BUỔI 1: TỔNG QUAN VỀ LẬP TRÌNH HỆ THỐNG

(Tutorial / Lab)

MÔN HỌC (SUBJECTS): Lập trình hệ thống – Systems Programming

CHUYÊN NGÀNH (MINOR): Tất cả các chuyên ngành – for all bachelor programs

---

Câu 1 (Nhận biết – Lý thuyết)

Hãy trình bày khái niệm lập trình hệ thống và nêu vai trò của lập trình hệ thống trong phát triển phần mềm. Theo bạn, vì sao lập trình hệ thống lại quan trọng đối với phát triển ứng dụng?

Trả lời:

Lập trình hệ thống là thiết kế và viết các phần mềm máy tính có khả năng làm việc trực tiếp với các thành phần cốt lõi của hệ thống máy tính như hệ điều hành, bộ nhớ, CPU, tiến trình, thiết bị vào/ra và các tài nguyên hệ thống khác. Mục tiêu chính của nó là tạo ra nền tảng, dịch vụ để các phần mềm ứng dụng khác có thể chạy trên đó.

Lập trình hệ thống giữ vai trò nền tảng trong toàn bộ quá trình phát triển phần mềm, vì mọi ứng dụng đều phụ thuộc vào các dịch vụ do hệ thống cung cấp để có thể chạy ổn định và hiệu quả. Nhờ lập trình hệ thống, tài nguyên như CPU, bộ nhớ và thiết bị vào/ra được quản lý và phân phối hợp lý, giúp ứng dụng đạt hiệu năng cao, tránh xung đột và lỗi nghiêm trọng. Bên cạnh đó, lập trình hệ thống còn đảm bảo tính an toàn, ổn định và khả năng mở rộng của môi trường thực thi, đồng thời giúp lập trình viên hiểu sâu cơ chế bên dưới.

Theo em lập trình hệ thống quan trọng vì nó giúp lập trình viên hiểu rõ cách ứng dụng thực sự hoạt động trên máy tính, thay vì chỉ sử dụng thư viện hay framework một cách máy móc. Khi nắm được cách hệ điều hành quản lý bộ nhớ, CPU, tiến trình và luồng, lập trình viên có thể viết ứng dụng tối ưu hơn, ít lỗi hơn và dễ mở rộng hơn.

Câu 2 (So sánh – Hiểu)

So sánh lập trình hệ thống và lập trình ứng dụng theo các tiêu chí:

1. Mức độ tương tác với hệ điều hành
2. Quản lý tài nguyên, dữ liệu
3. Yêu cầu về hiệu năng và độ an toàn
4. Nêu ví dụ minh họa cho mỗi loại.

Trả lời

Tiêu chí	**Lập trình hệ thống**	**Lập trình ứng dụng**
Mức độ tương tác với hệ điều hành	Rất cao. Làm việc trực tiếp với kernel, driver, và các lời gọi hệ thống	Thấp. Tương tác thông qua các thư viện, API hoặc Framework do hệ thống cung cấp.
Quản lý tài nguyên, dữ liệu	Thủ công. Lập trình viên phải tự quản lý bộ nhớ, quản lý luồng xử lý và I/O cấp thấp.	Tự động. Thường dựa vào Garbage Collector hoặc các lớp quản lý dữ liệu cấp cao của framework.
Yêu cầu về hiệu năng và độ an toàn	Phải tối ưu từng byte bộ nhớ. Lỗi nhỏ có thể làm treo cả hệ thống.	Hiệu năng quan trọng nhưng không khắt khe bằng. Lỗi thường chỉ làm tắt ứng dụng, không sập máy.
Nêu ví dụ minh họa cho mỗi loại	Trình biên dịch (Compiler), Hệ điều hành (Windows, Linux), Device Driver, Web Server (Nginx).	Microsoft Word, Trình duyệt Chrome, Game Flappy Bird, Ứng dụng quản lý bán hàng.

Câu 3 (Phân tích – Hiểu)

Giải thích mối quan hệ giữa hệ điều hành và chương trình ứng dụng.

Theo bạn, tại sao ứng dụng không thể làm việc trực tiếp với phần cứng mà phải thông qua hệ điều hành?

Trả lời:

Hệ điều hành là lớp phần mềm trung gian nằm giữa phần cứng và chương trình ứng dụng. Nó chịu trách nhiệm quản lý và điều phối toàn bộ tài nguyên hệ thống như CPU, bộ

nhớ, thiết bị vào/ra và lưu trữ, đồng thời cung cấp các dịch vụ và API để ứng dụng có thể thực thi. Chương trình ứng dụng không truy cập trực tiếp vào phần cứng mà gửi các yêu cầu đến hệ điều hành sau đó hệ điều hành sẽ xử lý, phân quyền và thực hiện các thao tác cần thiết trên phần cứng. Nhờ mối quan hệ này, nhiều ứng dụng có thể chạy đồng thời trên cùng một hệ thống mà vẫn đảm bảo tính ổn định và an toàn.

Theo em nếu ứng dụng làm việc trực tiếp với phần cứng thì hệ thống sẽ rất dễ mất kiểm soát, xảy ra xung đột tài nguyên và gây lỗi nghiêm trọng. Hệ điều hành đóng vai trò kiểm soát và bảo vệ, đảm bảo mỗi ứng dụng chỉ sử dụng tài nguyên trong phạm vi được cấp phép, tránh ảnh hưởng đến các ứng dụng khác và toàn bộ hệ thống.

Câu 4 (Vận dụng – Thực hành ngắn)

Sử dụng C#/.NET, hãy viết một chương trình đơn giản để:

In ra thông tin cơ bản về môi trường hệ thống (phiên bản hệ điều hành, thư mục hiện tại, thời gian hệ thống).

Sinh viên giải thích chương trình đang sử dụng dịch vụ nào của hệ điều hành.

Trả lời:

Link repository: [SystemsProgramming_Exercises/Tutorial_01]

Giải thích chương trình:

Environment.OSVersion: Gọi xuống API của hệ điều hành để truy xuất thông tin về nền tảng và phiên bản kernel đang chạy, bao gồm loại hệ điều hành và phiên bản tương ứng.

RuntimeInformation.FrameworkDescription: Truy xuất thông tin về môi trường runtime (.NET/CLR) đang thực thi ứng dụng trên nền hệ điều hành.

Environment.Is64BitOperatingSystem: Kiểm tra kiến trúc của hệ điều hành (32-bit hay 64-bit) do hệ điều hành cung cấp.

Environment.Is64BitProcess: Xác định kiến trúc của tiến trình hiện tại, cho biết ứng dụng đang chạy ở chế độ 32-bit hay 64-bit.

Environment.ProcessorCount: Lấy số lượng lõi xử lý logic mà hệ điều hành cấp phát cho tiến trình.

GC.GetTotalMemory(): Truy xuất lượng bộ nhớ managed heap đang được runtime cấp phát và quản lý cho ứng dụng.

Environment.CurrentDirectory: Lấy thư mục làm việc hiện tại của tiến trình do hệ điều hành thiết lập khi khởi chạy chương trình.

DateTime.Now: Lấy thời gian hệ thống theo múi giờ cục bộ từ đồng hồ của hệ điều hành.

DateTime.UtcNow: Lấy thời gian hệ thống theo chuẩn UTC, độc lập với múi giờ cục bộ.

Câu 5 (Liên hệ – Thảo luận)

Hãy nêu 3 ví dụ bạn cần sử dụng lập trình hệ thống trong quá trình phát triển ứng dụng hoặc phần mềm. Giải thích vì sao các ứng dụng đó cần làm việc gần với hệ điều hành và quản lý tài nguyên hệ thống.

Trả lời:

Ứng dụng quản lý tiến trình và tài nguyên hệ thống: Các ứng dụng như Task Manager cần truy xuất trực tiếp thông tin về tiến trình, CPU, bộ nhớ và trạng thái hoạt động của hệ thống. Để làm được điều này, chương trình phải làm việc gần với hệ điều hành nhằm gọi các API hệ thống để liệt kê tiến trình, đo mức sử dụng tài nguyên và điều khiển việc kết thúc hoặc ưu tiên tiến trình.

Phần mềm sao lưu dữ liệu: Phần mềm sao lưu dữ liệu cần sử dụng lập trình hệ thống vì phải làm việc trực tiếp với hệ điều hành để truy cập hệ thống tệp, đọc/ghi dữ liệu dung lượng lớn và quản lý quyền truy cập file. Ứng dụng phải gọi các API hệ thống để kiểm soát tiến trình sao lưu, xử lý song song nhiều tệp, theo dõi trạng thái ổ đĩa và đảm bảo dữ liệu không bị xung đột khi các chương trình khác đang sử dụng. Việc làm việc gần với hệ điều hành giúp phần mềm sao lưu hoạt động ổn định, hiệu quả và đảm bảo an toàn dữ liệu.

Ứng dụng xử lý tệp tin và hệ thống lưu trữ: Các phần mềm quản lý tệp tin hoặc sao lưu dữ liệu cần làm việc trực tiếp với hệ thống tệp do hệ điều hành quản lý. Lập trình hệ thống cho phép ứng dụng truy cập, đọc/ghi, phân quyền và kiểm soát trạng thái của tệp tin và thư mục một cách an toàn và hiệu quả. Việc làm việc gần với hệ điều hành giúp ứng dụng đảm bảo tính toàn vẹn dữ liệu, xử lý các tệp lớn và tối ưu hiệu năng truy xuất ổ đĩa.