# Đề cương ôn tập PTTT HĐT

## Chương 3:

1. Biểu đồ ca sử dụng: Hệ thống của chúng ta sẽ tương tác với thế giới

bên ngoài như thế nào?

Hệ thống của chúng ta có thể tương tác với thế giới bên ngoài thông qua giao diện người dùng, các API công cộng, giao thức web, nhận diện giọng nói, gửi và nhận email, và kết nối cơ sở dữ liệu.

2. Biểu đồ giao tiếp: Các đối tượng sẽ tương tác với nhau như thế nào?

Bằng cách gửi và nhận thông điệp

3. Biểu đồ tuần tự: Các đối tượng sẽ tương tác với nhau như thế nào?

 Tác dụng tương tự như biểu đồ giao tiếp, nhưng rõ hơn về thứ tự thực

hiện của các thông điệp

4. Biểu đồ máy trạng thái: Các đối tượng có các trạng thái nào?

 Một số đối tượng, trong thời điểm nào đó có một trạng thái nhất định

5. Biểu đồ gói: Chúng ta mô-đun hóa quá trình phát triển của mình như

thế nào?

 Tương tự như “thư mục” trong quản lý tệp tin

 Có thể có các gói các ca sử dụng, gói các lớp (khi phân tích một hệ

thống lớn)

6. Biểu đồ thành phần: Các thành phần phần mềm của chúng ta sẽ liên

quan với nhau như thế nào?

 Mô tả sự phụ thuộc giữa các thành phần phần mềm (files, link libraries,

executables)

7. Biểu đồ triển khai: Phần mềm sẽ được triển khai như thế nào?

 Mô tả về cách phần mềm của chúng ta sẽ được triển khai

## Chương 4:

1. Vòng đời phát triển hệ thống (system development life cycle)

là gì?

Vòng đời phát triển hệ thống (SDLC) là quy trình quản lý và tổ chức để phát triển, triển khai và duy trì hệ thống phần mềm hoặc hệ thống thông tin. Nó bao gồm các giai đoạn như phân tích yêu cầu, thiết kế, xây dựng, kiểm thử và triển khai.

2. Những pha chính trong phát triển hệ thống hướng đối tượng?

Khởi đầu, Hoạch định, Xây dựng, Chuyển giao

3. Trong phát triển hướng đối tượng, các pha (phases) và các

bước thực hiện (workflows) có liên quan thế nào?

Mối quan hệ giữa chúng là: mỗi pha sẽ bao gồm một hoặc nhiều workflows cần được thực hiện. Ví dụ, trong pha Khởi đầu, có thể cần thực hiện các workflows như xác định yêu cầu và phân tích rủi ro. Trong khi đó, pha Xây dựng sẽ tập trung vào các workflows như cài đặt và kiểm thử

4. Nêu sự khác nhau giữa vòng đời (life cycle) và phương pháp

phát triển (development method)?

Vòng đời phần mềm (Software Development Life Cycle - SDLC) là một quy trình bao gồm các giai đoạn liên tiếp từ khi bắt đầu dự án cho đến khi sản phẩm được triển khai và bảo trì. SDLC giúp định hình cách thức tổ chức quá trình phát triển, xác định rủi ro tiềm ẩn, và giải quyết vấn đề. Các giai đoạn của SDLC thường bao gồm: lập kế hoạch, phân tích yêu cầu, thiết kế, cài đặt, kiểm thử, triển khai và bảo trì1.

Phương pháp phát triển phần mềm là cách tiếp cận hoặc khuôn khổ được sử dụng để hướng dẫn quá trình phát triển phần mềm. Các phương pháp này có thể bao gồm Agile, Waterfall, V-Shaped, Iterative, Spiral, và nhiều phương pháp khác. Mỗi phương pháp có những đặc điểm riêng biệt về cách thức tổ chức công việc, giao tiếp giữa các thành viên trong đội ngũ, và cách thức giải quyết vấn đề12.

Vì vậy, SDLC là một quy trình tổng quát mô tả từng bước để tạo ra một sản phẩm phần mềm, trong khi phương pháp phát triển là cách tiếp cận cụ thể để thực hiện quy trình đó. Phương pháp phát triển có thể thay đổi tùy thuộc vào loại dự án, kích thước đội ngũ, và các yếu tố khác, nhưng SDLC vẫn là cấu trúc cơ bản mà tất cả các phương pháp phát triển đều tuân theo.

5. UML và RUP là gì?

UML (Unified Modeling Language) là một ngôn ngữ mô hình hóa thống nhất dùng để biểu diễn và mô tả các khía cạnh khác nhau của hệ thống hoặc phần mềm. UML cung cấp một bộ sơ đồ để hiển thị kiến trúc, hành vi, quy trình làm việc, và tương tác giữa các thành phần của hệ thống.

RUP (Rational Unified Process) là một quy trình phát triển phần mềm từ Rational, một bộ phận của IBM. RUP chia quá trình phát triển thành bốn giai đoạn riêng biệt: mô hình kinh doanh, phân tích và thiết kế, thực hiện, kiểm tra và triển khai. RUP là một cách tiếp cận hướng đối tượng và nhấn mạnh sự phát triển lặp đi lặp lại và tăng dần, sử dụng mô hình UML trong quá trình thiết kế

## Chương 5:

1. Những bước chính để xác định yêu cầu?

**Các bước chính để xác định yêu cầu** trong quá trình phát triển hệ thống (SDLC) bao gồm:

1. **Thu thập và phân tích yêu cầu**:
   * Giai đoạn này tập trung vào việc tìm câu trả lời cho câu hỏi “Các vấn đề hiện tại là gì?”.
   * Các cuộc họp với khách hàng hoặc các bên liên quan để thảo luận về nhu cầu, mục tiêu và kỳ vọng.
   * Xác định yêu cầu chi tiết và chính xác.
2. **Nghiên cứu khả thi**:
   * Đánh giá khả năng thực hiện dự án.
   * Xác định các rủi ro và cơ hội.
   * Đưa ra bức tranh rõ ràng về phạm vi và mục tiêu của dự án.
3. **Thiết kế**:
   * Mô tả chi tiết các chức năng và hoạt động của hệ thống.
   * Xác định cấu trúc dữ liệu và giao diện người dùng.
4. **Mã hóa**:
   * Chuyển đổi thiết kế thành mã nguồn.
   * Xây dựng các chức năng và tính năng.
5. **Thử nghiệm**:
   * Kiểm tra phần mềm ở nhiều mức độ khác nhau.
   * Đảm bảo tính chính xác và hiệu suất.
6. **Cài đặt / Triển khai**:
   * Triển khai hệ thống vào môi trường thực tế.
   * Đào tạo người dùng.
7. **Bảo trì**:
   * Cải tiến liên tục sau khi triển khai.
   * Đảm bảo hoạt động ổn định của hệ thống.

2. Những tài liệu nào cần trước và sau buổi phỏng vấn?

**Trước Buổi Phỏng Vấn**:

1. **Bản Mô Tả Công Việc (Job Description)**: Để hiểu rõ về vị trí và yêu cầu công việc.
2. **Danh Sách Câu Hỏi Phỏng Vấn**: Câu hỏi được chuẩn bị sẵn để đảm bảo rằng tất cả thông tin cần thiết được thu thập.
3. **Thông Tin Về Dự Án/Ứng Dụng**: Bao gồm các tài liệu hiện có, báo cáo, và mô tả về hệ thống hoặc ứng dụng cần phân tích.
4. **Hồ Sơ Người Dùng (User Profiles)**: Để hiểu rõ về người dùng cuối và nhu cầu của họ.
5. **Tài Liệu Nền Tảng**: Bất kỳ tài liệu nào cung cấp kiến thức cơ bản về lĩnh vực hoặc công nghệ liên quan.

**Sau Buổi Phỏng Vấn**:

1. **Biên Bản Phỏng Vấn (Interview Transcript)**: Ghi chép chi tiết về cuộc phỏng vấn.
2. **Bản Tóm Tắt Yêu Cầu (Requirements Summary)**: Tóm tắt các yêu cầu đã được xác định từ buổi phỏng vấn.
3. **Danh Sách Các Vấn Đề Cần Làm Rõ**: Bất kỳ điểm nào còn mơ hồ hoặc cần thêm thông tin.
4. **Kế Hoạch Tiếp Theo**: Bao gồm các bước tiếp theo, như phỏng vấn thêm người dùng hoặc phân tích thêm tài liệu.

3. Khi nào thì nên sử dụng bảng hỏi?

Khi không khai thác được thông tin theo chiều rộng

4. Các phần chính của đặc tả yêu cầu?

Đặc tả yêu cầu phần mềm (Software Requirements Specification - SRS) là một tài liệu quan trọng trong quá trình phát triển phần mềm, bao gồm các phần chính sau:

1. **Giới thiệu**: Phần này cung cấp cái nhìn tổng quan về tài liệu, bao gồm mục đích, phạm vi của hệ thống, định nghĩa, viết tắt và tài liệu tham khảo.
2. **Mô tả chung**: Đây là phần mô tả các yếu tố chung của hệ thống như ngữ cảnh hệ thống, chức năng chính, ràng buộc của hệ thống, và các giả định và phụ thuộc.
3. **Yêu cầu chức năng**: Phần này liệt kê chi tiết các chức năng cụ thể mà hệ thống phải thực hiện.
4. **Yêu cầu phi chức năng**: Bao gồm các yêu cầu về hiệu suất, bảo mật, khả năng mở rộng, độ tin cậy, và các yêu cầu khác không liên quan trực tiếp đến chức năng cụ thể của hệ thống.
5. **Giao diện người dùng và giao diện hệ thống**: Mô tả cách thức tương tác giữa người dùng và hệ thống, cũng như giao diện với các hệ thống khác.
6. **Ràng buộc thiết kế**: Định rõ các ràng buộc về thiết kế mà hệ thống phải tuân theo, bao gồm các tiêu chuẩn áp dụng, quy định pháp lý và hạn chế công nghệ.
7. **Thuộc tính chất lượng**: Chi tiết các yêu cầu về chất lượng như khả năng bảo trì, khả năng chuyển giao, khả năng sử dụng và khả năng thích ứng.
8. [**Phụ lục và chỉ mục**: Cung cấp thông tin bổ sung và tài liệu tham khảo cần thiết cho việc hiểu rõ hơn về các yêu cầu](https://tigosoftware.com/vi/dac-ta-yeu-cau-phan-mem).

## Chương 6:

1. Ca sử dụng được xây dựng trong những pha nào?

Ca sử dụng thường được xây dựng trong giai đoạn phân tích yêu cầu, thiết kế, mã hóa, và thử nghiệm của quy trình phát triển phần mềm. Chúng giúp xác định và giao tiếp các yêu cầu phần mềm một cách hiệu quả.

2. Mô hình ca sử dụng bao gồm những gì?

 Các biểu đồ ca sử dụng (use case diagrams), trong đó bao gồm các

ca sử dụng (use cases) và các tác nhân (actors), quan hệ ca sử dụng, phạm vi hệ thống

3. Nêu các cách để xác định các ca sử dụng?

Để xác định ca sử dụng:

1. Hiểu rõ ngữ cảnh hệ thống.
2. Xác định các tác nhân
3. Xác định các Usecase
4. Mô tả mối quan hệ giữa tác nhân và use case.
5. Chi tiết hóa từng ca sử dụng và tác nhân.

4. “Chức năng” như thế nào thì được coi là một “ca sử dụng”?

Mục tiêu rõ ràng, Tương tác người dùng, Kết quả đo lường được, độc lập, tính trừu tượng, phản ánh nghiệp vụ, có thể kiểm thử.

Như vậy, một “chức năng” có thể được coi là một “ca sử dụng” khi nó đáp ứng các tiêu chí trên và mô tả một quy trình hoàn chỉnh từ góc độ người dùng, không chỉ là một tác vụ đơn lẻ

5. Nêu mối quan hệ giữa kịch bản và ca sử dụng?

**Mối Quan Hệ**:

* Mỗi ca sử dụng có thể có nhiều kịch bản, bao gồm cả kịch bản thành công (happy path) và các kịch bản thất bại hoặc ngoại lệ.
* Kịch bản là cách thể hiện cụ thể của ca sử dụng, giúp làm rõ cách người dùng có thể tương tác với hệ thống để thực hiện một chức năng.
* Ca sử dụng cung cấp khung chung và mục tiêu, trong khi kịch bản cung cấp chi tiết về cách mục tiêu đó được đạt được thông qua các bước cụ thể.

6. Mô tả rút gọn ca sử dụng (high-level use case description) là gì?

Mô tả rút gọn ca sử dụng, hay còn gọi là high-level use case description, là một phương pháp trình bày đơn giản và súc tích các yêu cầu chức năng của hệ thống từ quan điểm của người sử dụng. Nó không đi vào chi tiết kỹ thuật hoặc các bước cụ thể, mà tập trung vào mục tiêu chính và kết quả mong muốn của một chức năng.

7. Mô tả đầy đủ ca sử dụng (expanded use case description) cần có

những mục gì? Khi nào cần mô tả đầy đủ?

1. Tên ca sử dụng

2. Các tác nhân liên hệ với ca sử dụng này

3. Mục tiêu (Goal)

4. Mô tả tổng quan (Overview)

5. Các yêu cầu tham khảo (Cross-reference)

6. Tiền điều kiện (Pre-condtion)

7. Dòng sự kiện chính (Typical course of events)

8. Các dòng sự kiện phụ (Alternative courses)

9. Hậu điều kiện (Post-condtion)

Mô tả đầy đủ ca sử dụng thường được sử dụng khi cần:

Chi Tiết Hóa Yêu Cầu: Để cung cấp thông tin chi tiết cho các bên liên quan, bao gồm cả nhóm phát triển và kiểm thử.

Phân Tích và Thiết Kế Hệ Thống: Khi xây dựng mô hình hệ thống và thiết kế cơ sở dữ liệu.

Tài Liệu Tham Khảo: Để làm tài liệu tham khảo trong suốt quá trình phát triển và bảo trì hệ thống.

Giao Tiếp: Khi cần truyền đạt thông tin rõ ràng và đầy đủ giữa các bên liên quan

8. Trình bày về các loại quan hệ giữa các ca sử dụng? Cho một ví dụ.

Quan hệ bao gồm (include), quan hệ mở rộng (extend).

**Ví dụ**: Giả sử chúng ta có một hệ thống ngân hàng trực tuyến với các ca sử dụng sau:

* “Kiểm tra Số Dư” (Check Balance)
* “Rút Tiền” (Withdraw Money)
* “In Biên Lai” (Print Receipt)

Trong trường hợp này:

* Ca sử dụng “Rút Tiền” có thể **bao gồm** ca sử dụng “Kiểm tra Số Dư” để đảm bảo người dùng có đủ tiền trước khi rút.
* Ca sử dụng “Rút Tiền” có thể **mở rộng** bằng ca sử dụng “In Biên Lai”, nơi người dùng có tùy chọn in biên lai sau khi rút tiền.

Chương 7:

1.Mục đích của biểu đồ hành động là gì?

Mục đích của biểu đồ hành động:

* Mô tả cách mà các đối tượng trong hệ thống tương tác với nhau để thực hiện một chức năng cụ thể.
* Đưa ra cái nhìn tổng quan về luồng làm việc của hệ thống.
* Hiểu rõ quá trình hoạt động của hệ thống và cách mà các thành phần tương tác với nhau.

2.Những loại quá trình nào có thể được mô tả bằng biểu đồ hành động

Biểu đồ hành động có thể được sử dụng để mô tả một loạt các quá trình và hoạt động trong hệ thống phần mềm. Dưới đây là một số loại quá trình phổ biến mà biểu đồ hành động có thể mô tả:

* Luồng làm việc của hệ thống: Biểu đồ hành động có thể mô tả các bước và tương tác giữa các đối tượng trong hệ thống để thực hiện một nhiệm vụ cụ thể hoặc một quy trình kinh doanh.
* Hành động trong từng kịch bản của ca sử dụng: Biểu đồ hành động có thể mô tả các hành động và tương tác giữa các đối tượng khi một ca sử dụng cụ thể diễn ra, giúp hiểu rõ quy trình xử lý cho mỗi tình huống cụ thể.
* Chi tiết hoạt động của một chức năng: Biểu đồ hành động cung cấp cái nhìn chi tiết về cách mà một chức năng hoặc tính năng cụ thể của hệ thống hoạt động, bao gồm các bước cụ thể và tương tác giữa các đối tượng.
* Thuật toán phức tạp: Biểu đồ hành động có thể được sử dụng để mô tả các thuật toán phức tạp, giúp diễn giải logic và quy trình của chúng một cách rõ ràng và dễ hiểu.
* Tương tác giữa các thành phần hệ thống: Biểu đồ hành động có thể mô tả cách mà các thành phần khác nhau của hệ thống tương tác với nhau để đạt được một mục tiêu cụ thể.

3. So sánh biểu đồ hành động và sơ đồ khối.

Biểu đồ hành động và sơ đồ khối là hai loại biểu đồ phổ biến được sử dụng trong phát triển phần mềm, nhưng chúng được sử dụng cho mục đích khác nhau và có các đặc điểm khác nhau. Dưới đây là một so sánh giữa hai loại biểu đồ này:

1. Mục đích:

* Biểu đồ hành động: Sử dụng để mô tả cách mà các đối tượng trong hệ thống tương tác với nhau để thực hiện một chức năng cụ thể.
* Sơ đồ khối: Sử dụng để mô tả cấu trúc của hệ thống và các thành phần chính của nó, bao gồm các khối, liên kết giữa chúng và dòng dữ liệu.

1. Phạm vi:

* Biểu đồ hành động: Tập trung vào quá trình thực hiện một chức năng cụ thể trong hệ thống.
* Sơ đồ khối: Tập trung vào cấu trúc tổng thể của hệ thống và các thành phần chính của nó.

1. Mức độ chi tiết:

* Biểu đồ hành động: Thường chứa các thông tin chi tiết về cách mà các đối tượng tương tác với nhau, bao gồm các hoạt động, phương thức và thông điệp.
* Sơ đồ khối: Thường là một biểu đồ tổng quan với mức độ chi tiết thấp, chỉ hiển thị cấu trúc tổng thể của hệ thống.

1. Dạng biểu diễn:

* Biểu đồ hành động: Thường được biểu diễn dưới dạng các đối tượng và mối quan hệ giữa chúng, thường trong ngữ cảnh của một kịch bản hoặc một phần của hệ thống.
* Sơ đồ khối: Thường được biểu diễn dưới dạng các hình khối đại diện cho các thành phần và các liên kết giữa chúng, thường được sắp xếp một cách trực quan và tổ chức.

1. Ứng dụng:

* Biểu đồ hành động: Thường được sử dụng trong quá trình phân tích và thiết kế hướng đối tượng để hiểu rõ quá trình hoạt động của hệ thống và tương tác giữa các đối tượng.
* Sơ đồ khối: Thường được sử dụng trong quá trình thiết kế hệ thống để mô tả cấu trúc tổng thể của hệ thống và tổ chức các thành phần của nó.

4.Những yếu tố nào trong hầu hết các ngôn ngữ lập trình có thể được mô tả trong biểu đồ hành động.

Trong hầu hết các ngôn ngữ lập trình, có một số yếu tố chính có thể được mô tả trong biểu đồ hành động. Dưới đây là một số yếu tố quan trọng thường xuất hiện trong biểu đồ hành động:

1. Các đối tượng (Objects): Các đối tượng trong biểu đồ hành động thường là các thực thể trong hệ thống, có thể là đối tượng hướng đối tượng, giao diện người dùng, hoặc thậm chí là các thực thể từ cơ sở dữ liệu.
2. Thuộc tính của đối tượng (Object Attributes): Các thuộc tính của đối tượng được mô tả để biểu thị các đặc điểm của đối tượng, thường được liên kết với các giá trị cụ thể.
3. Phương thức của đối tượng (Object Methods): Các phương thức của đối tượng là các hành động mà đối tượng có thể thực hiện, và chúng thường được mô tả trong biểu đồ hành động để biểu thị cách mà đối tượng tương tác với nhau.
4. Thông điệp (Messages): Thông điệp là cách mà các đối tượng gửi và nhận thông tin từ nhau trong quá trình thực hiện chức năng cụ thể. Các thông điệp thường biểu diễn các hành động hoặc tương tác giữa các đối tượng.
5. Hoạt động (Activities): Hoạt động là các bước cụ thể hoặc hành động được thực hiện bởi các đối tượng trong quá trình thực hiện một chức năng cụ thể. Chúng có thể bao gồm các hành động đơn giản như gán giá trị cho biến, gọi phương thức, hoặc thậm chí là các hoạt động phức tạp hơn như thuật toán.
6. Trạng thái (States): Trạng thái là các điều kiện hoặc tình trạng mà một đối tượng có thể tồn tại trong quá trình thực hiện chức năng, và chúng có thể được biểu diễn trong biểu đồ hành động để mô tả luồng đi của quá trình.
7. Điều kiện (Conditions): Điều kiện là các điều kiện hoặc ràng buộc mà cần được kiểm tra để quyết định các hành động hoặc luồng đi cụ thể trong quá trình thực hiện chức năng.

Chương 8.

1.Các đặc trưng của 1 đổi tượng.

* Trạng thái (State): Đối tượng có thể có các trạng thái khác nhau trong quá trình thực hiện của chương trình. Trạng thái của một đối tượng thường được xác định bởi các thuộc tính của nó và các giá trị của thuộc tính đó tại một thời điểm cụ thể.
* Thuộc tính (Attributes hoặc Properties): Đây là các đặc điểm của đối tượng, mô tả trạng thái của đối tượng. Ví dụ, nếu bạn có một đối tượng "xe hơi", các thuộc tính có thể bao gồm "màu sắc", "số lượng bánh xe", "tốc độ hiện tại", vv.
* Ứng xử (Behavior): Thể hiện các hành động của đối tượng.
* Định danh (Identity): Mỗi đối tượng trong chương trình có một định danh duy nhất, cho phép chương trình phân biệt giữa các đối tượng khác nhau. Định danh này thường được gán cho đối tượng khi nó được tạo ra.
* Tính kế thừa (Inheritance): Đối tượng có thể kế thừa các thuộc tính và phương thức từ các lớp cha của nó. Điều này cho phép tái sử dụng mã và tạo ra mối quan hệ phân cấp giữa các đối tượng.
* Tính đa hình (Polymorphism): Đối tượng có thể thể hiện các hành vi khác nhau tùy thuộc vào ngữ cảnh hoặc kiểu dữ liệu của nó. Điều này cho phép gọi các phương thức cùng tên nhưng có thể có hành vi khác nhau, tùy thuộc vào kiểu đối tượng thực sự.

2.Sự khác nhau giữa đối tượng và lớp.Ký hiệu UML của chúng thế nào ?

Đối tượng và lớp là hai khái niệm cơ bản trong lập trình hướng đối tượng, nhưng chúng có ý nghĩa và đặc điểm khác nhau.

**Lớp (Class)**:

* Lớp là một mô hình, một khuôn mẫu để tạo ra các đối tượng. Nó định nghĩa các thuộc tính và phương thức mà các đối tượng của nó sẽ có.
* Lớp là một khái niệm trừu tượng, không tồn tại trong bộ nhớ như các đối tượng. Thay vào đó, nó là một bản thiết kế để tạo ra các đối tượng.
* Lớp có thể coi là một khuôn mẫu để tạo ra các đối tượng có cùng cấu trúc và hành vi.
* Ký hiệu UML của một lớp là một hình chữ nhật chứa tên của lớp, các thuộc tính và phương thức của nó.

**Đối tượng (Object)**:

* Đối tượng là một phiên bản cụ thể của một lớp. Nó được tạo ra từ lớp và có các thuộc tính và phương thức cụ thể.
* Đối tượng là một thể hiện cụ thể của một lớp, có thể tồn tại trong bộ nhớ khi chương trình đang chạy.
* Mỗi đối tượng có trạng thái riêng biệt, tức là giá trị của các thuộc tính của nó có thể thay đổi trong quá trình thực thi.
* Ký hiệu UML của một đối tượng là một hình bầu dục hoặc hình tròn chứa tên của đối tượng.

3. Các đối tượng giao tiếp với nhau bằng cách nào ?

Các đối tượng trong lập trình hướng đối tượng giao tiếp với nhau thông qua việc gửi và nhận các thông điệp. Giao tiếp giữa các đối tượng xảy ra khi một đối tượng gửi một thông điệp đến một đối tượng khác và đối tượng đó phản ứng tương ứng.

Dưới đây là một số cách mà các đối tượng có thể giao tiếp với nhau:

1. Gọi phương thức (Method Invocation): Đối tượng A có thể gọi một phương thức của đối tượng B bằng cách sử dụng cú pháp của ngôn ngữ lập trình. Khi đó, đối tượng B sẽ thực thi phương thức được gọi và trả về kết quả (nếu có) cho đối tượng A.
2. Truyền thông điệp (Message Passing): Đối tượng A có thể gửi một thông điệp đến đối tượng B thông qua một phương thức. Đối tượng B sau đó nhận thông điệp và thực hiện hành động tương ứng. Cơ chế này thường được sử dụng trong các ngôn ngữ lập trình như Java và C++.
3. Sự kiện và xử lý sự kiện (Events and Event Handling): Trong một số trường hợp, các đối tượng có thể tương tác thông qua sự kiện. Một đối tượng có thể phát sinh một sự kiện khi có sự thay đổi trạng thái hoặc người dùng tương tác với nó. Các đối tượng khác sau đó có thể đăng ký để lắng nghe và xử lý sự kiện đó.
4. Truyền tham số (Parameter Passing): Khi gọi một phương thức của một đối tượng, thường có thể truyền các tham số cho phương thức đó. Các tham số này có thể là các đối tượng khác, cho phép các đối tượng tương tác thông qua dữ liệu.

4.Liệt kê 4 loại liên kết giữa các lớp.

1. Liên kết Kết tập (Aggregation):

* Aggregation là một loại liên kết mà một lớp đóng vai trò như một phần của một lớp khác. Nó biểu thị mối quan hệ "has-a" giữa các đối tượng, trong đó một đối tượng tồn tại như một phần của một đối tượng khác.
* Ví dụ: Một lớp "Đội" có thể chứa một tập hợp các đối tượng "Nhân viên". Khi đó, lớp "Đội" là lớp chủ và lớp "Nhân viên" là phần của lớp "Đội".

1. Liên kết Gộp (Composition):

* Composition là một dạng mạnh hơn của aggregation, trong đó một đối tượng tồn tại như một phần của một đối tượng khác và không thể tồn tại độc lập. Nếu đối tượng chủ bị hủy, các đối tượng phần cũng sẽ bị hủy.
* Ví dụ: Một lớp "Nhà" có thể chứa các đối tượng "Phòng", và các phòng không thể tồn tại nếu không có nhà. Khi nhà bị hủy, tất cả các phòng bên trong cũng sẽ bị hủy.

1. Liên kết Kết hợp (Association):

* Association là một loại liên kết mà các đối tượng của các lớp tương tác với nhau. Nó biểu thị mối quan hệ "is-a" hoặc "has-a" giữa các đối tượng.
* Ví dụ: Một lớp "Sinh viên" có thể có mối quan hệ association với một lớp "Môn học", trong đó sinh viên tham gia vào các môn học khác nhau.

1. Liên kết Kế thừa (Inheritance):

* Inheritance là một loại liên kết trong đó một lớp mới được tạo ra bằng cách mở rộng một lớp hiện có, được gọi là lớp cơ sở. Lớp mới kế thừa tất cả các thuộc tính và phương thức của lớp cơ sở và có thể mở rộng hoặc ghi đè chúng.
* Ví dụ: Một lớp "Học sinh" có thể kế thừa từ lớp "Sinh viên", trong đó học sinh là một loại sinh viên nhưng có thêm các thuộc tính hoặc phương thức riêng biệt.

5.Khác nhau giữa liên kết kết hợp và kết tập.

Liên kết Kết hợp (Composition):

* Liên kết kết hợp là một mối quan hệ mạnh mẽ giữa các đối tượng, trong đó một đối tượng tồn tại như là một phần của một đối tượng khác và không thể tồn tại độc lập.
* Trong liên kết kết hợp, đối tượng phần được tạo và quản lý bởi đối tượng chủ. Nếu đối tượng chủ bị hủy, tất cả các đối tượng phần cũng sẽ bị hủy.
* Liên kết kết hợp thường biểu thị một mối quan hệ "là một phần của" giữa các đối tượng.
* Ví dụ: Mối quan hệ giữa một nhà và các phòng bên trong. Nếu nhà bị hủy, tất cả các phòng bên trong cũng sẽ bị hủy.

Liên kết Kết tập (Aggregation):

* Liên kết kết tập là một mối quan hệ yếu hơn, trong đó một đối tượng tồn tại như là một phần của một đối tượng khác, nhưng có thể tồn tại độc lập.
* Trong liên kết kết tập, đối tượng phần tồn tại độc lập và có thể tồn tại trong nhiều ngữ cảnh khác nhau mà không phụ thuộc vào đối tượng chủ.
* Liên kết kết tập thường biểu thị một mối quan hệ "có một" giữa các đối tượng.
* Ví dụ: Mối quan hệ giữa một bộ đồng hồ và các bộ phận bên trong. Mỗi bộ phận có thể tồn tại độc lập và có thể thuộc về nhiều bộ đồng hồ khác nhau.

6.khác nhau giữa kết tập và gộp.

Kết tập (Aggregation):

* Kết tập là một mối quan hệ yếu hơn, trong đó một đối tượng tồn tại như là một phần của một đối tượng khác, nhưng có thể tồn tại độc lập.
* Đối tượng phần tồn tại độc lập và có thể tồn tại trong nhiều ngữ cảnh khác nhau mà không phụ thuộc vào đối tượng chủ.
* Kết tập thường biểu thị một mối quan hệ "có một" giữa các đối tượng.
* Ví dụ: Mối quan hệ giữa một lớp "Đội" và các đối tượng "Nhân viên". Một đội có thể có nhiều nhân viên, và mỗi nhân viên có thể thuộc về một hoặc nhiều đội khác nhau.

Gộp (Composition):

* Gộp là một mối quan hệ mạnh mẽ hơn, trong đó một đối tượng tồn tại như là một phần của một đối tượng khác và không thể tồn tại độc lập.
* Đối tượng phần được tạo và quản lý bởi đối tượng chủ. Nếu đối tượng chủ bị hủy, tất cả các đối tượng phần cũng sẽ bị hủy.
* Gộp thường biểu thị một mối quan hệ "là một phần của" giữa các đối tượng.
* Ví dụ: Mối quan hệ giữa một lớp "Nhà" và các đối tượng "Phòng". Một nhà chứa các phòng bên trong, và các phòng không thể tồn tại nếu không có nhà.

7.Khi nào có thể mô hình 1 lớp là lớp con của 1 lớp khác.

* Mối quan hệ "is-a" rõ ràng: Nếu một đối tượng được mô tả là một loại cụ thể của một loại chung, thì có thể sử dụng kế thừa để mô hình mối quan hệ này. Ví dụ, "Học sinh" có thể được mô hình là một lớp con của lớp "Người" vì học sinh là một loại người.
* Cần tái sử dụng mã nguồn: Khi có các thuộc tính và phương thức chung giữa các lớp, việc sử dụng kế thừa có thể giúp tái sử dụng mã nguồn và tránh lặp lại mã.
* Mở rộng chức năng: Khi lớp con cần mở rộng hoặc thay đổi một số chức năng của lớp cha, việc kế thừa cho phép lớp con thừa hưởng và mở rộng các chức năng này mà không cần phải triển khai lại từ đầu.
* Tổ chức và quản lý mã nguồn: Việc sử dụng kế thừa có thể giúp tổ chức mã nguồn một cách logic và dễ quản lý hơn, bằng cách tạo ra các lớp có mối quan hệ phân cấp rõ ràng và có cùng một cơ sở chung.

8.Sự khác nhau giữa 1 hoạt động (operation) và 1 phương thức (method)

Phương thức (Method):

* Phương thức là một hàm hoặc một phần của mã được định nghĩa trong một lớp. Nó mô tả các hành động cụ thể mà đối tượng của lớp đó có thể thực hiện.
* Phương thức có thể được gọi là các hàm thành viên của lớp, và chúng được gắn liền với các đối tượng của lớp đó.
* Phương thức thường được sử dụng để thực hiện các hoạt động cụ thể như tính toán, xử lý dữ liệu, và tương tác với các đối tượng khác.

Hoạt động (Operation):

* Hoạt động là một thuật ngữ chung hơn, nó bao gồm cả phương thức và các hoạt động khác mà một đối tượng có thể thực hiện.
* Hoạt động có thể bao gồm các phương thức, nhưng cũng có thể bao gồm các hoạt động khác như truy cập thuộc tính, truy vấn trạng thái, hoặc thậm chí là các hành động không được triển khai cụ thể mà chỉ được định nghĩa ở mức độ trừu tượng.
* Trong UML (Unified Modeling Language), "hoạt động" thường được sử dụng để chỉ các hành động mà một đối tượng có thể thực hiện, trong khi "phương thức" thường được sử dụng trong ngôn ngữ lập trình cụ thể để chỉ các hàm thành viên của lớp.

9.Một số tiêu chuẩn thiết kế của 1 lớp được coi là tốt.

1. Problem domain: Trong quá trình phân tích, các lớp nên phản ánh đúng các đối tượng trong phạm vi của bài toán
2. Functionality: Một lớp cần có cả dữ liệu và các hành vi (ít nhất là trong quá trình phân tích). Nếu một lớp chỉ có các hành vi thì nó nên thuộc vào các lớp khác. Một lớp cũng không nên chỉ có các thuộc tỉnh (dù có thể có thêm một số hàm set/get)
3. Cohesion: Mỗi lớp nên có tỉnh cổ kết cao, chỉ nên liên quan đến một việc chính
4. Substituability: Khi có kế thừa, đồi tượng của lớp dẫn xuất (lớp con) cần có thể thay thế được cho một đối tượng của lớp cơ sở (lớp cha)

Chương 9:

* 1. Hãy nêu tầm quan trọng của mô hình hóa cấu trúc hệ thống (structural modeling)
     + **Hiểu biết sâu sắc về hệ thống**
     + **Thiết kế và tối ưu hóa**
     + **Phát hiện và giải quyết vấn đề**
     + **Quản lý dự án và lập kế hoạch**
     + **Tương tác giữa các hệ thống**
     + **Giao tiếp và đào tạo**
  2. Nêu các bước chính để xây dựng biểu đồ lớp
     + Xác định các đối tượng trọng phạm vi bài toán, từ đó xác định các lớp tương ứng
     + Xác định các thuộc tính của từng lớp
     + Xác định các mối quan hệ giữa các lớp
     + Xác định và phân bổ trách nhiệm của từng lớp (từ đó xác định các hoạt động mà mỗi lớp có thể thực hiện )
  3. Phân loại đối tượng
     + Theo Shlaer và Mellor
       - Tangible things: Cars, pressure sensors
       - Roles: Mother, teacher, politician
       - Events: Landing, interrupt, request
       - Interactions: Loan, meeting, intersection
     + Theo Ross
       - People: Người thực hiện một số công việc
       - Places: Khu vực dành cho người hoặc đồ vật
       - Things: Các vật thể hoặc nhóm các đồ vật
       - Organizations: Tập hợp được tổ chức chính thức gồm con người, nguồn lực, cơ sở vật chất
       - Concepts: Các khái niệm (không hữu hình)
  4. Khi phân loại đối tượng, hãy liệt kê 4 loại đối tượng thường gặp
     + Tangible things(vật thể hữu hình) : Đây là các đối tượng mà chúng ta có thể cảm nhận được một cách vật lý, như xe hơi, máy móc, hoặc các thiết bị như cảm biến áp suất.
     + Roles(vai trò) : Các đối tượng này đại diện cho các vai trò mà con người hoặc thực thể khác đảm nhận trong xã hội hoặc trong một hệ thống cụ thể. Ví dụ như một người mẹ, giáo viên, hoặc chính trị gia
     + Events : Những sự kiện xảy ra trong một hệ thống, như sự kiện hạ cánh của một máy bay, một yêu cầu được gửi đi, hoặc một sự gián đoạn trong hệ thống.
     + Interactions: Đây là các đối tượng thể hiện mối quan hệ hoặc tương tác giữa các thực thể hoặc các sự kiện. Ví dụ như một cuộc vay, một cuộc họp, hoặc sự giao nhau của các đường đi
  5. Trình bày kỹ thuật, phân tích danh từ
     + Bước 1: Tìm các danh từ trong phạm vi bài toán
       - Định nghĩa vấn đề (Problem definition)
       - Các yêu cầu hệ thống (System requirements)
       - Mô tả các ca sử dụng (Use case description)
     + Bước 2: Xác định (gạch chân) các danh từ và cụm danh từ (có thể dựa thêm vào các cách phân loại)
     + Bước 3 : Loại bỏ những danh từ không phù hợp
       - Không phải danh từ hoặc cụm danh từ nào cũng có thể là lớp!
       - Một số danh từ không phù hợp cần loại bỏ
         * Chỉ có thể chứa giá trị (sẽ trở thành các thuộc tính)
         * Bị trùng lặp về ý nghĩa (với một danh từ đã lựa chọn)
         * Không rõ ràng hoặc không phản ảnh các đối tượng của bài toán
         * Nằm bên ngoài phạm vi hệ thống
         * Là toàn bộ hệ thống
     + Bước 4 : Sắp xếp những danh từ còn lại thành các lớp
  6. Khi phân tích danh từ , hãy liệt kê các lý do để loại bỏ một danh từ trong quá trình tìm ra các lớp?
     + - Không phải danh từ hoặc cụm danh từ nào cũng có thể là lớp!
       - Một số danh từ không phù hợp cần loại bỏ
         * Chỉ có thể chứa giá trị (sẽ trở thành các thuộc tính)
         * Bị trùng lặp về ý nghĩa (với một danh từ đã lựa chọn)
         * Không rõ ràng hoặc không phản ảnh các đối tượng của bài toán
         * Nằm bên ngoài phạm vi hệ thống
         * Là toàn bộ hệ thống
  7. Hãy liệt kê 4 loại quan hệ ( relationships) giữa các lớp ?
     + Kết hợp ( Association)
     + Kết tập (Aggregation)
     + Gộp ( Composition)
     + Kế thừa (Inheritance)

Chương 10:

* 1. Trình bày mục đích của kỹ thuật CRC?
     + Xác định trách nhiệm của từng lớp
     + Mô tả tương tác giữa các đối tượng
     + Đảm bảo tính toàn vẹn của dữ liệu
  2. Những yếu tố nào của một lớp có thể nắm bắt bởi CRC?
     + Lớp : Tên của lớp cung cấp một danh tính cho lớp và thường mô tả một nhóm đối tượng có cùng đặc điểm và hành vi.
     + Trách nhiệm (responsibility) : Mô tả những gì một lớp biết hoặc có thể thực hiện. Điều này bao gồm:
       - Dữ liệu mà lớp cần lưu trữ (trạng thái)
       - Các chức năng hoặc phương thức mà lớp cung cấp (hành vi
       - Các tác vụ mà lớp đảm nhận hoặc kết quả mà lớp cần phải đạt được
     + Cộng tác viên(Collabarotor) : Mô tả các lớp khác mà lớp này tương tác hoặc phụ thuộc vào để thực hiện trách nhiệm của mình. Điều này bao gồm:
       - Các lớp cung cấp thông tin cần thiết cho lớp này
       - Các lớp mà lớp này cập nhật hoặc tác động đến
       - Các lớp mà lớp này kế thừa hoặc triển kha
  3. Có thể sử dụng CRC trong làm việc theo nhóm ? Cách làm ?
     + Có thể
     + Cách làm :
       - B1: Tạo các thẻ CRC và phân chia cho mỗi thành viên trong nhóm
       - B2: Thảo luận và điền thẻ CRC
       - B3: Mô phỏng các tương tác giữa các lớp và kiểm tra sự đầy đủ và chính xác của mỗi lớp
       - B4: Lặp lại quá trình để tối ưu các thẻ
       - B5: Triển khai, sử dụng các thẻ
  4. Ưu điểm và nhược điểm của kỹ thuật phân tích động từ
     + Ưu điểm:
       - Phát hiện lỗi thực tế: Phân tích động từ có thể phát hiện lỗi xảy ra trong quá trình thực thi thực tế của chương trình, bao gồm các lỗi về bộ nhớ, lỗi logic, và các lỗi thời gian chạy khác mà phân tích tĩnh có thể bỏ qua
       - Hiệu quả cao trong kiểm tra môi trường thực
       - Cho phép kiểm tra trực quan giao diện ,trải nghiệm người dùng và các tương tác thực tế
     + Nhược điểm:
       - Không phát hiện tất cả các lỗi
       - Chi phí cao , tốn thời gian
       - Phụ thuộc vào dữ liệu đầu vào
       - Khó tái sử dụng
  5. Ưu điểm và nhược điểm của kỹ thuật CRC?
     + Ưu điểm:
       - Nhanh chóng xác định được trách nhiệm của từng lớp
       - Linh hoạt, dễ dàng trao đổi, hoàn thiện và phù hợp với làm việc nhóm
       - Có thể mô tả tương tác giữa các đối tượng
     + Nhược điểm:
       - Không mô tả được chi tiết trình tự tương tác giữa các đối tượng
       - Có thể khắc phục bằng cách sử dụng các loại biểu đồ khác
  6. Trình bày về hai loại biểu đồ tương tác : tác dụng, đặc điểm?

a.Biểu đồ trình tự(sequence diagram)

**Tác dụng:**

* Biểu đồ trình tự mô tả cách các đối tượng tương tác với nhau dựa trên thời gian.
* Nó được sử dụng để thể hiện các trao đổi thông điệp giữa các đối tượng trong một quá trình cụ thể hoặc để giải quyết một chức năng kinh doanh nhất định.
* Biểu đồ này rất hữu ích trong việc mô tả các kịch bản sử dụng cụ thể, từ đó giúp phát triển các test case cho quá trình kiểm thử.

**Đặc điểm:**

* Trục dọc của biểu đồ đại diện cho thời gian, trong khi trục ngang thể hiện các đối tượng tham gia.
* Các mũi tên thể hiện sự trao đổi thông điệp giữa các đối tượng, theo thứ tự thời gian từ trên xuống dưới.
* Các lifeline (đường đời) biểu diễn thời gian tồn tại của một đối tượng trong quá trình tương tác.
* Biểu đồ này có thể bao gồm các điều kiện, vòng lặp và cấu trúc rẽ nhánh để mô tả các logic phức tạp.

b.Biểu đồ giao tiếp(communication diagram)

**Tác dụng:**

* Biểu đồ giao tiếp cung cấp một cái nhìn về mối quan hệ và sự tương tác giữa các đối tượng dựa trên các thông điệp được trao đổi.
* Được sử dụng để mô tả tổ chức và sự phụ thuộc lẫn nhau giữa các đối tượng, phù hợp khi cần mô tả các tương tác phức tạp và nhiều đối tượng.
* Biểu đồ này giúp hiểu cách các đối tượng cộng tác để hoàn thành một nhiệm vụ.

**Đặc điểm:**

* Biểu đồ giao tiếp tập trung vào mối quan hệ giữa các đối tượng hơn là trình tự thời gian.
* Các đối tượng được thể hiện bằng hình chữ nhật, và mũi tên giữa chúng chỉ thị sự trao đổi thông điệp.
* Mỗi mũi tên có thể được đánh số để thể hiện thứ tự trao đổi thông điệp, nhưng không nhất thiết theo trình tự thời gian.
* Các mối quan hệ phức tạp và điều kiện có thể được thể hiện thông qua các liên kết và thông điệp.