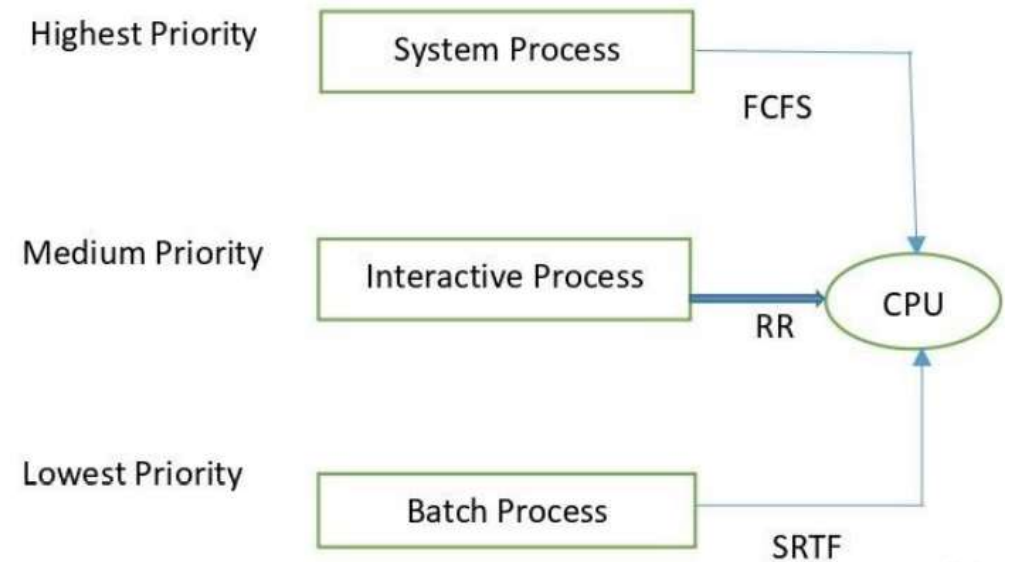


Sharing is learning

ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI

7. Multilevel Queue Scheduling



- ✓ Process được gán cố định vào một hàng đợi, mỗi hàng đợi sử dụng giải thuật định thời riêng
- ✓ Quá trình được chạy ở chế độ giao tiếp (foreground hay interactive process) được ưu tiên hơn so với quá trình chạy nền (background process)



ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI



7. Multilevel Queue Scheduling

Hệ điều hành cần phải định thời cho các hàng đợi

- **Fixed priority scheduling:** phục vụ từng hàng đợi có độ ưu tiên cao đến thấp
 - ❑ Vấn đề: Có thể gây ra tình trạng starvation (đói tài nguyên)
- **Time slice:** Mỗi hàng đợi được nhận một khoảng thời gian chiếm CPU và phân phối cho các process trong hàng đợi khoảng thời gian đó
 - ❑ Ví dụ: 80% cho hàng đợi foreground định thời bằng Round Robin và 20% cho hàng đợi background định thời bằng giải thuật FCFS

ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI



8. Multilevel Feedback Queue

Vấn đề của Multilevel queue

Process không thể chuyển từ hàng đợi này sang hàng đợi khác

→ **Giải pháp:** Cơ chế feedback

Cơ chế feedback

- Giải quyết được vấn đề của Multilevel queue
- Nếu một quá trình dùng quá nhiều thời gian của CPU thì nó bị di chuyển đến hàng đợi có độ ưu tiên thấp
- Ngược lại, khi chờ lâu trong hàng đợi có độ ưu tiên thấp quá lâu thì nó có thể được di chuyển sang hàng chờ có độ ưu tiên cao

ĐỊNH THỜI CPU

CÁC GIẢI THUẬT ĐỊNH THỜI



8. Multilevel Feedback Queue

Ưu và nhược điểm

- ✓ Là thuật toán định thời phổ biến và phức tạp nhất.
- ✓ Những quá trình trong thời gian t nào đó được đáp ứng nhanh

Yêu cầu cần giải quyết

- Số lượng hàng đợi bao nhiêu là thích hợp?
- Dùng giải thuật nào ở mỗi hàng đợi?
- Làm sao để xác định thời điểm để chuyển một process đến hàng đợi cao hoặc thấp hơn ?
- Khi process yêu cầu được xử lý thì hàng đợi nào là hợp lý nhất ?

ĐỊNH THỜI CPU

CÂU HỎI TRẮC NGHIỆM & TỰ LUẬN



Câu 1: Chọn phát biểu SAI trong các phát biểu về giải thuật định thời bên dưới?

- A. Trong giải thuật định thời Multilevel Queue, độ ưu tiên của một tiến trình có thể thay đổi
- B. Giải thuật FCFS có thể được xem như giải thuật Round Robin với thời gian Quantum rất lớn
- C. Giải thuật SRTF là giải thuật định thời CPU theo độ ưu tiên với chế độ quyết định trưng dụng
- D. Một trong những kĩ thuật thường dùng để ước lượng thời gian cần CPU tiếp theo của tiến trình là sử dụng bình hàm mũ (exponential averaging) của các thời gian sử dụng CPU trong quá khứ.

ĐỊNH THỜI CPU

CÂU HỎI TRẮC NGHIỆM & TỰ LUẬN



Câu 2: Khi thực hiện giải thuật định thời Round Robin, người ta nhận thấy với time quantum = 10ms thì thời gian lâu nhất mà một tiến trình có thể phải chờ đợi cho đến khi nó được đáp ứng là 120 ms. Hỏi có bao nhiêu tiến trình đang nằm trong hàng đợi ready ?

A .10

B.11

C.12

D.13

ĐỊNH THỜI CPU

CÂU HỎI TRẮC NGHIỆM & TỰ LUẬN

Câu 3: Chọn phát biểu **SAI** trong các phát biểu bên dưới ?

- A. Trong giải thuật SJF có thể xảy ra tình trạng “đói” đối với các tiến trình có CPU – burst nhỏ khi có nhiều tiến trình với CPU burst lớn đến hệ thống.
- B. Trong giải thuật Multilevel Queue, hàng đợi ready được chia thành nhiều hàng đợi, mỗi hàng đợi sử dụng giải thuật định thời riêng.
- C. Giải thuật Multilevel Feedback Queue cho phép các tiến trình có độ ưu tiên khác nhau.
- D. Không thể sử dụng giải thuật Round Robin nếu muốn các tiến trình có độ ưu tiên khác nhau

ĐỊNH THỜI CPU

CÂU HỎI TRẮC NGHIỆM & TỰ LUẬN

Câu 4: Cho 5 tiến trình P1, P2, P3, P4, P5 với thời gian vào cho Ready Queue và thời gian cần CPU tương ứng như bảng sau:

Process	Arival Time	Burst - Time
P1	0	13
P2	6	6
P3	3	4
P4	9	3
P5	10	9

Vẽ giản đồ Gantt. Tính thời gian đáp ứng, chờ đợi, hoàn thành trung bình cho từng giải thuật:

Round Robin, quantum time = 5.

ĐỊNH THỜI CPU

CÂU HỎI TRẮC NGHIỆM & TỰ LUẬN

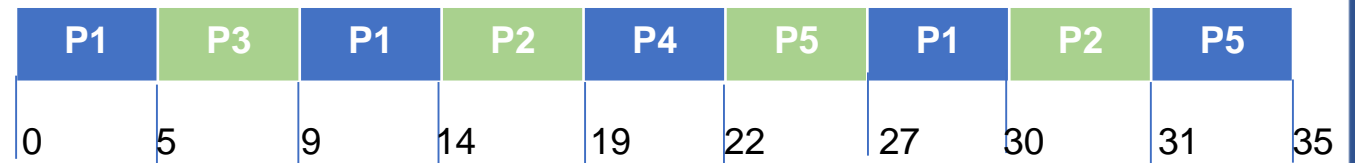
Round Robin, quantum time = 5

Sắp xếp các process tính theo Arival Time theo thứ tự tăng dần

Ta có: P1, P3, P2, P4, P5

=> Giải đồ Gantt là

Process	Arival Time	Burst - Time
P1	0	13
P2	6	6
P3	3	4
P4	9	3
P5	10	9



ĐỊNH THỜI CPU

CÂU HỎI TRẮC NGHIỆM & TỰ LUẬN

Thời gian đáp ứng:

$P1 = 0, P2 = 8, P3 = 2, P4 = 10, P5 = 12$
 \Rightarrow Thời gian đáp ứng trung bình là: 6.4

Thời gian chờ:

$P1 = (9-5) + (27 - 14) = 17,$
 $P2 = (14-6) + (30-19) = 19, P3 = 5-3 = 2,$
 $P4 = 19 - 9 = 10,$
 $P5 = 22 - 10 + 31 - 27 = 16$
 \Rightarrow Thời gian chờ trung bình là: 12.8

Process	Arival Time	Burst - Time
P1	0	13
P2	6	6
P3	3	4
P4	9	3
P5	10	9

P1	P3	P1	P2	P4	P5	P1	P2	P5	
0	5	9	14	19	22	27	30	31	35

ĐỊNH THỜI CPU

CÂU HỎI TRẮC NGHIỆM & TỰ LUẬN

Thời gian hoàn thành: $P1 = 30$,

$P2 = 31 - 6 = 25$,

$P3 = 9 - 3 = 6$,

$P4 = 22 - 9 = 13$,

$P5 = 35 - 10 = 25$

⇒ Thời gian hoàn thành
trung bình là: 19.8

Process	Arival Time	Burst - Time
P1	0	13
P2	6	6
P3	3	4
P4	9	3
P5	10	9

P1	P3	P1	P2	P4	P5	P1	P2	P5	
0	5	9	14	19	22	27	30	31	35



BAN HỌC TẬP CÔNG NGHỆ PHẦN MỀM

TRAINING GIỮA KỲ HỌC KỲ I NĂM HỌC 2022 – 2023



Sharing is learning

HẾT

**CẢM ƠN CÁC BẠN ĐÃ THEO DÕI
CHÚC CÁC BẠN CÓ KẾT QUẢ THI THẬT TỐT!**

 **BAN HỌC TẬP**

Khoa Công nghệ Phần mềm

Trường Đại học Công nghệ Thông tin

Đại học Quốc gia thành phố Hồ Chí Minh

 **CONTACT**

bht.cnpm.uit@gmail.com

fb.com/bhtcnpm

fb.com/groups/bht.cnpm.uit