**Bài tập tổng hợp về Nguyên tắc và Quy trình Testing**

**Phần 1: Lý thuyết**

**Nguyên tắc cơ bản của Testing**

**a. Liệt kê và tóm tắt 7 nguyên tắc cơ bản của testing**

1. **Testing cho thấy sự hiện diện của lỗi, không phải sự vắng mặt của chúng**
   * Tóm tắt: Testing chỉ có thể chứng minh rằng phần mềm có lỗi, không thể đảm bảo phần mềm không có lỗi.
   * Ví dụ: Trong một ứng dụng đặt vé máy bay, kiểm thử chức năng tìm kiếm chuyến bay có thể phát hiện lỗi khi nhập ngày không hợp lệ, nhưng không đảm bảo mọi trường hợp đều đúng.
2. **Không thể kiểm thử toàn diện**
   * Tóm tắt: Không thể kiểm thử mọi kịch bản do giới hạn về thời gian và tài nguyên.
   * Ví dụ: Trong ứng dụng chat, không thể kiểm thử mọi tổ hợp tin nhắn với mọi ký tự đặc biệt trên mọi thiết bị.
3. **Kiểm thử sớm**
   * Tóm tắt: Testing nên bắt đầu sớm trong vòng đời phát triển phần mềm để giảm chi phí sửa lỗi.
   * Ví dụ: Kiểm tra yêu cầu chức năng của một ứng dụng ngân hàng ngay từ giai đoạn phân tích để phát hiện thiếu sót trong logic chuyển khoản.
4. **Tập trung lỗi**
   * Tóm tắt: Lỗi thường tập trung ở một số mô-đun nhất định trong hệ thống.
   * Ví dụ: Trong một ứng dụng thương mại điện tử, chức năng thanh toán thường có nhiều lỗi hơn chức năng hiển thị sản phẩm.
5. **Nghịch lý thuốc trừ sâu**
   * Tóm tắt: Các kịch bản kiểm thử lặp lại nhiều lần có thể không phát hiện được lỗi mới do phần mềm đã được "miễn nhiễm" với chúng.
   * Ví dụ: Kiểm thử đăng nhập bằng tài khoản hợp lệ lặp đi lặp lại sẽ không phát hiện lỗi khi người dùng nhập sai định dạng email.
6. **Testing phụ thuộc vào ngữ cảnh**
   * Tóm tắt: Quy trình testing phải được điều chỉnh theo loại dự án, mục tiêu, và rủi ro.
   * Ví dụ: Testing một ứng dụng y tế yêu cầu kiểm thử bảo mật nghiêm ngặt hơn so với một ứng dụng giải trí.
7. **Không có lỗi không có nghĩa là phần mềm hữu ích**
   * Tóm tắt: Phần mềm không có lỗi vẫn có thể không đáp ứng được nhu cầu người dùng.
   * Ví dụ: Một ứng dụng ghi chú hoạt động đúng nhưng không có chức năng đồng bộ hóa sẽ không đáp ứng kỳ vọng người dùng.

**b. Áp dụng nguyên tắc vào tình huống thực tế**

**Nguyên tắc chọn: Kiểm thử sớm**

* **Tình huống**: Kiểm thử ứng dụng thương mại điện tử, cụ thể là chức năng thanh toán.
* **Cách áp dụng**:
  + Trong giai đoạn phân tích yêu cầu, đội kiểm thử tham gia đánh giá tài liệu yêu cầu để đảm bảo rằng các trường hợp như thanh toán thất bại, hoàn tiền, hoặc tích hợp với cổng thanh toán được mô tả rõ ràng.
  + Ví dụ: Phát hiện sớm rằng yêu cầu không đề cập đến xử lý khi người dùng hủy thanh toán giữa chừng, từ đó yêu cầu nhóm phát triển bổ sung logic trước khi coding.
  + Kết quả: Giảm chi phí sửa lỗi so với việc phát hiện vấn đề sau khi triển khai.

**Hoạt động Testing và Testware**

**a. Các hoạt động chính trong quy trình testing**

1. **Lập kế hoạch kiểm thử**: Xác định mục tiêu, phạm vi, tài nguyên, và lịch trình kiểm thử.
2. **Thiết kế kiểm thử**: Tạo các test case, test scenario dựa trên yêu cầu phần mềm.
3. **Thực thi kiểm thử**: Chạy các test case, ghi nhận kết quả, và báo cáo lỗi.
4. **Báo cáo và đánh giá**: Tổng hợp kết quả kiểm thử, đánh giá chất lượng sản phẩm, và báo cáo cho các bên liên quan.

**b. Testware là gì?**

* **Định nghĩa**: Testware là tập hợp các tài liệu, công cụ, và dữ liệu được sử dụng trong quá trình kiểm thử phần mềm.
* **3 loại testware và vai trò**:
  1. **Test Plan**: Tài liệu mô tả mục tiêu, phạm vi, và chiến lược kiểm thử. Vai trò: Định hướng cho toàn bộ quá trình kiểm thử, đảm bảo các bên hiểu rõ kế hoạch.
  2. **Test Case**: Các kịch bản kiểm thử chi tiết với đầu vào, đầu ra mong đợi. Vai trò: Hướng dẫn tester thực thi kiểm thử một cách nhất quán và có hệ thống.
  3. **Test Data**: Dữ liệu mẫu dùng để kiểm thử. Vai trò: Đảm bảo kiểm thử mô phỏng các tình huống thực tế (ví dụ: dữ liệu khách hàng giả lập trong ứng dụng ngân hàng).

**c. Ví dụ sử dụng testware**

* **Dự án**: Ứng dụng di động đặt vé xem phim.
* **Testware sử dụng**:
  + **Test Plan**: Mô tả kiểm thử các chức năng như chọn ghế, thanh toán, và gửi vé qua email.
  + **Test Case**: Kiểm tra trường hợp người dùng chọn ghế đã được đặt trước, với kết quả mong đợi là hiển thị thông báo lỗi.
  + **Test Data**: Danh sách giả lập khách hàng và thông tin vé (ví dụ: mã vé, thời gian chiếu).
* **Ứng dụng**: Test case giúp tester kiểm tra chức năng chọn ghế một cách có hệ thống, test data mô phỏng tình huống thực tế, và test plan đảm bảo toàn bộ quá trình được thực hiện đúng tiến độ.

**Vai trò của các thành viên trong đội kiểm thử**

**a. 3 vai trò chính và trách nhiệm**

1. **Test Manager**: Lập kế hoạch kiểm thử, phân bổ tài nguyên, giám sát tiến độ, và báo cáo kết quả cho các bên liên quan.
2. **Test Analyst**: Phân tích yêu cầu, thiết kế test case, và xác định các kịch bản kiểm thử phức tạp.
3. **Tester**: Thực thi test case, ghi nhận lỗi, và báo cáo kết quả kiểm thử.

**b. Phối hợp giữa các vai trò**

* **Ví dụ dự án**: Phát triển ứng dụng quản lý kho hàng.
* **Cách phối hợp**:
  + Test Manager xây dựng kế hoạch kiểm thử, xác định cần kiểm thử các chức năng như nhập kho, xuất kho, và báo cáo tồn kho.
  + Test Analyst thiết kế test case chi tiết, ví dụ: kiểm tra trường hợp nhập kho với số lượng âm.
  + Tester thực hiện kiểm thử, ghi nhận lỗi (như hệ thống chấp nhận số lượng âm), và báo cáo lại cho Test Analyst để phân tích.
  + Test Manager tổng hợp báo cáo và làm việc với đội phát triển để sửa lỗi, đảm bảo chất lượng sản phẩm trước khi phát hành.

**Điều chỉnh quy trình Testing theo ngữ cảnh dự án**

**a. Tại sao cần điều chỉnh và ví dụ**

* **Lý do**: Mỗi dự án có yêu cầu, rủi ro, và tài nguyên khác nhau, đòi hỏi quy trình testing phù hợp để tối ưu hiệu quả và chi phí.
* **Ví dụ**:
  + **Ứng dụng ngân hàng**:
    - Quy trình testing ưu tiên kiểm thử bảo mật (ví dụ: mã hóa dữ liệu, xác thực người dùng) và kiểm thử tích hợp với các hệ thống ngân hàng khác.
    - Sử dụng kiểm thử tự động để kiểm tra các giao dịch lặp lại như chuyển khoản.
  + **Trò chơi di động**:
    - Tập trung vào kiểm thử hiệu suất (ví dụ: tốc độ tải trên các thiết bị khác nhau) và trải nghiệm người dùng (UI/UX).
    - Sử dụng kiểm thử thủ công để đánh giá tính thú vị của trò chơi.

**b. 3 yếu tố cần xem xét**

1. **Quy mô dự án**: Dự án lớn (như hệ thống ERP) yêu cầu kiểm thử toàn diện hơn so với ứng dụng nhỏ (như ứng dụng ghi chú).
2. **Mức độ rủi ro**: Dự án có rủi ro cao (như phần mềm y tế) cần kiểm thử nghiêm ngặt hơn về độ tin cậy và bảo mật.
3. **Thời gian**: Dự án có thời gian ngắn cần ưu tiên các test case quan trọng và sử dụng công cụ tự động hóa để tăng tốc độ.

**Phần 2: Bài tập thực hành**

**Áp dụng nguyên tắc Testing**

* **Ứng dụng**: Ứng dụng đặt đồ ăn trực tuyến, chức năng "thêm món ăn vào giỏ hàng".
* **3 nguyên tắc và cách áp dụng**:
  1. **Testing cho thấy sự hiện diện của lỗi**:
     + Kiểm tra trường hợp người dùng nhập số lượng món ăn âm hoặc vượt quá giới hạn (ví dụ: 1000 món).
     + Kết quả mong đợi: Hệ thống hiển thị thông báo lỗi và không cho phép thêm vào giỏ hàng.
  2. **Kiểm thử sớm**:
     + Tham gia vào giai đoạn phân tích yêu cầu để đảm bảo rằng chức năng thêm món có xử lý các trường hợp như món hết hàng hoặc món bị xóa khỏi thực đơn.
     + Ví dụ: Phát hiện sớm yêu cầu thiếu logic kiểm tra tồn kho trước khi thêm vào giỏ.
  3. **Testing phụ thuộc vào ngữ cảnh**:
     + Tập trung kiểm thử trên các thiết bị di động phổ biến (Android, iOS) vì ứng dụng chủ yếu được dùng trên di động.
     + Kiểm tra trải nghiệm người dùng khi thêm món, như tốc độ phản hồi và hiển thị thông báo thành công.

**Thiết kế quy trình Testing**

* **Ứng dụng**: Ứng dụng đặt đồ ăn trực tuyến.
* **Quy trình testing cơ bản**:
  1. **Lập kế hoạch kiểm thử**:
     + **Bước thực hiện**: Xác định mục tiêu (kiểm thử chức năng thêm món, thanh toán, giao hàng), phạm vi (các tính năng chính), và tài nguyên (2 tester, 1 tuần).
     + **Testware**: Test Plan (mô tả mục tiêu, lịch trình, và chiến lược kiểm thử).
  2. **Thiết kế kiểm thử**:
     + **Bước thực hiện**: Tạo test case cho chức năng thêm món, như kiểm tra thêm món với số lượng hợp lệ, không hợp lệ, hoặc món hết hàng.
     + **Testware**: Test Case (danh sách các trường hợp kiểm thử chi tiết).
  3. **Thực thi kiểm thử**:
     + **Bước thực hiện**: Tester chạy các test case, ghi nhận lỗi (ví dụ: hệ thống cho phép thêm món đã hết hàng).
     + **Testware**: Test Data (danh sách món ăn, số lượng, trạng thái tồn kho).
  4. **Báo cáo và đánh giá**:
     + **Bước thực hiện**: Tổng hợp kết quả kiểm thử, báo cáo lỗi cho đội phát triển, và đánh giá chất lượng tính năng.
     + **Testware**: Test Report (tóm tắt số lượng test case pass/fail, lỗi nghiêm trọng).

**Phân tích ngữ cảnh dự án**

* **Dự án 1: Hệ thống quản lý bệnh viện**
  + **Yếu tố ngữ cảnh**:
    1. **Yêu cầu bảo mật**: Dữ liệu bệnh nhân phải được mã hóa và tuân thủ quy định như HIPAA.
    2. **Độ tin cậy cao**: Hệ thống cần hoạt động ổn định 24/7 để hỗ trợ chăm sóc y tế.
  + **Điều chỉnh quy trình testing**:
    1. Ưu tiên kiểm thử bảo mật (ví dụ: kiểm tra xâm nhập, xác thực người dùng).
    2. Sử dụng kiểm thử tích hợp để đảm bảo các module (lịch hẹn, hồ sơ bệnh nhân) hoạt động đồng bộ.
    3. Thực hiện kiểm thử tải để đảm bảo hệ thống xử lý được số lượng lớn người dùng đồng thời.
* **Dự án 2: Ứng dụng chỉnh sửa ảnh trên di động**
  + **Yếu tố ngữ cảnh**:
    1. **Trải nghiệm người dùng**: Giao diện phải mượt mà, dễ sử dụng trên các thiết bị di động.
    2. **Hiệu suất**: Ứng dụng cần xử lý nhanh các tác vụ như áp dụng bộ lọc ảnh.
  + **Điều chỉnh quy trình testing**:
    1. Tập trung kiểm thử giao diện người dùng (UI/UX) trên nhiều thiết bị và độ phân giải khác nhau.
    2. Sử dụng kiểm thử hiệu suất để đo thời gian áp dụng bộ lọc ảnh.
    3. Kiểm thử thủ công để đánh giá tính thân thiện của giao diện với người dùng.