# ÔN TẬP NỘI DUNG KIỂM THỬ

## Chương 1: Cơ bản về kiểm thử phần mềm

### Tại sao cần kiểm thử?

* Lỗi phần mềm có thể gây hại cho con người, môi trường hoặc doanh nghiệp.
* Kiểm thử giúp giảm rủi ro, nâng cao chất lượng phần mềm.
* Chi phí sửa lỗi tăng lên theo giai đoạn phát triển.

### Kiểm thử là gì?

* Là một quá trình gồm các hoạt động: lập kế hoạch, phân tích, thiết kế, thực thi, đánh giá và đóng kiểm thử.
* Mục tiêu: phát hiện lỗi, đảm bảo chất lượng, ngăn ngừa lỗi.
* Kiểm thử có nhiều cấp độ: kiểm thử phát triển, chấp nhận, vận hành.

### Nguyên tắc kiểm thử phần mềm:

* Kiểm thử chỉ có thể chứng minh sự tồn tại của lỗi, không thể chứng minh phần mềm không có lỗi.
* Kiểm thử toàn diện là không khả thi.
* Kiểm thử sớm giúp phát hiện lỗi sớm, giảm chi phí sửa lỗi.
* Lỗi thường tập trung vào một số ít thành phần của hệ thống (Defect clustering).
* Lặp lại cùng một kiểm thử sẽ không phát hiện lỗi mới (Pesticide Paradox).
* Kiểm thử phụ thuộc vào bối cảnh ứng dụng.
* Phần mềm không có lỗi chưa chắc đã đáp ứng yêu cầu của người dùng.

### Quy trình kiểm thử cơ bản:

* **Lập kế hoạch và kiểm soát:** Xác định phạm vi, mục tiêu, tài nguyên kiểm thử.
* **Phân tích và thiết kế:** Xác định điều kiện kiểm thử, thiết kế test case.
* **Thực thi kiểm thử:** Chạy test case, ghi nhận kết quả, báo cáo lỗi.
* **Đánh giá tiêu chí thoát:** So sánh kết quả thực tế với tiêu chí đặt ra.
* **Hoàn tất kiểm thử:** Lưu trữ tài liệu, đánh giá bài học kinh nghiệm.

### Tâm lý trong kiểm thử:

* Kiểm thử viên cần có tư duy độc lập với lập trình viên.
* Tránh chỉ trích cá nhân, tập trung vào chất lượng sản phẩm.
* Hợp tác giữa các nhóm phát triển và kiểm thử giúp đạt mục tiêu chung.

## Chương 2: Kiểm thử trong suốt vòng đời phần mềm

### Mô hình phát triển phần mềm

* **Mô hình V**: Kiểm thử song song với các giai đoạn phát triển, từ kiểm thử đơn vị đến kiểm thử chấp nhận.
* **Mô hình lặp – gia tăng**: Phát triển và kiểm thử theo từng phần nhỏ, gồm Agile, RUP, RAD, Prototyping.
* **Kiểm thử trong vòng đời phát triển**: Mỗi giai đoạn phát triển có hoạt động kiểm thử tương ứng.

### Các mức kiểm thử

* **Kiểm thử thành phần (Component Testing)**: Kiểm tra các module, chương trình nhỏ, do lập trình viên thực hiện.
* **Kiểm thử tích hợp (Integration Testing)**: Kiểm tra sự kết nối giữa các thành phần. Có các phương pháp: Big-Bang, Top-down, Bottom-up.
* **Kiểm thử hệ thống (System Testing)**: Kiểm thử toàn bộ hệ thống để đảm bảo đáp ứng yêu cầu.
* **Kiểm thử chấp nhận (Acceptance Testing)**: Do người dùng thực hiện để kiểm tra tính phù hợp của hệ thống.

### Các loại kiểm thử

* **Kiểm thử chức năng (Functional Testing)**: Kiểm tra phần mềm dựa trên yêu cầu chức năng.
* **Kiểm thử phi chức năng (Non-functional Testing)**: Kiểm tra các yếu tố như hiệu suất, khả năng bảo trì, tính di động.
* **Kiểm thử cấu trúc (Structural Testing)**: Dựa vào mã nguồn để kiểm thử, gồm kiểm thử hộp trắng.
* **Kiểm thử liên quan đến thay đổi**:
  + **Kiểm thử xác nhận (Confirmation Testing)**: Kiểm tra lại lỗi đã được sửa.
  + **Kiểm thử hồi quy (Regression Testing)**: Kiểm tra xem thay đổi có gây ảnh hưởng tiêu cực không.

### Kiểm thử bảo trì (Maintenance Testing)

* **Lý do kiểm thử bảo trì**: Chỉnh sửa, di chuyển hoặc ngừng hệ thống.
* **Phân tích tác động (Impact Analysis)**: Xác định phạm vi kiểm thử lại khi có thay đổi.
* **Các tác nhân kích hoạt kiểm thử bảo trì**: Sửa lỗi, cải tiến hệ thống, bảo trì định kỳ.

## Chương 3: Kỹ thuật kiểm thử tĩnh

### Kiểm thử tĩnh và quy trình kiểm thử

* **Kiểm thử tĩnh**: Kiểm thử mà không cần chạy phần mềm, gồm xem xét (review) và phân tích mã tĩnh.
* **Lợi ích của kiểm thử tĩnh**:
  + Phát hiện lỗi sớm, giảm chi phí sửa lỗi.
  + Tăng năng suất phát triển và nhận thức về chất lượng.

### Quy trình review

* **Loại review**:
  + **Review không chính thức**: Không theo quy trình tài liệu cụ thể.
  + **Review chính thức**: Có tài liệu và quy trình rõ ràng, gồm kiểm tra kỹ thuật (technical review), thanh tra (inspection) và walkthrough.
* **Giai đoạn review chính thức**:
  + Lập kế hoạch → Họp khởi động → Chuẩn bị cá nhân → Họp review → Sửa lỗi → Theo dõi.

### Phân tích mã tĩnh bằng công cụ

* **Mục tiêu**: Phát hiện lỗi trong mã nguồn mà không cần thực thi.
* **Lợi ích**:
  + Phát hiện lỗi sớm, giảm rủi ro.
  + Xác định các vấn đề khó tìm khi kiểm thử động.
* **Lỗi phổ biến được phát hiện**:
  + Biến không khởi tạo, mã không thể truy cập, vi phạm tiêu chuẩn lập trình, lỗ hổng bảo mật.

## Chương 4: Kỹ thuật thiết kế kiểm thử

### Quy trình phát triển kiểm thử

* **Phân tích kiểm thử**: Xác định điều kiện kiểm thử từ yêu cầu hệ thống.
* **Thiết kế kiểm thử**: Tạo test case với đầu vào, kết quả mong đợi.
* **Thực hiện kiểm thử**: Viết kịch bản kiểm thử, sắp xếp ưu tiên.

### Các loại kỹ thuật thiết kế kiểm thử

* **Hộp đen (Black-box)**: Kiểm thử dựa trên đặc tả yêu cầu, không xét mã nguồn.
* **Hộp trắng (White-box)**: Kiểm thử dựa trên cấu trúc mã nguồn.
* **Dựa vào kinh nghiệm (Experience-based)**: Kiểm thử dựa trên phán đoán và kinh nghiệm của tester.

### Kỹ thuật kiểm thử hộp đen

* **Phân vùng tương đương (Equivalence Partitioning - EP)**: Nhóm các giá trị đầu vào thành vùng hợp lệ và không hợp lệ để kiểm thử.
* **Phân tích giá trị biên (Boundary Value Analysis - BVA)**: Kiểm thử các giá trị tại biên của phân vùng.
* **Bảng quyết định (Decision Table Testing)**: Xác định các tổ hợp điều kiện và hành động tương ứng.
* **Chuyển trạng thái (State Transition Testing)**: Kiểm thử hệ thống dựa trên trạng thái và điều kiện chuyển đổi.
* **Kiểm thử theo kịch bản (Use Case Testing)**: Dựa trên các kịch bản sử dụng thực tế của người dùng.

### Kỹ thuật kiểm thử hộp trắng

* **Bao phủ câu lệnh (Statement Coverage)**: Đảm bảo tất cả các câu lệnh trong mã được thực thi ít nhất một lần.
* **Bao phủ quyết định (Decision Coverage)**: Kiểm thử tất cả các nhánh điều kiện (IF, SWITCH,...) trong mã.

### Kỹ thuật kiểm thử dựa vào kinh nghiệm

* **Phán đoán lỗi (Error Guessing)**: Tester dự đoán các lỗi dựa trên kinh nghiệm.
* **Kiểm thử thăm dò (Exploratory Testing)**: Kiểm thử linh hoạt dựa trên sự khám phá trong quá trình thực thi.

### Lựa chọn kỹ thuật kiểm thử

* Phụ thuộc vào loại hệ thống, yêu cầu khách hàng, tài liệu có sẵn, mức độ rủi ro và thời gian kiểm thử.

# Chương 5: Quản lý kiểm thử

### Tổ chức kiểm thử (Test Organization)

* Kiểm thử độc lập giúp phát hiện lỗi hiệu quả hơn.
* Có nhiều mức độ độc lập: từ lập trình viên tự kiểm thử đến đội kiểm thử độc lập hoặc bên thứ ba.
* Lợi ích: Giảm thiên vị, phát hiện lỗi khách quan hơn.
* Hạn chế: Có thể gây cô lập với đội phát triển, làm chậm tiến độ.

### Lập kế hoạch và ước lượng kiểm thử (Test Planning & Estimation)

* **Lập kế hoạch kiểm thử**: Xác định phạm vi, phương pháp, tài nguyên và lịch trình kiểm thử.
* **Ước lượng kiểm thử**:
  + **Dựa trên số liệu (metrics-based)**: Dùng dữ liệu từ các dự án trước.
  + **Dựa trên chuyên gia (expert-based)**: Dựa vào kinh nghiệm của tester.
* Tiêu chí dừng kiểm thử: Mức độ bao phủ, mật độ lỗi, chi phí, lịch trình.

### Giám sát và kiểm soát tiến độ kiểm thử (Test Progress Monitoring & Control)

* **Giám sát kiểm thử**: Theo dõi trạng thái kiểm thử, so sánh thực tế với kế hoạch.
* **Chỉ số kiểm thử phổ biến**: Tỷ lệ test case hoàn thành, số lỗi phát hiện, độ bao phủ kiểm thử.
* **Kiểm soát kiểm thử**: Điều chỉnh lịch trình, thay đổi mức ưu tiên test case khi cần thiết.

### Quản lý cấu hình (Configuration Management)

* Đảm bảo tất cả tài liệu, mã nguồn, bản build phần mềm được quản lý chính xác.
* Công cụ quản lý cấu hình giúp theo dõi thay đổi, kiểm soát phiên bản.

### Quản lý rủi ro trong kiểm thử (Risk & Testing)

* **Rủi ro dự án**: Thiếu tài nguyên, thay đổi yêu cầu, lỗi thiết kế.
* **Rủi ro sản phẩm**: Chất lượng kém, lỗi nghiêm trọng, không đáp ứng yêu cầu người dùng.
* **Quy trình quản lý rủi ro**: Xác định → Phân tích → Lập kế hoạch → Theo dõi → Kiểm soát.

### Quản lý sự cố (Incident Management)

* Mục tiêu: Ghi nhận, theo dõi và xử lý lỗi trong quá trình kiểm thử.
* **Báo cáo sự cố (Incident Report - IEEE 829)**: Ghi nhận thông tin lỗi, điều kiện tái hiện, ảnh hưởng của lỗi.