**LÍ THUYẾT ÔN TẬP HỆ ĐIỀU HÀNH**

[**CHƯƠNG 1** 1](#_Toc155974807)

[**Câu 1:**Các thành phàn của hệ thống máy tính và vai trò của HDH. 1](#_Toc155974808)

[**Câu 2:**  Khái niệm HDH, phân tích rõ 2 chức năng của OS 1](#_Toc155974809)

[**Câu 3:** Dịch vụ OS, những dvụ mà OS cung cấp, làm rõ quá trình tải và chạy khi PC mới khởi động. 1](#_Toc155974810)

[**Câu 4:** Trình bày giao diện lập trình của HDH 2](#_Toc155974811)

[**Câu 5:** Các thành phần của OS 2](#_Toc155974812)

[**Câu 6:**  Trình bày các HDH , ưu và nhược điểm của nó 3](#_Toc155974813)

[**Câu7:**  Hệ thống xử lí theo lô/ mẻ có cần HDH không ? -> Có 4](#_Toc155974814)

[**Câu 8: Đa chương trình là gì? Tại sao phải sử dụng nó\_ slide c1.2** 4](#_Toc155974815)

[**Câu 9:**  Tại sao sử dụng đa chương trình trong PC và lấy ví dụ minh họa chứng minh hiệu suất CPU của đa lớn hơn đơn 4](#_Toc155974816)

[**Câu 10.** Nhân của hdh là gì ? Phân biệt chế độ nhân và chế độ người dùng 4](#_Toc155974817)

[Chương 2: Quản lí file 5](#_Toc155974818)

[**1** 5](#_Toc155974819)

[**2** 6](#_Toc155974820)

[**CHƯƠNG 3: ĐỊA CHỈ LOGIC VÀ ĐỊA CHỈ VẬT LÍ** 6](#_Toc155974821)

[**Câu** 1: Các kĩ thuật phân chương cố định, động và cơ chế ánh xạ địa chỉ khi phân chương. 6](#_Toc155974822)

[**Câu** 2: Kĩ thuật phân chương bằng phương pháp kề cận. Nêu ưu và nhược điểm và sự giống và khác nhau giữa phương pháp kề cận với phân chương động và cố định 8](#_Toc155974823)

[**Câu** 3: Nêu kĩ thuật phân chương động bộ nhớ, ưu – nhược điểm so với phân chương cố định. Lấy VD cho các chiến lược sử dụng First Fit, Best Fit, Worst Fit. Khi di chuyển chương sang vị trí khác có cần thay đổi thông tin gì trong khối ánh xạ địa chỉ không. 9](#_Toc155974824)

[**Câu** 4: Phân tích ưu/nhược của kthuật tải trong lúc chạy và liên kết động, thư viện dùng chung 9](#_Toc155974825)

[**Câu** 5: Cấu trúc và cách ánh xạ địa chỉ của phương pháp phân đoạn bộ nhớ, so sánh với phân trang 10](#_Toc155974826)

[**Câu** 6: Phân trang bộ nhớ và cơ chế ánh xạ địa chỉ 10](#_Toc155974827)

[**Câu** 7:Trình bày cơ chế kết hợp phân trang với phân đoạn , vẽ sơ đồ giải thích ánh xạ địa chỉ đó. 11](#_Toc155974828)

[**Câu** 9: Trình bày các thuật toán đổi trang 12](#_Toc155974829)

[**Câu** 10: Trình bày thuật toán nạp trang theo nhu cầu của bộ nhớ ảo \_ (slide tr 34\_ Đọc thêm: kỹ thuật tăng tốc độ bảng trang và bảng trang nhiều mức ) + **Câu** 11: Trình bày thuật toán nạp trang theo như cầu của bộ nhớ. Phân tích cùng 1 lệnh có thể xảy ra nhiều sự kiện lỗi trang hay không ? 13](#_Toc155974830)

[**Câu** 12: Nêu kĩ thuật để tăng tốc độ truy cập bảng trang và bảng trang nhiều mức. 14](#_Toc155974831)

[**CHƯƠNG 4:** 14](#_Toc155974832)

[**Câu** 1: Nêu khái niệm tiến trình và so sánh sự khác nhau giữa tiến trình và chương trình. 14](#_Toc155974833)

[**Câu** 2: Nêu tên các thuật toán quản lí tiến trình . 14](#_Toc155974834)

[**Câu** 3: Thao tác khi tạo mới tiến trình, và trình bày tiến trình có thể bị kết thúc trong TH nào. 14](#_Toc155974835)

[**Câu** 4: Nêu khái niệm về dòng / luồng, thế nào là dòng ở mức nhân và ở mức người dùng 15](#_Toc155974836)

[**Câu** 5: Các thông tin nào được lưu trữ trong quản lí thông tin PCB. 16](#_Toc155974837)

[**Câu** 6: Nêu 5 trạng thái của tiến trình, vẽ sơ đồ và giải thích chuyển đổi giữa 5 trạng thái này. 16](#_Toc155974838)

[**Câu** 7: \*\* Điều độ dòng, điều độ tiến trình là gì ? 2 thuật toán có khác nhau không và giải thích? 17](#_Toc155974839)

[**Câu**  8: Trình bày thuật toán phân phối lại và không phân phối lại. 17](#_Toc155974840)

[**Câu** 9: Trình bày mô hình đa luồng, vấn đề sở hữu luồng và đa luồng. Nêu ưu/nhược điểm của nó 17](#_Toc155974841)

[**Câu** 10: Nêu các tiêu chí để đánh giá thuật toán điều độ ( 7 tiêu chí) 17](#_Toc155974842)

[**Câu** 11: Trình bày các giải pháp phần cứng cho vấn đề loại trừ tương hỗ và đoạn nguy hiểm. 18](#_Toc155974843)

[**Câu** 12:Sử dụng Test and Set cho bài toán triết gia ăn cơm và giải pháp này có gây ra vấn đề bế tắc hay đói không ? 18](#_Toc155974844)

[**Câu** 13: Phương pháp cờ hiệu Semanphore cho vấn đề loại trừ tương hỗ và đoạn nguy hiểm. 19](#_Toc155974845)

# **CHƯƠNG 1**

## **Câu 1:**Các thành phàn của hệ thống máy tính và vai trò của HDH.

* Một hệ thống máy tính nói chung được phân chia thành phần cứng và phần mềm:
  + Phần cứng: cung cấp các tài nguyên cần thiết cho việc tính toán, xử lý dữ liệu
  + Phần mềm: các chương trình cụ thể. (phần mềm hệ thống và phần mềm ứng dụng)
* Vai trò của HĐH: phần mềm đóng vai trò trung gian giữa phần cứng và người sử dụng chương trình ứng dụng, làm cho việc sử dụng hệ thống máy tính được tiện lợi và hiệu quả và phần cứng của máy tính nhằm thực hiện 2 chức năng cơ bản: quản lý tài nguyên, Quản lý việc thực hiện các chương trình

## **Câu 2:** Khái niệm HDH, phân tích rõ 2 chức năng của OS

* Hệ điều hành: được định nghĩa thông qua mục đích, vai trò, và chức năng trong hệ thống máy tính. Hệ điều hành là hệ thống phần mềm đóng vai trò trung gian giữa người

## **Câu 3:** Dịch vụ OS, những dvụ mà OS cung cấp, làm rõ quá trình tải và chạy khi PC mới khởi động.

* **Dịch vụ HDH :**
* Một trong những nhiệm vụ chủ yếu của HDH là tạo ra môi trường thuận lợi cho các chương trình khác thực hiện và giúp người sử dụng hệ thống dễ dàng.
* Các dịch vụ có thể thay đổi theo từng HDH. Một số HDH có thể cung cấp nhiều dịch vụ khi hệ điều hành khác có thể cung cấp ít dịch vụ hơn. Ví dụ như MS-DOS không cung cấp dịch vụ về bảo mật trong khi Windows NT lại rất chú trọng tới dịch vụ này.
* **Các dịch vụ HDH cung cấp:**
* Tải và chạy chương trình:
* Để thực hiện, chương trình được tải từ đĩa vào bộ nhớ, sau đó được trao quyền thực hiện các lệnh. Khi thực hiện xong, cần giải phóng bộ nhớ và các tài nguyên
* Toàn bộ quá trình này tương đối phức tạp song lại diễn ra thường xuyên.

HDH sẽ thực hiện công việc phức tạp và lặp đi lặp lại này

* Do HDH là chương trình đầu tiên được thực hiện khi khởi động hệ thống nên HDH tự tải mình vào bộ nhớ
* Nhờ có HĐH, lập trình viên, người sử dụng không cần quan tâm chi tiết đến việc tải và chạy chương trình.
* Giao diện với người dùng: cho phép giao tiếp giữa HDH và người dùng:
* Dưới dạng dòng lệnh (command-line): cho phép người dùng chỉ thị cho HĐH bằng cách gõ lệnh dưới dạng văn bản. Ví dụ: chtr cmd.exe của Windows.
* Giao diện đồ họa (Graphic User Interface-GUI): sử dựng hệ thống cửa sổ, thực đơn và thiết bị trỏ chuột, kết hợp với bàn phím để giao tiếp với hệ thống.
* Thực hiện các thao tác vào/ ra dữ liệu: Người dùng và chương trình trong khi thực hiện có thể có nhu cầu I/O dữ liệu với các đĩa và thiết bị ngoại vi. Để tránh cho chương trình không phải làm việc với phần cứng, yêu cầu I/O sẽ được giao cho hệ điều hành thực hiện.
* Làm việc với hệ thống file:

• Nhu cầu đọc, ghi, tạo, xóa, chép file hoặc làm việc với thư mục

• Quản lý quyền truy cập, sao lưu.

* Phát hiện và xử lý lỗi:

• Phát hiện và xử lý kịp thời các lỗi xuất hiện trong phần cứng cũng như phần mềm => Đảm bảo cho hệ thống hoạt động ổn định, an toàn

• Ví dụ: các lỗi phần cứng như hết bộ nhớ, mất điện, máy in hết mực,hết giấy,..

* Truyền thông: Cung cấp dịch vụ cho phép thiết lập liên lạc và truyền thông tin dưới dạng thông điệp hoặc qua BN dùng chung.
* Cấp phát tài nguyên:

• Trong các hệ thống cho phép nhiều chương trình thực hiện đồng thời cần có cơ chế cấp phát và phân phối tài nguyên hợp lý

• Người dùng và trình ứng dụng không phải tự thực hiện việc cấp phát tài nguyên mà vẫn đảm bảo cấp phát công bằng và hiệu quả

* Dịch vụ an ninh và bảo mật:

• Cấp phát tài nguyên:

• Đối với hệ thống nhiều người dùng thường xuất hiện yêu cầu bảo mật thông tin, tức là người dùng này không tiếp cận được thông tin của người khác nếu không được cho phép.

• Cần đảm bảo để tiến trình không truy cập trái phép tài nguyên (như vùng nhớ, file mở) của tiến trình khác hay chính HDH sẽ thực hiện bằng cách kiểm soát truy cấp tới tài nguyên

## **Câu 4:** Trình bày giao diện lập trình của HDH

* Giao diện này bao gồm các lời gọi hệ thống (system call) mà chương trình sử dụng yêu cầu một dịch vụ nào đó từ phía HDH.
* Lời gọi hệ thống: các lệnh đặc biệt mà CTUD gọi khi cần yêu cầu HDH thực hiện một việc gì đó.
* Lời gọi hệ thống được thực hiện qua những thư viện hàm gọi là thư viện hệ thống. Các hàm này sẽ giúp người lập trình gọi lời gọi hệ thống tương ứng của hệ điều hành.

## **Câu 5:** Các thành phần của OS

Hệ điều hành là một hệ thống phần mềm phức tạp được tạo thành từ nhiều thành phần đảm đương những nhiệm vụ hoặc cung cấp những dịch vụ khác nhau. Các thành phần thực hiện nhiệm vụ sau:

* Quản lý tiến trình
* Quản lý bộ nhớ
* Quản lý vào, ra
* Quản lý file và thư mục
* Hỗ trợ mạng và xử lý phân tán
* Giao diện với người dùng
* Các chương trình tiện ích và ứng dụng

1. Quản lý tiến trình:Một chương trình đang trong quá trình thực hiện được gọi là tiến trình.

|  |  |
| --- | --- |
| Chương trình | Tiến trình |
| - Là một thực thể tĩnh  - Được ghi dưới dạng những bit, những byte trên đĩa | - Là một thực thể động  - Đang tiến hành việc xử lý, tính toán, được cung cấp một số tài nguyên: thời gian CPU, bộ nhớ |

* Tạo và xoá tiến trình (bao gồm cả tiến trình người dùng và tiến trình hệ thống)
* Tạm treo và khôi phục các tiến trình bị treo
* Đồng bộ hoá các tiến trình (lập lịch cho các tiến trình .v.v.)
* Tạo cơ chế liên lạc giữa các tiến trình
* Giải quyết các bế tắc, ví dụ như khi có xung đột về tài nguyên
* Bế tắc: là chương trình đang cần tài nguyên những nó chờ đợi mà không được cung cấp.

1. Quản lí bộ nhớ

* Quản lý, cung cấp và giải phóng
* Cung cấp và giải phóng bộ nhớ theo yêu cầu của các tiến trình
* Quản lý không gian nhớ đã được cấp và không gian còn trống
* Quản lý việc phân phối bộ nhớ giữa các tiến trình => đảm bảo việc chạy song song giữa nhiều chương trình
* Tạo ra bộ nhớ ảo và ánh xạ địa chỉ bộ nhớ ảo vào bộ nhớ thực

1. Quản lý hệ thống vào ra:

* Quản lý thông qua các chương trình điều khiển
* Đơn giản hoá và tăng hiệu quả quá trình trao đổi thông tin giữa các tiến trình với thiết bị vào ra

1. Quản lý file và thư mục:

* Tạo, xóa file và thư mục
* Đọc ghi file
* Ánh xạ file và thư mục sang bộ nhớ ngoài

1. Hỗ trợ mạng và xử lý phân tán:

* Quản lý thiết bị mạng
* Hỗ trợ các giao thức truyền thông
* Quản lý truyền thông, cân bằng tải Thông qua các thành phần điều khiển, giao tiếp mạng.

1. Giao diện với người dùng:

* Đó là hệ thống thông dịch lệnh
* Giúp cho máy tính hiểu, xử lý được các chỉ thị, các lệnh của người dùng.

Ví dụ: bash của Linux, command của window

1. Các chương trình tiện ích và ứng dụng

## **Câu 6:** Trình bày các HDH , ưu và nhược điểm của nó

1. UNIX

* Ưu điểm

• Là một hệ điều hành đa nhiệm, được phát triển đầu tiên bởi Ken Thompson, Dennis Ritchie và Douglas Mcllroy tại AT & T Bell.

• UNIX được nghiên cứu tại các phòng thí nghiệm năm 1969 và dần cải tiến, phát triển và trở nên phổ biến. Unix lần đầu tiên được lập trình lại bởi Ken Thompson bằng ngôn ngữ C vào năm 1973.

• Tạo ra ngôn ngữ cấp cao trong các hệ điều hành

• Tạo ra hệ thống tập tin phân cấp

• Unix shell đã truyền cảm hứng cho nhiều trình thông dịch dòng lệnh phát triển sau đó.

• Giúp ngôn ngữ lập trình C trở nên phổ biến hơn

• Góp phần vào sự ra mắt của phong trào phần mềm miễn phí

* Nhược điểm

•Không có giao diện đồ họa mặc định (phải sử dụng các giao diện đồ họa của bên thứ ba).

1. MINIX (Từ mini-Unix)

* Ưu điểm

• Là một hệ điều hành máy tính tựa Unix dựa trên kiến trúc micro- kernel.

• Phiên bản đầu của MINIX được tạo ra bởi Andrew S. Tanenbaum cho mục đích giáo dục như minh họa, phục vụ đào tạo, có thể sử dụng miễn phí.

• MINIX bây giờ phát triển như là phần mềm nguồn mở.

* Nhược điểm:

• Có giới hạn trong số lượng ứng dụng và hỗ trợ so với các hệ điều hành lớn khác và chỉ cho phép việc sử dụng Minix trong giáo dục

1. LINUX

* Ưu điểm:

• Hệ điều hành nguồn mở với cộng đồng lớn hỗ trợ và phát triển.

• Linus Torvalds tạo ra Linux để vượt qua hạn chế của Minix.

• Hỗ trợ nhiều kiến trúc và thiết bị.

* Nhược điểm: Một số phiên bản Linux có thể khó sử dụng cho người mới học.

1. MS-DOS

• Là sản phẩm của hãng Microsoft và được trang bị cho các máy PC đầu tiên của IBM

* Ưu điểm

• Để có thể chạy trên PC với tài nguyên hạn chế, MS-DOS được xây dựng đơn giản và ít chức năng hơn

• Nhiều giải pháp kỹ thuật trong MS-DOS có nguồn gốc từ UNIS như giao diện lập trình (lời gọi hệ thống), cấu trúc phân cấp của thư mục, bộ dịch lệnh

* Nhược điểm: Không có các chức năng như bảo mật, hỗ trợ mạng, hỗ trợ nhiều tiến Trình

1. Windows NT (NT-new technology)

• Là một thành viên của họ điều hành thế hệ mới như Windows 2000, XP, Vista,7.

• Phiên bản đầu tiên được phát hành vào năm 1993

* Ưu điểm

• Đây là hđh sử dụng nhiều kỹ thuật tiên tiến trong lĩnh vực hđh đã được phát triển cho đến thời điểm này gồm các giải pháp lấy từ UNIX

• Là một hệ điều hành đa nhiệm, hỗ trợ mạng, có các chức năng bảo mật, có giao diện đồ họa dưới dạng cửa sổ và được dùng cho cả máy PC yêu cầu độ ổn định cao.

* Nhược điểm :

• Yêu cầu tài nguyên hệ thống cao hơn so với một số hệ điều hành khác.

• Có thể gặp vấn đề với các ứng dụng không tương thích trong quá trình nâng cấp.

## **Câu7:** Hệ thống xử lí theo lô/ mẻ có cần HDH không ? -> Có

Kĩ thuật xử lý theo mẻ:

• Chương trình được phân thành các mẻ: gồm những chương trình có yêu cầu giống nhau

• Toàn bộ mẻ được nạp vào băng từ và được tải vào máy để thực hiện lần lượt

## **Câu 8: Đa chương trình là gì? Tại sao phải sử dụng nó\_ slide c1.2**

* Đa chương trình:

• Hệ thống chứa đồng thời nhiều chương trình trong bộ nhớ

• Khi một chương trình phải dừng lại để thực hiện vào ra, HDH sẽ chuyển CPU sang thực hiện một chương trình khác. Giảm thời gian chạy không tải của CPU Sử dụng đa chương trình

• Thời gian chờ đợi của CPU trong chế độ đa chương trình giảm đáng kể so với trong trường hợp đơn chương trình

• HDH phức tạp hơn rất nhiều so với HDH đơn chương trình

• Đòi hỏi hỗ trợ từ phần cứng, đặc biệt khả năng vào/ra bằng ngắt và cơ chế DMA

* Giúp tăng hiệu suất sử dụng phần cứng máy tính, bởi vì khi một chương trình đang chờ một số sự kiện bên ngoài như đầu vào của người dùng hoặc chuyển giao đầu vào/đầu ra với một thiết bị ngoại vi để hoàn thành, bộ xử lý trung tâm vẫn có thể được sử dụng với một chương trình khác. Đa chương trình cũng cho phép nhiều người dùng sử dụng máy tính tại cùng một thời điểm bằng một CPU

## **Câu 9:** Tại sao sử dụng đa chương trình trong PC và lấy ví dụ minh họa chứng minh hiệu suất CPU của đa lớn hơn đơn

* - giúp tăng hiệu suất sử dụng phần cứng máy tính, bởi vì khi một chương trình đang chờ một số sự kiện bên ngoài như đầu vào của người dùng hoặc chuyển giao đầu vào/đầu ra với một thiết bị ngoại vi để hoàn thành, bộ xử lý trung tâm vẫn có thể được sử dụng với một chương trình khác. Đa chương trình cũng cho phép nhiều người dùng sử dụng máy tính tại cùng một thời điểm bằng một CPU.
* Một ví dụ minh họa về hiệu suất của đa chương trình so với đơn chương trình là khi bạn đang chơi một trò chơi trên máy tính của mình và đồng thời đang tải xuống một tệp lớn từ Internet. Nếu bạn chỉ chạy một chương trình, bạn sẽ phải chờ đợi cho đến khi tệp được tải xuống trước khi bắt đầu chơi trò chơi. Tuy nhiên, nếu bạn sử dụng đa chương trình, bạn có thể tải xuống tệp trong khi vẫn chơi trò chơi mà không bị gián đoạn.

## **Câu 10.** Nhân của hdh là gì ? Phân biệt chế độ nhân và chế độ người dùng

* Nhân (kernel) là phần cốt lõi, thực hiện các chức năng cơ bản nhất, quan trọng nhất của HDH và thường xuyên được giữ trong bộ nhớ. Kernel có nhiệm vụ quản lý tài nguyên hệ thống (liên lạc giữa các thành phần phần cứng và phần mềm)
* Phân biệt chế độ nhân và chế độ người dùng:

•Nhân chạy trong chế độ đặc quyền – chế độ nhân: là chế độ mà chương trình thực hiện trong đó có đầy đủ quyền truy cập và điều khiển phần cứng máy tính.

• Chế độ người dùng: chương trình thực hiện trong chế độ người dùng bị hạn chế rất nhiều quyền truy cập và sử dụng phần cứng.

Việc phân biệt chế độ nhân và chế độ người dùng nhằm mục đích ngăn không cho CTUD vô tình hoặc cố ý thực hiện những thao tác làm ảnh hưởng tới hệ thống.

# Chương 2: Quản lí file

## **1**

**a.Trình bày phương pháp cấp phát không gian cho file sử dung các khối liên kết. Khi nào nên sử dụng phương pháp này cho hệ thống file.**

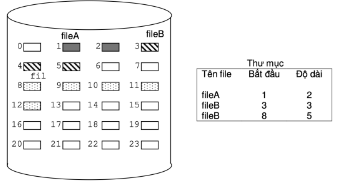
- Được cấp phát một khoảng không gian bao gồm các khối liên tiếp trên đĩa.

- Vị trí file trên đĩa được xác định bởi vị trí khối đầu tiên và độ dài (số khối) mà file đó chiếm.

- Khi có yêu cầu cấp phát, hệ điều hành có số lượng khối đủ cấp phát cho file đó

- Bảng cấp phát chỉ cần 1 khoản mục cho 1 file, chỉ ra khối bắt đầu, và độ dài của file tính = khối.

- Là cấp phát trước sử dụng khích thước phần thay đổi.

\* ví dụ:

**Ưu điểm**:

+ Cho phép truy cập trực tiếp và tuần tự

+ Đơn giản, tốc độ cao do tiết kiệm thời

gian di chuyển đầu từ khi đọc các khối

**Nhược điểm**:

- Phải biết trước kích thước file khi tạo

- Khó tìm ra khoảng không gian trông đủ lớn

trên đĩa để cấp phát cho file

- HĐH cần kiểm tra vùng trống khi cấp phát

- Gây phân mảnh ngoài

\* Phân mảnh ngoài: hiện tượng vùng trống còn lại trên đĩa có kích thước

quá nhỏ do vậy không thể cấp phát cho file có kích thước lơn hơn.

**b. Trình bày các bước cần thiết (trình bày bằng lời, không cần viết mã) để đọc bảng FAT từ thẻ nhớ USB vào bộ nhớ**.

- Đọc thông tin về hệ thống file FAT từ thẻ nhớ USB.

-Xác định vị trí của thư mục gốc trong hệ thống file.

-Đọc thông tin về thư mục gốc từ vị trí đã xác định.

-Xử lý thông tin trong thư mục gốc để lấy danh sách các tệp tin và thư mục con.

**c. Giả sử FAT đã được đọc vào tại bộ nhớ địa chỉ << void \*fat >>, viết đoạn chương trình trên C/C++ để liệt kê các cluster trống trong số N cluster đầu tiên. Giả sử 1 file bắt đầu tai cluster n, viết chương trình các cluster thuộc về file đó.**

#include <stdio.h>

void listEmptyClusters(void \*fat, int N) {

int i;

for (i = 2; i < N + 2; i++) {

if (\*((unsigned short \*)fat + i) == 0x0000) {

printf("Empty cluster: %d\n", i);

}

}

}

int main() {

void \*fat; // Địa chỉ của hệ thống file FAT

int N; // Số lượng cluster cần kiểm tra

// Gán giá trị cho fat và N

listEmptyClusters(fat, N);

return 0;

}

Đoạn mã trên sẽ liệt kê tất cả các cluster trống trong số N cluster đầu tiên của hệ thống file FAT.

**2**

**a. Trình bày phương pháp sử dụng danh sách kết nối trên bảng chỉ số khi cấp phát không gian cho file.**

▶ Các khối trống được liên kết với nhau thành danh sách

▶ Mỗi khối trống chứa con trỏ chỉ tới khối trống tiếp theo

▶ Địa chỉ khối trống đầu tiên được lưu ở vị trí đặc biệt trên đĩa và

được HDH giữ trong MEM khi cần làm việc với các file

▶ Đòi hỏi truy cập lần lượt khi cần duyệt danh sách này

▶ HĐH có thể cấp phát ngay các khối ở đầu danh sách

**b. Để đọc thư mục gốc từ thẻ nhớ USB vào bộ nhớ trong trường hợp hệ thống file của thẻ nhớ là FAT 16, bạn cần thực hiện các bước sau:**

- Đọc thông tin về hệ thống file FAT 16 từ thẻ nhớ USB.

-Xác định vị trí của thư mục gốc trong hệ thống file.

-Đọc thông tin về thư mục gốc từ vị trí đã xác định.

-Xử lý thông tin trong thư mục gốc để lấy danh sách các tệp tin và thư mục con.

**c. Để in tên của một file trong thư mục gốc ROOT có cluster đầu tiên là n, bạn có thể sử dụng đoạn mã sau đây:**

#include <stdio.h>

void printFileName(void \*fat, int n) {

void \*root = fat + 512 \* n; // Địa chỉ của thư mục gốc ROOT

char \*filename = (char \*)root + 0x00; // Địa chỉ của tên file trong thư mục gốc printf("File name: %s\n", filename);

}

int main() {

void \*fat; // Địa chỉ của hệ thống file FAT

int n; // Cluster đầu tiên của thư mục gốc ROOT

// Gán giá trị cho fat và n

printFileName(fat, n);

} Đoạn mã trên sẽ in tên của file trong thư mục gốc ROOT có cluster đầu tiên là n.

**CHƯƠNG 3: ĐỊA CHỈ LOGIC VÀ ĐỊA CHỈ VẬT LÍ**

## **Câu** 1: Các kĩ thuật phân chương cố định, động và cơ chế ánh xạ địa chỉ khi phân chương.

1. Phân chương cố định

* Bộ nhớ được chia thành n phần

• Mỗi phần được gọi là một chương (partition)

• Mỗi chương ở một vị trí cố định

• Chương được sử dụng như một vùng nhớ độc lập:

* Mỗi chương chứa được đúng một tiến trình
* Khi được tải vào, tiến trình được cấp phát một chương. Sau khi tiến trình kết thúc, HĐH giải phóng chương và chương có thể được cấp phát cho tiến trình mới.

• Chương có thể có kích thước bằng nhau hoặc khác nhau

* Kích thước các chương bằng nhau:

• Ưu điểm: Đơn giản

• Nhược điểm: Kích thước chương trình > kích thước chương dẫn đến không thể cấp phát

• Gây phân mảnh trong

* Kích thước các chương khác nhau:

• Có hai cách lựa chọn chương nhớ để cấp cho tiến trình đang chờ đợi bằng cách chọn chương có kích thước nhỏ nhất

* Mỗi chương có một hàng đợi riêng
* Một hàng đợi chung cho tất cả các chương
* Mỗi chương có một hàng đợi riêng: tiến trình có kích thước phù hợp với chương nào sẽ nằm trong hàng đợi của chương đó.
* Một hàng đợi chung cho tất cả các chương: Mỗi khi có một chương trống tiến trình nằm gần đầu hàng đợi nhất và có kích thước phù hợp với chương nhất sẽ được tải và thực hiện
* Ưu điểm

• Đơn giản và ít xử lý

• Giảm thời gian tìm kiếm

* Nhược điểm

• Hệ số song song không thể vượt quá số lượng chương của bộ nhớ

• Bị phân đoạn bộ nhớ

• Kích thước chương trình lớn hơn kích thước chương lớn nhất

• Tổng bộ nhớ tự do còn lớn, nhưng không dùng để nạp các chương trình khác

1. Phân chương động

* Kích thước, số lượng và vị trí chương đều có thể thay đổi
* Các vùng trống bộ nhớ cũng có thể được liên kết thành một danh sách kết nối
* Khi một tiến trình yêu cầu bộ nhớ

• HĐH cấp cho tiến trình 1 chương có kích thước đúng bằng tiến trình đó đang cần

• Nếu tìm thấy vùng nhớ tương ứng

* Vùng trống được chia thành 2 phần
* Một phần cung cấp theo yêu cầu
* Một phần trả lại danh sách vùng trống tự do

• Nếu không tìm thấy

* Phải chờ tới khi có được một vùng trống thỏa mãn
* Cho phép tiến trình khác trong hàng đợi thực hiện (nếu độ ưu tiên đảm bảo)
* Khi tiến trình kết thúc sẽ tạo vùng trống trong Bộ nhớ và các vùng trống nằm cạnh nhau được nhập lại thành vùng lớn hơn
* Các chiến lược cấp chương chọn vùng chống:

• First Fit: Chọn vùng thích hợp đầu tiên

• Best Fit: Vùng thích hợp nhất

• Worst Fit: Vùng trống thích hợp nhất - vùng kích thước lớn nhất

* Ưu điểm : Phân chương động tránh việc phân mảnh trong
* Nhược điểm: Gây phân mảnh ngoài: dồn những vùng trống nhỏ thành lớn (nén)

1. Ánh xạ địa chỉ khi phân chương

* Cho phép các tiến trình truy cập vào bộ nhớ thông qua địa chỉ logic. Khi một tiến trình yêu cầu bộ nhớ, hệ điều hành sẽ ánh xạ địa chỉ logic thành địa chỉ vật lý tương ứng trên bộ nhớ. Kỹ thuật này giúp tăng tốc độ truy cập bộ nhớ và giảm thời gian truy cập, nhưng đòi hỏi phải có phần cứng hỗ trợ
* Khi tiến trình được tải vào bộ nhớ, CPU dành 2 thanh ghi:

• Thanh ghi cơ sở: chứa địa chỉ bắt đầu của tiến trình

• Thanh ghi giới hạn: chứa độ dài chương

* Địa chỉ logic được so sánh với nội dung của thanh ghi giới hạn

• Nếu lớn hơn: lỗi truy cập

• Nhỏ hơn: đưa tới bộ cộng với thanh ghi cơ sở thành địa chỉ vật lý

* Nếu chương bị di chuyển thì nội dung của thanh ghi cơ sở bị thay đổi chứa địa chỉ mới

## **Câu** 2: Kĩ thuật phân chương bằng phương pháp kề cận. Nêu ưu và nhược điểm và sự giống và khác nhau giữa phương pháp kề cận với phân chương động và cố định

Phương pháp kề cận - Buddy system

* Nguyên tắc: Chia đôi liên tiếp vùng trống tự do cho tới khi thu được vùng trống nhỏ nhất thỏa mãn
* Các chương và khối trống có kích thước là 2k (L ≤ k ≤ H)

▶ 2L: kích thước nhỏ nhất của chương

▶ 2H : kích thước của toàn bộ không gian nhớ

▶ Yêu cầu cấp vùng nhớ S

• 2H−1 < S ≤ 2H: cấp cả 2H

• S ≤ 2H−1 Chia vùng trống tìm được thành 2 khối bằng nhau (gọi là buddies) 2H−1

* Nếu 2H−2 < S ≤ 2H−1: cấp cả 2H−1
* Tiếp tục chia đôi tới khi tìm được vùng thỏa mãn 2k−1 < S ≤ 2k
* Phương pháp kề cận cung cấp bộ nhớ nhanh: với bộ nhớ kích thước n, cần duyệt log2n danh sách
* Thu hồi vùng nhớ: Có thể kết hợp 2 vùng kề nhau có cùng kích thước, tiếp tục kết hợp liên tiếp cho tới khi tạo ra vùng trống lớn nhất có thể.

So sánh

|  |  |  |  |
| --- | --- | --- | --- |
| PChg | cố định | động | kề cận |
| Ưu | * đơn giản * dễ hiểu | * linh hoạt * tận dụng tối đa bộ nhớ | giảm lãng phí bộ nhớ |
| Nhược | * không linh hoạt * không thể tận dụng tối đa bộ nhớ | phức tạp hơn phương pháp phân chương cố định | làm tăng thời gian cấp phát bộ nhớ |

## **Câu** 3: Nêu kĩ thuật phân chương động bộ nhớ, ưu – nhược điểm so với phân chương cố định. Lấy VD cho các chiến lược sử dụng First Fit, Best Fit, Worst Fit. Khi di chuyển chương sang vị trí khác có cần thay đổi thông tin gì trong khối ánh xạ địa chỉ không.

* Kích thước, số lượng và vị trí chương đều có thể thay đổi
* Các vùng trống bộ nhớ cũng có thể được liên kết thành một danh sách kết nối
* Khi một tiến trình yêu cầu bộ nhớ

• HĐH cấp cho tiến trình 1 chương có kích thước đúng bằng tiến trình đó đang cần

• Nếu tìm thấy vùng nhớ tương ứng

* Vùng trống được chia thành 2 phần
* Một phần cung cấp theo yêu cầu
* Một phần trả lại danh sách vùng trống tự do

• Nếu không tìm thấy

* Phải chờ tới khi có được một vùng trống thỏa mãn
* Cho phép tiến trình khác trong hàng đợi thực hiện (nếu độ ưu tiên đảm bảo)
* Khi tiến trình kết thúc sẽ tạo vùng trống trong Bộ nhớ và các vùng trống nằm cạnh nhau được nhập lại thành vùng lớn hơn
* Các chiến lược cấp chương chọn vùng chống:

• First Fit: Chọn vùng thích hợp đầu tiên

• Best Fit: Vùng thích hợp nhất

• Worst Fit: Vùng trống thích hợp nhất - vùng kích thước lớn nhất

* Ưu điểm : Phân chương động tránh việc phân mảnh trong và linh hoạt hơn phân chương cố định
* Nhược điểm: Gây phân mảnh ngoài: dồn những vùng trống nhỏ thành lớn (nén), phức tạp hơn

Ví dụ: trong bộ nhớ có 4 vùng trống có kích thước lần lượt như sau 3MB, 8MB, 7MB, và 10MB; yêu cầu cấp phát vùng nhớ kích thước 6MB. Ba chiến lược cấp phát ở trên sẽ cho kết quả như sau:

- Chiến lược first fit sẽ chọn khối 8MB để chia và cấp phát.

- Chiến lược best fit sẽ chọn vùng trống 7MB.

- Chiến lược worst fit sẽ chọn vùng trống 10MB.

## **Câu** 4: Phân tích ưu/nhược của kthuật tải trong lúc chạy và liên kết động, thư viện dùng chung

1. Tải trong quá trình chạy

* Toàn bộ chương trình thông thường được tải vào bộ nhớ để thực hiện.
* Ưu điểm: giúp giảm thời gian khởi động chương trình và tiết kiệm bộ nhớ
* Nhược điểm: Đối với chương trình lớn, trong một phiên làm việc, một số phần của chương trình có thể không dùng tới. Các hàm nãy sẽ chiếm vô ích trong bộ nhớ, đồng thời tăng thời gian tải chương trình lúc đầu.
* Giải pháp:

▶ Hàm chưa bị gọi thì chưa tải vào bộ nhớ

▶ Chương trình chính được load vào bộ nhớ và chạy

▶ Khi có lời gọi hàm:

• Chương trình sẽ kiểm tra hàm đó được tải vào chưa.

• Nếu chưa, chương trình sẽ tiến hành tải sau đó ánh xạ địa chỉ hàm vào không gian chung của chương trình và thay đổi bảng địa chỉ để ghi lại các ánh xạ đó

▶ Lập trình viên đảm nhiệm kiểm tra và tải hàm, HĐH chỉ cung cấp các hàm thư viện cho module

1. Liên kết động và Thư viện dùng chung

* Trong quá trình liên kết tĩnh các hàm và thư viện được liên kết luôn vào mã chương trình
* Kích thước chương trình = Kích thước chương trình vừa được dịch + Kích thước các hàm thư viện
* Giải quyết sử dụng kỹ thuật liên kết động : Trong giai đoạn liên kết, không kết nối các hàm thư viện vào chương trình mà chỉ chèn các thông tin về hàm thư viện đó (stub).
* Các module thư viện được liên kết trong quá trình thực hiện:

• Không giữ bản sao các module thư viện mà tiến trình giữ đoạn mã nhỏ chứa thông tin về modul thư viện

• Khi đoạn mã nhỏ được gọi, nó kiểm tra module tương ứng đã có trong bộ nhớ chưa. Nếu chưa, thì tải phần còn lại và dùng.

• Lần tiếp theo cần sử dụng, modul thư viện sẽ được chạy trực tiếp

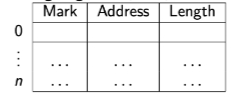
• Mỗi module thư viện chỉ có 1 bản sao duy nhất chứa trong MEM

* • Cần hỗ trợ từ HĐH
* Ưu điểm: giảm kích thước của chương trình và tiết kiệm bộ nhớ,
* Nhược điểm :Các hàm sẽ có mặt lặp đi lặp lại trong các chương trình, Lãng phí không gian cả trên đĩa và bộ nhớ trong, tăng tg truy nhập

## **Câu** 5: Cấu trúc và cách ánh xạ địa chỉ của phương pháp phân đoạn bộ nhớ, so sánh với phân trang

1. Ánh xạ địa chỉ của phương pháp phân đoạn

* Sử dụng bảng đoạn (SCB: Segement Control Block) cho mỗi tiến trình. Mỗi phần tử của bảng tương ứng với 1 đoạn, chứa:



• Dấu hiệu (Mark (0/1)): Đoạn đã tồn tại trong bộ nhớ

• Địa chỉ cơ sở (Address): Vị trí bắt đầu của đoạn trong bộ nhớ

• Độ dài đoạn (Length): Sử dụng để chống truy cập trái phép ngoàiđoạn

* Địa chỉ logic gồm 2 thành phần (s, o): s: số thứ tự, tên đoạn và o: độ dịch trong đoạn
* Địa chỉ truy nhập: <Tên (số stt) đoạn, Độ dịch trong đoạn>
* Từ chỉ số đoạn s, vào bảng đoạn, tìm địa chỉ vật lý bắt đầu của đoạn
* So sánh độ dịch o với chiều dài đoạn, nếu lớn hơn =>địa chỉ sai ->Lỗi truy cập
* Địa chỉ vật lý mong muốn là tổng của địa chỉ vật lý bắt đầu đoạn và địa chỉ lệch

1. So sánh với phân trang

|  |  |  |
| --- | --- | --- |
|  | Phân đoạn | Phân trang |
| Ưu | Kỹ thuật này linh hoạt và tận dụng tối đa bộ nhớ. | Đơn giản, dễ hiểu |
| Nhược | phức tạp hơn kỹ thuật phân tran | Không linh hoạt và lãng phí bộ nhớ |

## **Câu** 6: Phân trang bộ nhớ và cơ chế ánh xạ địa chỉ

1. Phân trang bộ nhớ

▶ Bộ nhớ vật lý được chia thành từng khối có kích thước bằng nhau: khung trang vật lý (page frame):

• Khung trang vật lý được đánh số 0,1,2,... : địa chỉ vật lý của khung trang

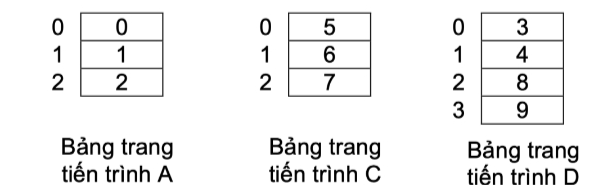
• Khung trang được dùng làm đơn vị phân phối nhớ

▶ Không gian địa chỉ logic của tiến trình hay bộ nhớ logic (chương trình) được chia thành những khối gọi là trang (page) có kích thước bằng khung trang.

▶ Tiến trình được cấp các khung để chứa các trang của mình.

▶ Trang có thể chứa trong các khung nằm rải rác trong bộ nhớ

Bảng trang: HĐH quản lý việc cấp phát khung cho mỗi tiến trình bằng bảng trang (Page table): mỗi ô tương ứng với 1 trang và chứa số khung cấp cho trang đó.

Mỗi tiến trình có bảng trang riêng

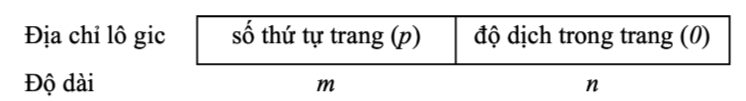
1. Ánh xạ địa chỉ

Ánh xạ địa chỉ địa chỉ logic khi phân trang sang địa chỉ vật lý.

▶ Ánh xạ địa chỉ gồm 2 phần:

• Số thứ tự trang (p)

• Độ dịch (địa chỉ lệch) của địa chỉ so với đầu trang (o)



Nếu kích thước trang là 2n. Biểu diễn địa chỉ logic dưới dạng địa chỉ có độ dài (m + n) bit

• m bit cao: biểu diễn số thứ tự trang

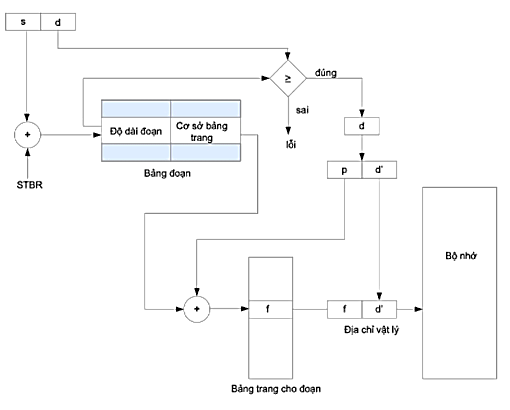
• n bit thấp: biểu diễn độ dịch trong trang nhớ

## **Câu** 7:Trình bày cơ chế kết hợp phân trang với phân đoạn , vẽ sơ đồ giải thích ánh xạ địa chỉ đó.

CƠ CHẾ KẾT HỢP PHÂN TRANG VÀ PHÂN ĐOẠN

▶ Phân đoạn chương trình, mỗi đoạn sẽ tiến hành phân trang

▶ Địa chỉ gồm: số thứ tự đoạn, số thự tự trang, độ dịch trong trang

▶ Tiến trình có 1 bảng phân đoạn, mỗi đoạn có 1 bảng phân trang

**Câu**  8: Tại sao phải đổi trang và các bước phải đổi trang (khi xảy ra tình trạng thiếu trang …)

* Khi xảy ra tình trạng thiếu trang: HĐH phải tìm ra một khung trống trong bộ nhớ đọc trang thiếu vào khung và tiến trình sau đó hoạt động bình thường.
* Bộ nhớ ảo > bộ nhớ thực và chế độ đa chương trình có lúc không còn khung nào trống để nạp trang mới
* Các bước đổi trang

Bước 1: Xác định trang trên đĩa cần nạp vào MEM

Bước 2: Nếu có khung trống trên MEM thì chuyển sang B4

Bước 3:

• Lựa chọn 1 khung trên MEM để giải phóng, theo 1 thuật toán nào đó

• Ghi nội dung khung bị đổi ra đĩa (nếu cần), cập nhật bảng trang và bảng khung

Bước 4: Đọc trang cần nạp vào khung vừa giải phóng; cập nhật bảng trang và bảng khung

Bước 5: Thực hiện tiếp tiến trình từ điểm bị dừng trước khi đổi trang

## **Câu** 9: Trình bày các thuật toán đổi trang

1. Đổi trang tối ưu (OPT):

* Chọn trang sẽ không được dùng tới trong khoảng thời gian lâu nhất trong tương lai để đổi hay trang có lần sử dụng tiếp theo xa nhất
* Cho phép giảm tối thiểu sự thiếu trang và tối ưu theo tiêu chuẩn này
* HĐH đoán trước được nhu cầu sử dụng các trang trong tương lai
* Không áp dụng trong thực tế chỉ so sánh với các chiến lược khác

1. Vào trước ra trước (FIFO):

- Trang nào được nạp vào trước thì bị đổi ra trước

- Phương pháp đơn giản nhất

- Trang bị trao đổi là trang nằm lâu nhất trong bộ nhớ

1. Đổi trang có lần sử dụng cuối cách lâu nhất (LRU):

- Trang bị đổi là trang mà thời gian từ lần truy cập cuối cùng đến thời điểm hiện tại là lâu nhất (lâu rồi không truy cập thì thay thế)

- Theo nguyên tắc cục bộ về thời gian, đó chính là trang ít có khả năng sử dụng tới nhất trong tương lai

- Thực tế LRU cho kết quả tốt gần như phương pháp đổi trang tối ưu

1. CLOCK: Thuật toán đồng hồ Cải tiến FIFO nhằm tránh thay những trang được nạp vào lâu vẫn có khả năng sử dụng

- Mỗi trang được gắn thêm 1 bit sử dụng U

• Khi trang được truy cập đọc/ ghi: U = 1

• Ngay khi trang được đọc vào bộ nhớ: U =1

- Các khung/ trang có thể bị đổi được liên kết vào 1 danh sách vòng

- Khi một trang nào đó bị đổi, con trỏ được dịch chuyển để trỏ vàotrang tiếp theo trong danh sách

- Khi có nhu cầu đổi trang, HDH kiểm tra trang đang bị trỏ tới

• Nếu U=0: trang sẽ bị đổi ngay

• Nếu U=1: đặt U=0 và trỏ sang trang tiếp theo, lặp lại quá trình

- U của tất cả các trang trong danh sách =1 con trỏ sẽ quay đúng 1vòng, đặt U của các trang =0 và trang hiện thời bị trỏ sẽ bị đổi

**Thuật toán đồng hồ cải tiến:**

- Các bước thực hiện đổi trang:

- Bước 1:

• Bắt đầu từ vị trí hiện tại của con trỏ, kiểm tra các trang

• Trang đầu tiên có U=0 và M=0 sẽ bị đổi

• Chỉ kiểm tra mà không thay đổi nội dung bit U, bit M

- Bước 2:

• Nếu quay hết 1 vòng mà không tìm được trang có U và M bằng 0 thì quét lại danh sách lần 2

• Trang đầu tiên có U=0, M=1 sẽ bị đổi

• Đặt bit U của những trang đã quét đến nhưng được bỏ qua là 0

- Nếu chưa tìm được thì lặp lại bước 1 và cả bước 2 nếu cần

## **Câu** 10: Trình bày thuật toán nạp trang theo nhu cầu của bộ nhớ ảo \_ (slide tr 34\_ Đọc thêm: kỹ thuật tăng tốc độ bảng trang và bảng trang nhiều mức ) + **Câu** 11: Trình bày thuật toán nạp trang theo như cầu của bộ nhớ. Phân tích cùng 1 lệnh có thể xảy ra nhiều sự kiện lỗi trang hay không ?

1. Nạp trang theo nhu cầu của bộ nhớ ảo

* Nạp trang theo nhu cầu dựa trên phân trang kết hợp trao đổi bộ nhớ-đĩa => Tiến trình được phân trang và các trang chứa trên đĩa
* Khi thực thi, nạp tiến trình vào MEM: chỉ nạp trang cần dùng => Các trang không nhất thiết nạp cùng một lúc
* Tiến trình gồm các trang trên đĩa và trong MEM: thêm bit P trong khoản mục bảng trang để phân biệt (P=1: đã nạp vào MEM)
* Quá trình kiểm tra và nạp trang:

• Tiến trình truy cập tới 1 trang, kiểm tra bit P. Nếu P=1, truy cập diễn ra bình thường. Nếu P=0, xảy ra sự kiện thiếu trang

* Ngắt xử lý thiếu trang:

• Bước 1: HĐH tìm 1 khung trống trong MEM

• Bước 2: Đọc trang bị thiếu vào khung trang vừa tìm được

• Bước 3: Sửa lại khoản mục tương ứng trong bảng trang: đổi bit P=1 và số khung đã cấp cho trang

• Bước 4: Khôi phục lại trạng thái của tiến trình và thực hiện tiếp lệnh bị ngắt4

* Nạp trang hoàn toàn theo nhu cầu:

• Bắt đầu một tiến trình mà không nạp bất kỳ trang nào vào bộ nhớ

• Khi con trỏ lệnh được HĐH chuyển tới lệnh đầu tiên của tiến trình để thực hiện, sự kiện thiếu trang sẽ sinh ra và trang tương ứng được nạp vào

• Tiến trình sau đó thực hiện bình thường cho tới lần thiếu trang tiếp theo

* Nạp trang trước: khác với nạp trang theo nhu cầu

• Các trang chưa cần đến cũng được nạp vào bộ nhớ

• Không hiệu quả

1. Phân tích cùng 1 lệnh có thể xảy ra nhiều sự kiện lỗi trang hay không ?

Cùng một lệnh có thể xảy ra nhiều sự kiện lỗi trang, tùy thuộc vào số lượng trang mà lệnh đó sử dụng.

* Nếu lệnh sử dụng một trang đã được nạp vào bộ nhớ, không có sự kiện lỗi trang xảy ra.
* Nếu lệnh sử dụng một trang chưa được nạp vào bộ nhớ, sẽ xảy ra một sự kiện lỗi trang và trang đó sẽ được nạp vào bộ nhớ

## **Câu** 12: Nêu kĩ thuật để tăng tốc độ truy cập bảng trang và bảng trang nhiều mức.

* Mỗi thao tác truy cập bộ nhớ đều đòi hỏi truy cập bảng phân trang => tổ chức bảng phân trang sao cho tốc độ truy cập là cao nhất
* Sử dụng tập hợp các thanh ghi làm bảng phân trang:

• Tốc độ truy cập rất cao

• Số lượng thanh ghi hạn chế => không áp dụng được

* Giữ các bảng trang trong MEM:

• Vị trí mỗi bảng được trỏ bởi thanh ghi cơ sở bảng trang PTBR (Page Table Base Register)

• Thời gian để truy cập bảng => sử dụng bộ nhớ cache tốc độ cao

**Bảng trang nhiều mức:**

* Các hệ thống tính toán hiện đại cho phép sử dụng không gian địachỉ logic lớn (232− > 264) => Số lượng trang cần quản lý tăng dẫn đến kích thước bảng trang tăng
* Nguyên tắc: Bảng quản lý trang được phân trang

• Chia bảng trang thành những phần nhỏ hơn

• Tổ chức bảng trang nhiều mức: Khoản mục của bảng mức trên chỉ tới bảng trang khác

* Bảng trang được phân trang. Số hiệu trang được chia thành 2 phần:

• P1: 10 bit cho phép định vị khoản mục trong bảng mức trên

• P2: 10 bit định vị khoản mục trong bảng mức dưới

• O: 12 bit, chứa độ dịch trong trang

* Địa chỉ truy nhập có dạng P1, P2, o

# **CHƯƠNG 4:**

## **Câu** 1: Nêu khái niệm tiến trình và so sánh sự khác nhau giữa tiến trình và chương trình.

* Tiến trình là một chương trình đang trong quá trình thực hiện.

|  |  |
| --- | --- |
| **Chương trình** | **Tiến trình** |
| Thực thể tĩnh | Thực thể động |
| Không sở hữu tài nguyên cụ thể | Được cấp một số tài nguyên để chứa tiến trình và thực hiện lệnh |

* Tiến trình sinh ra khi chương trình được tải vào bộ nhớ để:

• Thực hiện tiến trình người dùng

• Thực hiện tiến trình hệ thống

## **Câu** 2: Nêu tên các thuật toán quản lí tiến trình .

1. Thuật toán đến trước phục vụ trước (FCFS: First Come, First Served)

2. Điều độ quay vòng (RR: round robin)

3. Điều độ ưu tiên tiến trình ngắn nhất (SPF: Shortest Job First)

4. Điều độ ưu tiên thời gian còn lại ngắn nhất (SRTF: Shortest Remaining Time First)

5. Điều độ có mức ưu tiên (Priority Scheduling)

## **Câu** 3: Thao tác khi tạo mới tiến trình, và trình bày tiến trình có thể bị kết thúc trong TH nào.

1. Tạo mới tiến trình:

* Để tạo ra một tiến trình mới, OS thực hiện một số bước như sau:

• Gán số định danh cho tiến trình được tạo mới và tạo một ô trongbảng tiến trình

• Tạo không gian nhớ cho tiến trình và PCB

• Khởi tạo PCB: HĐH gán giá trị các thành phần của PCB

• Liên kết PCB của tiến trình vào các danh sách quản lý, ví dụ vào ds tiến trình đang chạy.

1. Kết thúc tiến trình:

* Kết thúc bình thường: yêu cầu HDH kết thúc mình bằng cách gọi lời gọi hệ thống exit().
* Tiến trình bị kết thúc trong các trường hợp sau:

• Bị tiến trình cha kết thúc

• Do các lỗi

• Yêu cầu nhiều bộ nhớ hơn so với số lương hệ thống có thể cung cấp

• Thực hiện lâu hơn thời gian giới hạn

• Do quản trị hệ thống hoặc hệ điều hành kết thúc

## **Câu** 4: Nêu khái niệm về dòng / luồng, thế nào là dòng ở mức nhân và ở mức người dùng

1. Khái niệm

* Mỗi đơn vị thực hiện lệnh của tiến trình, tức là 1 chuỗi lệnh được cấp phát CPU để thực hiện độc lập được gọi là một luồng thực hiện
* HĐH hiện nay thường hỗ trợ đa luồng (multithreading):

• Tiến trình có thể có nhiều luồng

• Cho phép nhiều chuỗi lệnh, thực hiện nhiều nhiệm vụ cùng một lúc

1. Luồng/ dòng ở mức nhân và mức người dùng

* Luồng mức người dùng: được tạo ra và quản lý không có sự hỗ trợ của hệ điều hành
* Luồng mức nhân: được tạo ra và quản lý bởi hệ điều hành

**Luồng ở mức người dùng:**

▶ Do trình ứng dụng tạo ra và quản lý

▶ Việc phân phối CPU được thực hiện cho cả tiến trình

▶ OS vẫn coi tiến trình như một đơn vị duy nhất với một trạng thái duy nhất

▶ Ưu điểm:

• Tiết kiệm thời gian do việc chuyển đổi luồng không đòi hỏi chuyển sang chế độ nhân

• Trình ứng dụng có thể điều độ riêng, không phụ thuộc vào cách điều độ của OS

• Có thể sử dụng khi OS không hỗ trợ đa luồng

▶ Nhược điểm

• Không tận dụng ưu điểm về tính đáp ứng của mô hình đa luồng

• Không cho phép tận dụng kiến trúc nhiều CPU

**Luồng ở mức nhân:**

▶ Luồng mức nhân được HĐH tạo ra và quản lý

▶ OS cung cấp giao diện lập trình

▶ Cung cấp các lời gọi hệ thống mà trình ứng dụng có thể yêu cầu tạo, xóa luồng và thay đổi tham số liên quan quản lý dòng

▶ Ưu điểm: Tăng tính đáp ứng và khả năng thực hiện đồng thời của các luồng trong cùng tiến trình

▶ Nhược điểm: Tốc độ chậm do tạo và chuyển đổi luồng thực hiện trong chế độ nhân

▶ HĐH Windows và Linux hỗ trợ luồng mức nhân

## **Câu** 5: Các thông tin nào được lưu trữ trong quản lí thông tin PCB.

Thông tin của tiến trình được lưu trong một cấu trúc dữ liệu gọi là khối quản lý tiến trình - PCB (Process Control Block)

Các thông tin chính trong PCB:

• Số định danh của tiến trình (PID)

• Trạng thái tiến trình: một trong năm trạng thái

• Nội dung một số thanh ghi CPU:

▶ Thanh ghi con trỏ lệnh: trỏ tới lệnh tiếp theo

▶ Thanh ghi con trỏ ngăn xếp: lưu tham số / tình trạng hàm khi thực hiện lời gọi hàm/ thủ tục của chương trình

▶ Các thanh ghi điều kiện và trạng thái

▶ Các thanh ghi đa năng

• Thông tin quản lý bộ nhớ

• Thông tin tài nguyên có thể sử dụng

## **Câu** 6: Nêu 5 trạng thái của tiến trình, vẽ sơ đồ và giải thích chuyển đổi giữa 5 trạng thái này.

Trạng thái của tiến trình là một phần trong hoạt động hiện tại của tiến trình bao gồm 5 trạng thái:

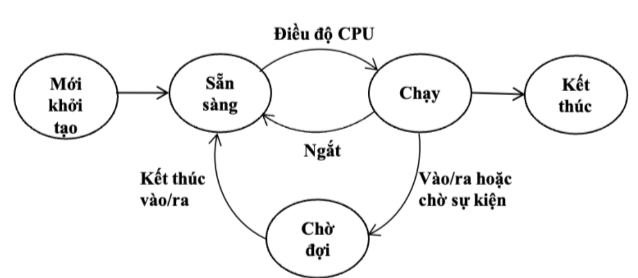
• Mới khởi tạo (New): tiến trình đang được tạo ra

• Sẵn sàng (Ready): tiến trình chờ được cấp CPU để thực hiện lệnh

• Chạy (Running): lệnh của tiến trình được CPU thực hiện

• Chờ đợi (Waiting): tiến trình chờ đợi một sự kiện nào xảy ra (blocked)

• Kết thúc (Terminated): tiến trình đã kết thúc việc thực hiện nhưng vẫn chưa bị xóa



▶ Khởi tạo -> Sẵn sàng: tiến trình khởi tạo xong và đã được tải vào bộ nhớ, chỉ chờ được cấp CPU để chạy.

▶ Sẵn sàng -> Chạy: do kết quả điều độ CPU của OS, tiến trình được OS cấp phát CPU và chuyển sang trạng thái chạy

▶ Chạy -> Sẵn sàng: OS cấp phát CPU cho tiến trình khác, do kết quả điều độ/do ngắt xảy ra, tiến trình hiện thời chuyển sang trạng thái sẵn sàng và chờ được cấp CPU để chạy tiếp.

▶ Chạy -> Chờ đợi: Tiến trình chỉ chạy khi có 1 sự kiện nào đó xảy ra, chuyển sang trạng thái chờ được phân phối CPU để chạy tiếp.

▶ Chạy -> Kết thúc: khi tiến trình đã thực hiện

## **Câu** 7: \*\* Điều độ dòng, điều độ tiến trình là gì ? 2 thuật toán có khác nhau không và giải thích?

Điều độ (scheduling) hay lập lịch là quyết định tiến trình nào được sử dụng tài nguyên phần cứng khi nào, trong thời gian bao lâu.

▶ Tập trung vào vấn đề điều độ đối với CPU là quyết định thứ tự và thời gian sử dụng CPU

▶ Điều độ tiến trình và điều độ luồng:

• Hệ thống trước kia: tiến trình là đơn vị thực hiện chính, điều độ thực hiện với tiến trình

• Hệ thống hỗ trợ luồng: luồng mức nhân là đơn vị thực hiện được HĐH cấp CPU chứ không phải tiến trình.

• Sử dụng thuật ngữ điều độ tiến trình rộng rãi tương đương điều độ luồng

* SỰ KHÁC NHAU GIỮA 2 THUẬT TOÁN
* Điều độ tiến trình: liên quan đến quản lý thực hiện các tiến trình độc lập.
* Điều độ luồng: tập trung vào quản lý và lên lịch thực hiện của các luồng bên trong một tiến trình

## **Câu** 8: Trình bày thuật toán phân phối lại và không phân phối lại.

▶ Điều độ có phân phối lại (preemptive):

• HĐH có thể sử dụng cơ chế ngắt để thu hồi CPU của một tiến trình đang trong trạng thái chạy

▶ Điều độ không phân phối lại (nonpreemptive):

• Tiến trình đang ở trạng thái chạy sẽ được sử dụng CPU cho đến khi xảy ra một trong các tình huống sau:

▶ Tiến trình kết thúc

▶ Tiến trình phải chuyển sang trạng thái chờ đợi do thực hiện I/O

• Điều độ hợp tác: do chỉ thực hiện được khi tiến trình hợp tác và nhường CPU

• Nếu tiến trình không hợp tác, dùng CPU vô hạn nên các tiến trình khác không được cấp CPU (Windows 96, NT)

## **Câu** 9: Trình bày mô hình đa luồng, vấn đề sở hữu luồng và đa luồng. Nêu ưu/nhược điểm của nó

* Mô hình đa luồng là một mô hình lập trình và thực thi được sử dụng rất phổ biến trong các hệ điều hành đa nhiệm. Mô hình này cho phép tạo ra và sử dụng nhiều luồng trong khuôn khổ một tiến trình. Mỗi luồng đều có thể thực hiện các công việc độc lập với nhau, giúp tăng hiệu năng và tiết kiệm thời gian.
* Vấn đề sở hữu luồng và đa luồng là một trong những vấn đề phổ biến nhất của mô hình đa luồng. Khi nhiều luồng cùng truy cập vào một tài nguyên chung, chúng có thể xảy ra xung đột và gây ra lỗi .
* Ưu :

▶ Tăng hiệu năng và tiết kiệm thời gian

▶ Dễ dàng chia sẻ tài nguyên và thông tin

▶ Tăng tính đáp ứng

▶ Tận dụng được kiến trúc xử lý với nhiều CPU

▶ Thuận lợi cho việc tổ chức chương trình

* Nhược điểm: xử lý chương trình phức tạp và xử lý vấn đề tranh chấp bộ nhớ, đồng bộ dữ liệu khá phức tạp

## **Câu** 10: Nêu các tiêu chí để đánh giá thuật toán điều độ ( 7 tiêu chí)

Một số tiêu chí thường được sử dụng:

▶ Lượng tiến trình được thực hiện xong:

• Số lượng tiến trình thực hiện xong trong 1 đơn vị thời gian

• Đo tính hiệu quả của hệ thống

▶ Hiệu suất sử dụng CPU: Cố gắng để CPU càng ít phải nghỉ càng tốt

▶ Thời gian vòng đời trung bình của tiến trình: Từ lúc có yêu cầu tạo tiến trình đến khi kết thúc

▶ Thời gian chờ đợi:

• Tổng thời gian tiến trình nằm trong trạng thái sẵn sàng và chờ cấp CPU

• Ảnh hưởng trực tiếp của thuật toán điều độ tiến trình

▶ Thời gian đáp ứng: Đây là tiêu chí hướng tới người dùng và thường được sử dụng trghệ thống tương tác trực tiếp.

▶ Tính dự đoán được: Vòng đời, thời gian chờ đợi, thời gian đáp ứng phải ổn định, không phụ thuộc vào tải của hệ thống

▶ Tính công bằng: Các tiến trình cùng độ ưu tiên phải được đối xử như nhau

## **Câu** 11: Trình bày các giải pháp phần cứng cho vấn đề loại trừ tương hỗ và đoạn nguy hiểm.

▶ Phần cứng máy tính có thể được thiết kế để giải quyết vấn đề loại trừ trong tương hỗ và đoạn nguy hiểm.

▶ Giải pháp phần cứng thường dễ sử dụng và có tốc độ tốt.

▶ Cấm các ngăt: cấm không nể xẩy ra ngắt trong thời gian tiến trình đang ở trong đoạn nguy hiểm nể truy cập tài nguyên.

▶ Sử dụng lệnh máy đặc biệt: Phần cứng được thiết kế có một số lệnh máy đặc biệt

• Hai thao tác kiểm tra và thay đổi giá trị cho một biến (ô nhớ) kiểm tra và xác lập Test\_and\_Set, hoặc các thao tác so sánh và hoán đổi giá trị hai biến, được thực hiện trong cùng một lệnh máy

• Đảm bảo được thực hiện cùng nhau mà không bị xen vào giữa – thao tác nguyên tử (atomic)

## **Câu** 12:Sử dụng Test and Set cho bài toán triết gia ăn cơm và giải pháp này có gây ra vấn đề bế tắc hay đói không ?

Giải pháp này có thể xảy ra tình trạng bế tắc như sau:

* B1: Tất cả các triết gia đều thử kiểm tra và giữ đầu tiên đũa bên trái của mình.
* B2: Sau đó, mỗi triết gia thử kiểm tra và giữ đầu tiên đũa bên phải của mình.

Nếu tất cả triết gia thực hiện bước 1 rồi mới đến bước 2, có thể xảy ra bế tắc, vì mỗi triết gia đang giữ một đầu đũa mà cần một đầu khác để bắt đầu ăn.

Để tránh vấn đề bế tắc, cần phải thực hiện các biện pháp như:

* Thứ Tự Ưu Tiên: Đặt một thứ tự ưu tiên cho việc kiểm tra và giữ đũa, đảm bảo chỉ có một triết gia có thể giữ đồng thời cả hai đầu.
* Giới Hạn Số Lượng Triết Gia Cùng Lúc: Hạn chế số lượng triết gia có thể thực hiện bước kiểm tra và giữ cùng một lúc để giảm khả năng xảy ra vấn đề bế tắc.
* Giải Pháp Ưu Tiên Cho Triết Gia Đói: Cho phép triết gia giữ đầu tiên đũa bên trái, sau đó kiểm tra và giữ đầu tiên đũa bên phải. Nếu không giữ được cả hai, thả đầu tiên đũa bên trái và thử lại

## **Câu** 13: Phương pháp cờ hiệu Semanphore cho vấn đề loại trừ tương hỗ và đoạn nguy hiểm.

* Semaphore: Là một cấu trúc dữ liệu cờ hiệu (flag) được sử dụng để đồng bộ hóa quy trình và tránh tình trạng đoạn nguy hiểm.
* Vấn Đề Loại Trừ Tương Hỗ: Semaphore có thể được sử dụng để kiểm soát việc truy cập vào các tài nguyên chia sẻ, đảm bảo rằng chỉ có một tiến trình hoặc luồng được phép sử dụng tài nguyên tại một thời điểm.
* Vấn Đề Đoạn Nguy Hiểm: Semaphore cũng có thể được sử dụng để giải quyết vấn đề đoạn nguy hiểm, nơi mà nhiều tiến trình hoặc luồng có thể cùng một lúc ghi hoặc đọc dữ liệu mà không gây xung đột.