**PHẦN 1: Các khái niệm căn bản về kiểm thử phần mềm**

**Bài 1: Kiểm thử phầm mềm là gì?**

**Các cách nhìn khác nhau về chất lượng (quality) của một sản phẩm phần mềm**

* Trước khi nói về kiểm thử phần mềm, chúng ta phải nói về chất lượng sản phẩm phần mềm:
  + Đứng ở các góc nhìn khác nhau, định nghĩa về một sản phẩm phần mềm chất lượng cũng sẽ khác nhau
    - Từ phía người dùng: sản phẩm phù hợp với mục đích sử dụng của người đó
    - Từ phía nhà phân bổ, cung cấp sản phẩm (product view): một sản phẩm đạt chất lượng là sản phẩm có được các đặc tính về sản phẩm, đạt được những tiêu chú đánh giá do nhà cung cấp sản phẩm đề ra
      * Tnhs chính xác
      * Tính bảo mật
      * Tính dễ sử dụng
      * …
    - Từ phía nhà sản suất ra sản phẩm phân mềm: đạt được chất lượng cần thiết để đưa ra dùng hay sử dụng phải được xây dựng đúng như những yêu cầu đặc tả (do người sử dụng hoặc nhà cung cấp sản phẩm đề ra)
  + Từ những góc nhìn trên, tam giác chất lượng của sản phẩm phần mềm sẽ được xây dựng như sau:
    - Đỉnh 1: view của nhà cung cấp sản phẩm (Product’s view)
    - Đỉnh 2: view của người sử dụng (user’s view)
    - Đỉnh 3: view của nhà sản xuất (manufacture’s view)
* Một sản phẩm phần mềm sẽ được xây dựng dựa trên yêu ầu người dùng + yêu cầu đặc tả
  + Thông thường, trong quy trình sản xuất phần mềm, nhà cung cấp sản phẩm phần mềm sẽ theo những yêu cầu của user để viết ra một customer requirement (hay có lúc được gọi là high level requirement)
  + Sau đó nhà sản xuất sẽ phát triển thành một tài liệu là requirement specification (yêu cầu đặc tả) mô tả chi tiết yêu cầu về các chức năng, tính năng của sản phẩm 🡪 Developers sẽ dựa vào đó để xẩy dựng / phát triển sản phẩm phần mềm
  + Khoảng cách giữa các đỉnh của tam giác giữa sản phẩm thực tế và yêu cầu chức năng, tính năng của sản phẩm càng lớn, nghĩa là **chất lượng (quality)** của sản phẩm càng kém và ngược lại.

**Kiểm thử phần mềm là gì?**

* Định nghĩa:
  + Theo IEEE 829: Testing là một quá trình thực hiện đánh giá một hệ thống, một sản phẩm tự động bằng tay để xác nhận rằng sản phẩm có thỏa mãn đặc tả hay không
  + Theo ISTQB: Testing là một quá trìn bao gồm một vòng lặp các hoạt động cả tĩnh và động, từ khi lập kế hoạch, chuẩn bị và đánh giá sản phẩm để xác định sản phẩm có thõa mãn yêu cầu đặc tả không, chứng minh sản phẩm đó có phù hợp với mục đích sử dụng không và để phát hiện lỗi.
* Mục tiêu cyả kiểm thử phần mềm:
  + Xác định sản phẩm có thõa mãn yêu cầu đặc tả hay không
  + Xác định sản phẩm có thõa mãn những cái người dùng cần không
  + Có được tự tin để cung cấp một sản phẩm có chất lượng không
  + Xác định các lỗi, nguyên nhân để đưa ra các hành động để hạn chế, ngăn chặn các lỗi trong tương lai
  + Có thể đưa ra các cải tiến quy trình, tìm thêm các công cụ để phục vụ công việc testing hiệu quả hơn

**Sự khác nhau giữa lý thuyết và thực tế:**

1. Testing là một công việc đơn giản, dễ làm: thực tế, testing là 1 công việc thách thức, có những trường hợp kiểm thử đòi hỏi kỹ năng phân tích cao
2. Ai cũng có thể làm manual testing: thực tế testing yêu cầu nhiều kỹ năng cần có và phải có để làm nghề tester
3. Testing để đảm bảo phần mềm sau khi kiểm thử không có lỗi, nhưng thức tế không phải lúc nào việc testing cũng có thể phát hiện được tất cả các lỗi của phần mềm
4. Test tự động sẽ tốt hơn, hiệu quả hơn việc test bằng tay: không thể tự động test hoàn toàn tất cả các trường hợp của phần mềm
5. Testing và debugging là giống nhau: thực tế là hoàn toàn khác nhau:
   1. Testing là tìm lỗi, debugging là hành động để sửa lỗi
   2. Testing thực hiện bởi tester, còn debugging thực hiện bởi developers
   3. Testing là để tìm được càng nhiều lỗi càng tốt, còn debugging là để loại bỏ những lỗi đó

1 B

2 D

3 D

4 D

5 A

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 2: Tại sao phải thực hiện kiểm thử phần mềm?**

**Tại sao việc kiểm thử phần mềm lại cần thiết**

1. Sản phẩm phần mềm nào cũng chắc chắn có lỗi, do đó sẽ luôn cần một người, nhóm người hoặc tổ chức độc lập đứng ra kiểm thử xem sản phẩm có lỗi hay không
2. Kiểm thử giúp tìm hiểu độ tin cậy của phần mềm
3. Các lỗi trong quá trình phần mềm được đưa vào sử dụng có thể rất tốn chi phí để khắc phụ và có thể gây ra những hậu quả nghiêm trọng
4. Tránh được sự kiện tụng của khách hàng
5. Doanh nghiệp muốn phát triển trong một lĩnh vực nào đó luôn cần đảm bảo sản phẩm do mình cung cấp đến người dùng có chất lượng tốt 🡪 cần kiểm thử là đương nhiên

**Lỗi trong sản phẩm phần mềm**

* Lỗi trong phần mềm có thể dẫn đến:
  + Mất tiền để xử lý, khắc phục
  + Thời gian để xử lý, khắc phục
  + Mất sự tin tưởng từ phía khách hàng
  + Thương tổn về tài chính, con người.
* Các khái niệm về lỗi trong sản phẩm, dự án phần mềm:
  + Error: là chỉ hành động của con người gây ra một kết quả không đúng
  + Fault: là một bước sai, quy trình sai hay định nghĩa dữ liệu sai, fault là kết quả của error
  + Failure: là lỗi tìm được trong quá trình sản phẩm phần mềm đã được đưa vào sử dụng, đó chính là độ lệch (GAP) giữa sản phẩm và mong muốn của người sử dụng
* Lỗi xuất hiện khi nào:
  + Bất cứ khi nào tạo ra một sản phẩm thì đều có khả năng sản phẩm đó chứa lỗi
* Chi phí khắc phục lỗi thay đổi như thế nào?
  + Chi phí khắc phục, xử lý lỗi sẽ tăng dần theo thời gian
  + Những lỗi mà ở giai đoạn live use mới được phát hiện, nếu chiếu theo thuật ngữ về lỗi thì sẽ là failure, là những lỗi tốn nhiều chi phí nhất để sửa chữa và khắc phục

1 C

2 C

3 A

4 B

5 C

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 3: các chân lý trong kiểm thử phần mềm**

**Chân lý thứ nhất: testing shows presence of defects (kiểm thử sẽ chỉ ra các lỗi hiện diện hay việc kiểm thử sẽ tìm ra lỗi)**

* Testing sẽ chỉ ra có lỗi tồn tại trong hệ thống, chứ không chứng minh rằng sản phẩm phần mềm đó không có lỗi, hay chứng minh rằng sản phẩm phần mềm đó hoạt động đúng
* Testing làm giảm xác suất lỗi chưa tìm thấy, nhưng ngay cả khi không tìm thấy lỗi, nó cũng không chứng minh được rằng sản phẩm phần mềm đó hoạt động chính xác và không có lỗi

**Chân lý thứ hai: kiểm thử tất cả các trường hợp là không khả thi**

* Kiểm thử mọi thứ là không thể vì thời gian và chi phí sẽ tăng theo cấp số nhân
* Thay vì kiểm thử toàn bộ, việc phân tích rủi ro và dựa trên mức độ ưu tiên chúng ta có thể tập trung kiểm thử vào một số điểm cần thiết

**Chân lý thứ 3: kiểm thử sớm**

* Các hoạt động kiểm thử nên bắt đầu càng sớm càng tốt trong quy trình phát triển và nên tập trung vào các mục tiêu đã định ra từ trước
* Thực hiện kiểm thử sớm giúp phát hiện các lỗi sớm, chi phí để sửa chữa, khắc phục các lỗi đó sẽ rẻ hơn

**Chân lý thứ 4: Defect clustering (sự tập trung lỗi)**

* Trong quá trình thực hiện kiểm thử phần mềm, sẽ có lúc chúng ta thẫy lỗi tập trung ở một số ít các module trong sản phẩm, thông thường là những chức năng chính đòi hỏi xử lý phức tạp, hoặc chức năng do người có ít kinh nghiệm thực hiện

**Chân lý thứ 5: Pesticide Paradox (nghịch lý thuốc trừ sâu)**

* Nếu việc kiểm thử lặp đi lặp lại nhiều lần, thì cuối cùng sẽ có 1 số trường hợp kiểm thử không còn tìm thấy lỗi mới nào 🡺 nghịch lý thuốc trừ sâu. Để khắc phục điều này, cần xem xét sửa đổi các tình huống kiểm thử, các test cases cũng cần được “làm mới” để tìm ra các lỗi tiềm ẩn hơn nữa

**Chân lý thứ 6: Testing is context dependent (kiểm thử theo ngữ cảnh)**

**-** Với mỗi sản phẩm khác nhau, trong những lãnh vực khác nhau thì việc kiểm thử cũng sẽ khác nhau

**Chân lý thứ 7: sự sai lầm về việc không có lỗi**

* Việc tìm và sửa chữa lỗi sẽ là vô nghĩa nếu hệ thống xây dựng xong không thể dùng được và không đáp ứng được nhu cầu và mong đợi của người dùng
* Nghĩa là phần mềm, sau khi đã làm đầy đủ công đoạn để phát triển ra một sản phẩm phần mềm hoàn chỉnh nhưng sản phẩm đưa ra đến tay người dùng không đáp ứng được nhu cầu của người sử dụng thì phần mềm đó coi như thất bại, mặc dù các hoạt động đảm bảo chất lượng phần mềm, kiểm thử vẫn diễn ra đầy đủ

**1 D**

**2 D**

**3 C**

**4 C**

**5 B**

**--------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- ---------------**

**Phần 2: quy trình kiểm thử phần mềm căn bản**

**Bài 4: lập kế hoạch và kiểm soát hoạt động kiểm thử**

**Quy trình kiểm thử căn bản**

* Một quy trình kiểm thử căn bản được định nghĩa gồm 5 hoạt động chính:
  + Lên kế hoạch và kiểm soát hoạt động kiểm thử sản phẩm, hệ thống phần mềm – Test planning & control
  + Phân tích và thiết kế các tình huống kiểm thử - Test analyst & design
  + Viết các tình huống kiểm thử, chuẩn bị dữ liệu kiểm thử (implementation) và chạy chương trình để kiểm thử - test implementation & execution
  + Đánh giá tiêu chí hoàn thành và báo cáo kết quả kiểm thử - Evaluating exit & reporting
  + Tổng kết quá trình kiểm thử - Test closure activities

**Lên kế hoạch kiểm thử sản phẩm**

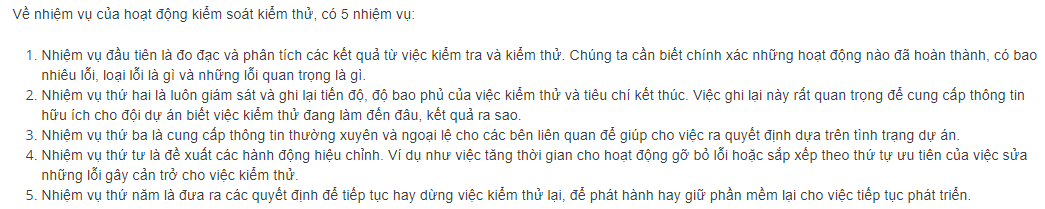
* Bản kế hoạch kiểm thử là một tài liệu mô tả về phạm vi, cách tiếp cận, nguồn lực và kế hoạch thực hiện các hoạt động kiểm thử cho một dự án, sản phẩm phần mềm .
  + **Kế hoạch kiểm thử** (test plan) là một sản phẩm của **quá trình lập kế hoạch kiểm thử** (testing)
  + Việc lập kế hoạch kiểm thử là một hoạt động xây dựng và cập nhật kế hoạch kiểm thử
* Mục tiêu của hoạt động lập kế hoạch kiểm thử:
  + Thiết lập mục tiêu dài và ngắn hạn của việc kiểm thử, cần đạt được cần hiểu các đối tượng chính tham gia vào quá trình sản xuất phần mềm:
    - Khách hàng
    - Các stakeholder
    - Mục tiêu dự án
  + Nhận biết được các rủi ro
  + Xác định được cách tiếp cận và kế hoạch cho việc kiểm thử
* Nhiệm vụ chính trong hoạt động lập kế hoạch:
  + Xác định được phạm vi (scope) kiểm thử
  + Xác định được cách tiếp cận với việc kiểm thử trong dự án
    - Các kỹ thuật kiểm thử
    - Đối tượng kiểm thử
    - Độ bao phủ
    - Người tham gia kiểm thử
    - Các tài liệu dự án, đặc tả phần mềm
  + Thực thi theo chính sách và chiến lược kiểm thử
  + Xác định được nguồn lực cần thiết
    - Con người
    - Môi trường
  + Lên được kế hoạch cho các hoạt động phân tích, thiết kế. thực thi việc kiểm thử
  + Xác định tiêu chí **kết thúc** việc kiểm thử
* 2 điểm quan trọng nhất
  + Tìm được cách tiếp cận tốt nhất cho hoạt động kiểm thử
  + Biết được khi nào việc kiểm thử kết thúc

**Kiểm soát các hoạt động kiểm thử**

Trong mỗi hoạt động, việc quản lý không chỉ kết thúc khi chúng ta lên kế hoạch mà còn kiểm soát nó trong quá trình vận hành xem có đúng với kế hoạch không

Kiểm soát kiểm thử cũng là một hoạt động quản lý và nó diễn ra liên tục

* 2 mục tiêu trong hoạt động kiểm soát
  + Giám sát được tình trạng hiện tại của việc kiểm thử
  + Đưa ra hành động kịp thời kiểm soát được và đạt được mục tiêu chung của dự án



**1 D**

**2 B**

**3 D**

**4 C**

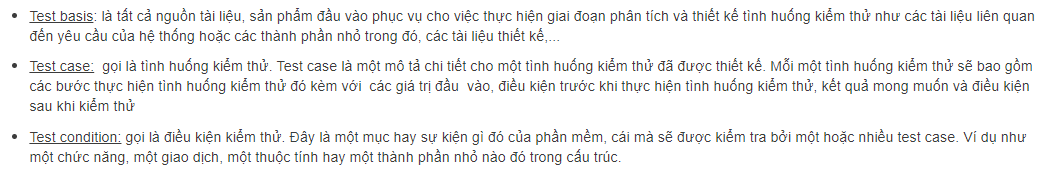
**5 A**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 5: Phân tích yêu cầu và thiết kế các tình huống kiểm thử**

**Phân tích yêu cầu thiết kế các tình huống kiểm thử**

* Là hoạt động thứ 2 trong quy trình kiểm thử và là hoạt động quan trọng để thực hiện kiểm thử cho hệ thống, sản phẩm phần mềm
* Hoạt dộng này yêu cầu phân tích cụ thể yêu cầu đặc tả của các chức năng trong sản phẩm, từ đó thiết kế các tình huống kiểm thử cho chức năng đó
  + Giai đoạn này rất quan trọng, đảm bảo các mục tiêu đã đề ra sẽ được đưa vào các tình huống kiểm thử trong các quá trình thực thi sau này
  + Mục tiêu: xây dựng được các bộ khung các tình huống cần kiểm thử (test design) dựa trên yêu cầu của khách hàng (test basis), mục tiêu (test obj) và các kỹ thuật để thiết kế tình huống kiểm thử (test techniques)
* Các hoạt động chính trong giai đoạn phân tích và thiết kế tình huống kiểm thử:
  + Ktra lại các tài liệu dự án, bao gồm SRS, design, thiết kế kiến trúc … để hiểu rõ hơn về phần mềm, các sơ hở, không rõ ràng trong bản yêu cầu đặc tả
  + Phân tích, đánh giá khả năng kiểm thử của phần mềm dựa trên yêu cầu của khách hàng
  + Xác định thứ tự ưu tiên cho các điều kiện kiểm thử dựa trên kết quả phân tích các chức năng cần kiểm thử
  + Thiết kế và đặt ưu tiên cho các tình huống kiểm thử mức cao
  + Xác định được các dữ liệu kiểm thử (test data) cho các test conditions và các test cases
  + Thiết kế cho việc thiết lập môi trường kiểm thử, yêu cầu về cơ sở hạ tầng, các công cụ kèm theo
  + Tạo được mối liên hệ giữa test basis và test cases để kiểm soát được việc kiểm thử và sự thay đổi của requirements sau này (nếu có)

****

**1 B**

**2 A**

**3 C**

**4 A**

**5 B**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 6: Chuẩn bị và thực hiện kiểm thử**

**Chuẩn bị cho việc kiểm thử**

* Là một quá trình:
  + Phát triển và đặt thứ tự ưu tiên cho các thủ tục kiểm thử, bao gồm các tình huống kiểm thử (test cases) hoặc các bộ tình huống kiểm thử (test suite)
  + Tạo dữ liệu kiểm thử (test data)
  + Chuẩn bị các dụng cụ kiểm thử nếu có (test harness)
  + Viết kịch bản kiểm thử tự động (test script) nếu thực hiện kiểm thử tự động
  + Chuẩn bị MTKT (test environment)
* Mục tiêu:
  + Xây dựng được các thủ tục kiểm thử (test procedure)
  + Cài đặt môi trường, tạo được dữ liệu kiểm thử
* Các nhiệm vụ chính trong thi hành / chuẩn bị kiểm thử:
  + Phát triển và xét thứ tự ưu tiên cho các tình huống kiểm thử, dữ liệu và các công cụ kiểm thử cũng như viết test script nếu có kiểm thử tự động
  + Xây dựng được các bộ kiểm thử từ các tình huống kiểm thử
  + Cài đặt, kiểm tra lại xem môi trường kiểm thử có đúng chưa, có thể chạy kiểm thử cho các trường hợp đặc biệt không

**Thực hiện kiểm thử (test execution)**

* Là quy trình chạy thử một thành phần chức năng hay cả hệ thống dựa trên các tài liệu kiểm thử đã chuẩn bị ở hành động chuẩn bị cho kiêm thử (test implementation) để có kết quả thực tế
* **Mục tiêu:** thực hiện kiểm thử và báo cáo sự bất lường (incident / bug) nếu kết quả thực tế khác mong đợi
* 5 nhiệm vụ chính khi thực hiện kiểm thử:
  + Thực hiện kiểm thử theo một suite hoặc các test cases riêng lẻ với thứ tự đề ra trong thủ tục kiểm thử
  + Ghi lại kết quả của việc kiểm thử, các trạng thái và phiên bản của phần mềm đang kiểm thử, các công cụ kiểm thử (test tool) và test ware
  + Báo cáo sự cố nếu có và phân tích để xác định nguyên nhân
  + Lặp lại việc testing (re-testing) cho mỗi sự cố tìm được, nhằm chắc chắn rằng chúng đã được loại bỏ, và cần đảm bảo lỗi xảy ra sẽ không ảnh hưởng đến các vùng khác của phần mềm (regression testing).

**1 C - SAI / B – SAI / A**

**2 D – SAI / A**

**3 A**

**4 B – SAI / C – SAI / D – SAI / A**

**5 B – SAI / A**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 7: Đánh giá và đóng các hoạt động kiểm thử (Evaluating exit criteria)**

**Đánh giá các hoạt động kiểm thử**

* Là hoạt động t4 trong quy trình kiểm thử: đánh giá kết quả của việc kiểm thử so với mục tiêu đề ra
  + Là hoạt động để biết được việc kiểm thử như thế nào là đủ cho môi giai đoạn kiểm thử
  + Xác định được hoạt động kiểm thử trong giai đoạn có thể kết thúc để chuyển sang giai đoạn tiếp theo hay chưa
* Nhiệm vụ chính:
  + Kiểm tra các kết quả kiểm thử so với các tiêu chí dừng, tiêu chí thoát khỏi giai đaonj kiểm thử đã được xác định và mô tả trong quá trình lập kế hoạch
  + Đánh giá tình trạng hiện tại để nếu cần sẽ phải xác định lại các tiêu chí để dừng
  + Viết báo cáo tổng kết các hoạt động kiểm thử đã thực hiện trong giai đoạn kiểm thử để gửi các bên liên quan

**Đóng các hoạt động kiểm thử (test closure)**

* Là hoạt động thu thập dữ liệu từ các hoạt động kiểm thử, tổng hợp từ kinh nghiệm dựa trên việc ktra và hoàn thiện bộ sản phẩm kiểm thử. Dữ liệu sau khi thu thập sẽ là căn cứ, cơ sở để phân tích số liệu đưa ra các bài học để áp dụng cho tương lai. Đây là hoạt động cuối trong quy trình kiểm thử
* Mục tiêu của Test Closure này là thu thập dữ liệu để:
  + Cung cấp dữ liệu cho bàn giao sản phẩm phần mềm
  + Lưu trử dữ liệu
  + Phân tích cho hoạt động cải tiến sau này
* Các nhiệm vụ chính (7 nhiệm vụ):
  + Kiểm tra sản phẩm thực tế được bàn giao so với kế hoạch như thế nào, và chắc chắn rằng tất cả lỗi đã được giải quyết hay có kế hoạch giải quyết trong các lần bàn giao sau.
  + Đóng các báo cáo về sự cố lại, hoặc ghi chép các thay đổi cho bất cứ vấn đề nào còn đang mở
  + Viết 1 biên bản chấp nhận phần mềm
  + Lưu trữ các sản phẩm kiểm thử, môi trường kiểm thử và cơ sở hạ tầng để sử dụng lần sau, đặc biệt nếu phải thực hiện kiểm thử bảo trì sản phẩm.
  + Bàn giao các sản phẩm kiểm thử cho bộ phận quản lý dữ liệu, bộ phận bảo trì để tiếp tục hỗ trợ trong việc sửa lỗi nếu có hoặc tiếp tục phát triển theo những yêu cầu mới.
  + Phân tích các bài học để xác định những điểm cần thay đổi cho dự án sau
  + Sử dụng các thông tin thu thập được để cải tiến công việc kiểm thử một cách định kỳ như cải tiến việc thiết kế các test cases, các thực hiện kiểm thử …

1 C

2 D / SAI - C

3 D / SAI - B

4 D

5 C / SAI – D

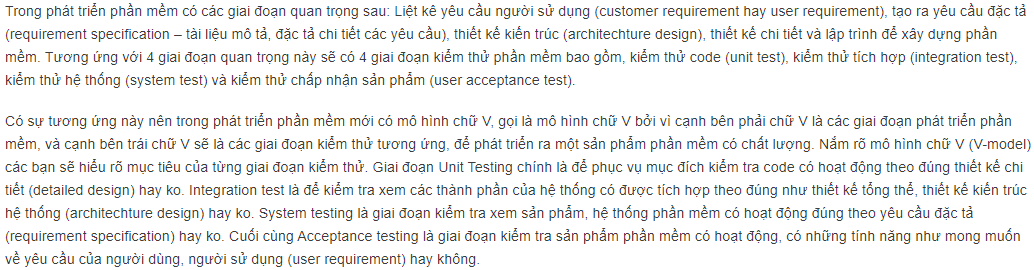
**--------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- ---------------**

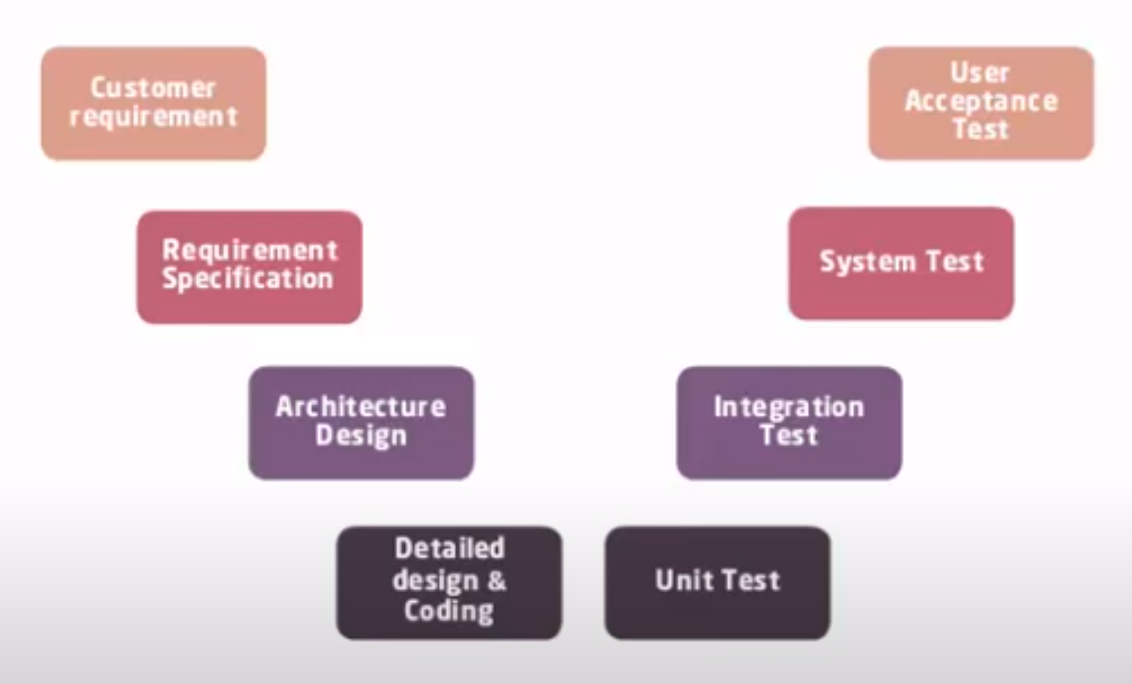
**Phần 3: các mức độ hay các giai đoạn kiểm thử**

**Bài 8: kiểm thử đơn vị**

**Định nghĩa của test levels**

* Là một tập hợp các hoạt động kiểm thử mà được tổ chức, quản lý cùng nhau, phục vụ một mục tiêu kiểm thử nào đó
* Mỗi giai đoạn kiểm thử thường phục vụ cụ thể cho một giai đoạn phát triển phần mềm.
* Các giai đoạn kiểm thử:
  + Unit testing: kiểm thử đơn vị
  + Integration testing: kiểm thử tích hợp
  + System testing: kiểm thử hệ thống
  + Acceptance Testing: kiểm thử chấp nhận (kiểm thử để chấp nhận sản phẩm)

****

****

**Kiểm thử đơn vị (unit test)**

* Là việc kiểm thử từng đơn vị phần mềm xem có hoạt động đúng như thiết kế hay không. Đơn vị cần kiểm thử có thể là:
  + Một dòng lệnh
  + Một hàm
  + Một module
  + Một chương trình
* Mức độ đơn vị to – nhỏ thế nào do người thực hiện làm kiểm thử quyết định, mức độ càng nhỏ thì cơ hội tìm kiếm được lỗi càng lớn
* Các loại kiểm thử đơn vị:
  + Kiểm thử câu lệnh – statement testing: tất cả câu lệnh đều được kiểm thử bằng cách chạy qua ít nhất 1 lần, do đó cần chọn data hợp lý để có thể thực hiện các dòng lệnh viết ra
  + Kiểm thử quyết định hay nhánh (decision or branch testing): tất cả các nhánh đều được chạy qua ít nhất 1 lần
  + Kiểm thử điều kiện – condition testing: kiểm tra sự kết hợp của các điều kiện
  + Kiểm thử đường đi – path testing: kiểm tra tất cả các đường đi từ bắt đầu đến kết thuchs trong 1 hàm ít nhất 1 lần
* Người kiểm thử: lập trình viên, người tạo ra những dòng code
* Sử dụng một hoặc nhiều loại kiểm thử đơn vị để thực hiện việc kiểm thử, cách thức thực hiện có thể như sau:
  + Từ trên xuống, sử dụng các **stubs:** là các đơn vị chưa có mà được gọi bởi 1 đơn vị đã hoàn thành,
  + Từ dưới lên, sử dụng các **drivers:** là các đơn vị chưa có gọi đến các đơn vị đã hoàn thành

**1 A**

**2 B**

**3 A**

**4 A**

**5 C**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 9: kiểm thử tích hợp**

**Kiểm thử tích hợp**

* KTTH là kiểm thử các hoạt động tích hợp các thành phần của phần mềm với nhau để tạo thành 1 hệ thống hoàn chỉnh
* Kiểm thử các hoạt động tích hợp là việc kiểm thử việc giao tiếp hay kết nối về giao diện giữa các thành phần phần mềm với nhau chứ không tập trung vào kiểm thử chức năng, nghiệp vụ của mỗi chức năng của sản phẩm đó.
* Các lỗi tìm được qua việc Regression test là các lỗi thể hiện việc tương tác, giao tiếp giữa 2 thành phần thích hợp
* Tại sao phải kiểm thử tích hợp?
  + UT chỉ kiểm thử cho từng đơn vị riêng lẻ
  + Rất nhiều lỗi có thể phát sinh sau khi chúng ta tích hợp các thành phần của một sản phẩm, hệ thống phần mềm hay tích hợp các hệ thống phần mềm với nhau.
  + Đảm bảo được việc tích hợp các chức năng, modules của hệ thống phần mềm tuân theo thiết kế kiến trúc của sản phẩm, hệ thống đã được kiểm tra và phê duyệt từ trước.
  + Đảm bảo việc giao tiếp, liên kết giữa các chức năng, modules hoạt động chính xác.
* Việc thực hiện kiểm thử tích hợp khi nào tùy thuộc vào mức độ thích hợp:
  + Tích hợp các thành phần thì kiểm thử tích hợp được thực hiện khi các thành phần cần tích hợp đã được thực hiện UT rồi
  + Tích hợp các hệ thống khi mà các hệ thống đã được thực hiện kiểm thử hệ thống rồi

**Các mức độ thích hợp**

1. Tích hợp các thành phần:
   1. Khi phát triển 1 sản phẩm được tạo nên bởi các thành phần (components integration)
2. Tích hợp các hệ thống:
   1. Tích hợp các hệ thống với nhau sử dụng chung 1 nguồn tài nguyên nào đó (system integration)

* Tương ứng với mỗi các mức độ tích hợp thì cách thức thực hiện kiểm thử cũng khác nhau
  + Với tích hợp các thành phần: tập trung vào kiểm thử các giao tiếp của các thành phần được tích hợp và được thực hiện sai khi mỗi thành phần tích hợp đã được kiểm thử đơn vị
  + Với tích hợp hệ thống: tập trung vào kiểm thử việc trao đổi, giao tiếp của các hệ thống được tích hợp và được thực hiện sau khi mỗi hệ thống tích hợp đã được kiểm thử hệ thống

**Các loại tích hợp**

* Tích hợp đồng thời (Big-bang integration): tích hợp tất cả thành phần, hệ thống cùng 1 lúc, tại cùng 1 thời điểm
* Tích hợp dần dần (incremental integration): tích hợp các thành phần, các hệ thống dần dần từng cái một

**1 C**

**2 A**

**3 D**

**4 A**

**5 A**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 10: Kiểm thử hệ thống**

Kiểm thử hệ thống: là kiểm thử trên 1 hệ thống đầy đủ (whole system – nghĩa là toàn bộ các chức năng, nghiệp vụ của hệ thống đều đã có đầy đủ) như đã xác định trong scope của phần mềm

**Tại sao phải thực hiện kiểm thử hệ thống:**

* Để kiểm tra , đảm bảo rằng các chức năng, đặc tính của 1 PM được phát triển đúng và đủ theo SRS của sản phẩm đó
* Được thực hiện trên 1 môi trường gần giống môi trường thật
* Là bước kiểm thử cuối cùng đại diện cho **dự án, nhóm phát triển phần mềm** để kiểm tra sản phẩm trước khi bàn giao cho khách hàng

**Kiểm thử hệ thống được thực hiện khi nào?**

* Hệ thống phần mềm đã hoàn thiện, phát triển xong
* Regression test và unit test đã được thực hiện đầy đủ
* Các tài liệu đặc tả hệ thống đã hoàn thành và không còn sự thay đổi
* Các documents, sản phẩm hỗ trợ cho việc kiểm thử hệ thống cũng như các test cases, test data hay lịch trình kiểm thử đã phải sẵn sàng để sử dụng

**Phương thức, cách thức thực hiện khi system testing**

* Kiểm thử chức năng: sử dụng phương thức kiểm thử hộp đen
* Kiểm thử phi chức năng: kiểm tra hiệu năng, độ tin cậy
* Kiểm thử việc cài đặt

**Người thực hiện: tester**

**1 A**

**2 D**

**3 C**

**4 C**

**5 D**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 11: kiểm thử chấp nhận (Acceptance Testing)**

* Còn được biết là kiểm thử người dùng chấp nhận – user acceptance testing: là hoạt động kiểm thử được thực hiện bởi người dùng cuối cùng, hay khách hàng với mục tiêu là kiểm tra sản phẩm có đáp ứng được như những yêu cầu của người sử dụng và thõa màn các tiêu chí chấp nhận sản phẩm đã thống nhất hay không
  + Là hoạt động kiểm thử cuối cùngtrong quy trình kiểm thử sản phẩm phần mềm
* Tại sao phải thực hiện kiểm thử chấp nhận?
  + Để chấp nhận sản phẩm dựa trên các tiêu chí đã được đề ra từ trước
  + Đảm bảo các chức năng cần có và các chức năng mong muốn trong sản phẩm đều đã được phát triển và được bao gồm trong sản phẩm phần mềm
* Thực hiện khi nào?
  + Thường được thực hiện sau khi sản phẩm phần mềm được bàn giao cho khách hàng và sản phẩm đó đã được kiểm thử hệ thống
  + Tuy nhiên do việc thực hiện AC là do khách hàng đảm nhận, hay ngb sử dụng sản phẩm đó trong thực tế, do đó kiểm thử chấp nhận được thực hiện khi nào còn tùy thuộc vào yêu cầu của khách hàng
  + Đôi khi AC sẽ hơi trùm lên ST (system testing), nghĩa là đôi khi phát triển sản phẩm đang thực hiện kiểm thử hệ hệ thống thì khách hàng muốn tham gia vào thực hiện việc kiểm thử - *alpha testing*

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Các loại kiểm thử chấp nhận**

1. AC operational (kiểm thử chấp nhận hoạt động): đảm bảo các khía cạnh phi chức năng của một hệ thống được kiểm tra đảm bảo rằng hệ thống đó thỏa mãn các thông số cụ thể đã thống nhất. AC operational bao gồm các kiểm thử sau:
   1. Sao lưu – khôi phục: sao lưu và khôi phục dữ liệu khi có vấn đề xảy ra sẽ tránh được mất mát dữ liệu
   2. Sự khô phục sau thảm họa: nếu có 1 sự cố thảm họa xảy ra, hệ thống ở địa điểm đó được phục hồi, khôi phục như thế nào
   3. Các việc liên quan đến bảo trì: như hệ thống được nâng cấp, sửa chữa các vấn đề trong quá trình vận hành
   4. Các điểm yếu về bảo mật: ví dụ như có tấn công từ virus vào hệ thống thì sau đó hệ thống được phục hồi như thế nào …
2. Compliance AC (kiểm tra tính tuân thủ trong AC): kiểm thử các tiêu chí liên quan đến chính sách, quy định đã thống nhất từ ban đầu như các chính sách của chính phủ, pháp luật … Thường có 2 giai đoạn:
   1. Alpha testing: thực hiện bởi khách hàng ở development environment
   2. Beta testing: thực hiện bởi khách hàng tại môi trường của khách hàng, môi trường giống hoặc gần giống với thực tế

**1 B: sai / D**

**2 D: sai / B**

**3 A**

**4 B**

**5 C**

**--------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- ---------------**

**Phần 4: các loại hình kiểm thử**

**Bài 12: các loại kiểm thử chức năng**

**Định nghĩa của test type**

* Khi thực hiện kiểm thử cho một dự án, có nhiều yêu cầu khác nhau về sản phẩm phần mềm cần kiểm thử, với mỗi loại yêu cầu cụ thể cần 1 loại kiểm thử tương ứng. Như vậy có thể hiểu môi giai đoạn kiểm thử sẽ giải quyết cho việc kiểm thử một vấn đề cụ thể. Trong mỗi giai đoạn kiểm thử có thể dùng nhiều loại kiểm thử khác nhau để đạt được mục tiêu kiểm thử cho giai đoạn đó

**Kiểm thử được phân loại như thế nào?**

* Các loại kiểm thử chức năng: mục tiêu là test các chức năng của 1 sản phẩm phần mềm . Dùng từ *các loại* vì chức năng cũng có thể có nhiều loại và tương ứng với mỗi loại sẽ có những loại kiểm thử khác nhau
  + Chức năng bình thường
  + Chức năng chạy ngầm
  + …
* Các loại kiểm thử phi chức năng: thông thường là kiểm tra các tính chất của 1 hệ thống hay sản phẩm như hiệu suất, tính bảo mật …
* Các loại kiểm thử cấu trúc
* Các loại kiểm thử cho trường hợp sản phẩm có sự thay đổi (kiểm thử lại, kiểm thử hồi quy)

**Kiểm thử chức năng**

* KTCN tập trung vào các chức năng của sản phẩm có hoạt động đúng như yêu cầu. KTCN tập trung vào các tương tác, các xử lý từ bên ngoài vào hệ thống phần mềm
* Có thể được sử dụng ở bất kỳ giai đoạn kiểm thử nào, từ UT đến AC
* Có 2 cách tiếp cận:
  + Dựa vào đặc tả
  + Dựa vào việc sử dụng: dựa trên việc sử dụng hàng ngày của các chức năng, các business của hệ thống của người sử dụng để thực hiện việc kiểm thử chức năng

**Các loại kiểm thử chức năng**

* Kiểm thử chức năng:
  + Đảm bảo đúng mục tiêu của KTCN bao gồm:
    - Mở chức năng
    - Nhập dữ liệu
    - Xử lý
    - Lấy và kiểm tra kết quả trả về
    - …
* Kiểm thử giao diện người dùng (UI testing): kiểm tra giao diện của chức năng có đúng như thiết kế mong muốn hay không
* Kiểm thử sự tích hợp giữa dữ liệu và CSDL (data & data integrity tétting): kiểm tra xem các chức năng của 1 sản phẩm phần mềm hoạt động có đúng không sau khi sản phẩm phần mềm đó đã có sự tích hợp hay việc đưa những dữ liệu cũ, dữ liệu có sẵn từ bên ngoài vào sản phẩm
* Kiểm thử vòng lặp công việc (business cycle testing): đảm bảo hoạt động của các chức năng cụ thể được hoạt động đúng như yêu cầu và lịch trình
* Kiểm thử kiểm soát truy cập (access control testing): kiểm tra để đảm bảo:
  + Các actor chỉ có quyền truy cập vào những chức năng hay những vùng mà họdc phân quyền
  + Chỉ những user được phân quyền mới có thể truy cập vào hệ thống thông qua các gateway thích hợp

**1 B**

**2 C: sai / B: sai / A**

**3 D**

**4 C**

**5 C: sai / A: sai / B**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 13: các loại kiểm thử phi chức năng**

* Là kiểm thử xem sản phẩm phần mềm có những đặc điểm, đặc tính tốt như thế nào (how well)
* Cũng có thể được sử dụng ở bất cứ giai đaonj nào như **tốt nhất** nên là ở **system test (**kiểm thử hệ thống) hoặc **user acceptance test** (kiểm thử nghiệm thu sản phẩm) vì đây là những giai đoạn này đặc tính sản phẩm, hệ thống mới là tổng hợp nhất, giống với thực tế nhất, các chức năng lúc này mới hoạt động ổn định nhất.

**Các loại kiểm thử Phi chức năng**

* Có rất nhiều các loại kiểm thử phi chức năng, tuy nhiên chúng ta sẽ nói đến 4 loại chính:
  + Kiểm thử hiệu năng (performance testing):
  + Kiểm thử tải (load tesrting)
  + Kiểm thử tập trung (tra tấn) – stress testing
  + Kiểm thử với lượng dữ liệu lớn – volumn testing

1. Kiểm thử hiệu năng:
   1. Kiểm thử để xác định 1 sản phẩm phần mềm thực hiện các chức năng cụ thể nhanh như thế nào
   2. Cũng có thể sử dụng để xác nhận và xác minh những thuộc tính chất lượng khác của hệ thống như khả năng mở rộng, độ tin cậy, sử dụng tài nguyên
   3. Như vậy nếu như trong yêu cầu của khách hàng về sản phẩm phần mềm có yêu cầu về response time của các chức năng thì phải áp dụng loại hình kiểm thử này
2. Kiểm thử tải:
   1. Có thể gọi là kiểm thử đồng thời (nhiều người dùng hệ thống đồng thời)
   2. Xác định hiệu suất của hệ thống, sản phẩm trong điều kiện tải cụ thể
3. Stress testing:
   1. Là việc xác định đặc tính hiệu suất của hệ thống dưới các điều kiện ngặt nghèo như:
      1. Số lượng người dùng đồng thời tối đa
      2. Bộ nhớ tối thiểu
      3. Ổ cứng không đủ dung lượng
   2. Kiểm tra xem giới hạn đặc tính của sản phẩm phần mềm ở đâu
   3. Khác với load testing là load testing phải kiểm thử hệ thống dưới 1 điều kiện tải **cụ thể,** còn stress testing lại phải tìm điểm ngưỡng mà hệ thống không chịu được tải đó
4. Volumn testing – kiểm thử với lượng dữ liệu lớn
   1. Tập trung vào việc xác định hoặc xác nhận đặc tính hiệu suất của hệ thống trong điều kiện là hệ thống có lượng dữ liệu rất lớn, có thể là CSDL lớn hoặc dữ liệu trong file được upload lên hệ thống lớn

**1 C**

**2 B**

**3 D**

**4 A**

**5 B**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 14: các loại kiểm thử cấu trúc**

**Kiểm thử cấu trúc**

* Là kiểm thử cấu trúc của một hệ thống hoặc 1 chức năng, 1 hàm trong hệ thống
* Còn gọi là **kiểm thử hộp trắng (white box test)** vì trong kiểm thử cấu trúc, kiểm thử viên phải kiểm thử cái gì đang diễn ra, cái gì đang hoạt động, đang được thực hiện **bên trong** hệ thống
* Phải có kiến thức về lập trình, biết đọc code và hiểu được các dòng code đang xử lý việc gì và luồng đi của nó như thế nào thì mới kiểm thử được
* Được thực hiện từ giai đaonj UT, thường do devs làm

**Các loại kiểm thử cấu trúc**

* Statement testing: kiểm thử toàn bộ câu lệnh, dòng code được viết ra ít nhất 1 lần
* Decision testing: kiểm thử toàn bộ quyết định, các nhánh trong đoạn code được viết ra
* Condition testing: kiểm thử toàn bộ các điều kiện trong code được viết ra
* Path testing: kiểm thử toàn bộ các đường đi từ điểm bắt đầu đến kết thúc trong code được viết ra

**1 D**

**2 B**

**3 A**

**4 A**

**5 B**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 15: Các loại kiểm thử sự thay đổi**

**Kiểm thử liên quan đến sự thay đổi**

Định nghĩa:

* Thực hiện kiểm thử khi có sự thay đổi **bên trong** sản phẩm phần mềm
* Sự thay đổi này có thể là:
  + Sửa chữa các lỗi tim được thông qua các hoạt động đảm bảo chất lượng sản phẩm như kiểm thử (testing) hay đánh giá (review)
  + Có sự thay đổi về chức năng

Các loại kiểm thử sự thay đổi:

* Kiểm thử xác nhận (confirmation testing): hay còn gọi là kiểm thử lại (re-testing)
* Kiểm thử hồi quy (regression test)

1. Kiểm thử xác nhận:
   1. Nếu có lỗi được ghi nhận khi thực hiện các tình huống kiểm thử, nếu lỗi đó được sửa chữatrong phiên bản mới nhất, kiểm thử viên sẽ thực hiện kiểm thử lại tình huống kiểm thử đó và kiểm tra xem lỗi đã ghi nhận đã được sửa chữa hay chưa 🡺 kiểm thử xác nhận là hành động kiểm thử lại để xác nhận tình huống kiểm thử mà không đạt lúc trước đã đạt hay chưa
   2. Trong tình huống kiểm thử không đạt mà có nhiều lỗi được tìm thấy, kiểm thử viên cũng phải thực hiện kiểm thử xác nhận để xem toàn bộ các lỗi tìm được lúc trước của tình huống kiểm thử không đạt đã được sửa chữa hay chưa
   3. Khi thực hiện CT, quan trọng là phải đảm bảo rằng thực hiện kiểm thử theo đúng các bước đã được mô tả trong test cases, sử dụng cùng đầu vào, cùng dữ liệu và môi trường
2. Kiểm thử hồi quy:
   1. Thực hiện kiểm tra các sự thay đổi của sản phẩm, đảm bảo rằng những thay đổi này không làm ảnh hưởng đến những phần đã hoàn thiện trước đó của sản phẩm
   2. Kiểm thử hồi quy được thực hiện khi:
      1. Sản phẩm cần kiểm thử có sự thay đổi (thêm tính năng mới, sửa đổi tính năng cũ, thay đổi thiết kế … )
      2. Môi trường sản phẩm đang được deployed có sự thay đổi
   3. Đối với 1 sản phẩm phần mềm được phát triển theo mô hình phát triển tăng dần (iterative development model) thì test leader của dự án đó cũng phải tính đến việc thực hiện kiểm thử hồi quy.
   4. Đối với các loại dự án bảo trì cũng tương tự do qua thời gian có thể có lỗi hoặc nhu cầu về tính năng của sản phẩm được nâng cấp, lúc đó cũng cần phải thực hiện regression test
   5. Trong kiểm thử hồi quy, tester sẽ phải tận dụng các test cases đã thiết kế để kiểm thử các chức năng không bị thay đổi để kiểm tra xem các thay đổi có ảnh hưởng đến việc hoạt động của các chức năng cũ hay không
   6. Do việc thực hiện regression test sẽ tốn nhiều thời gian, công sức của các kiểm thử viên, do đó sẽ trong regression test nếu có thể chuyển thành kiểm thử tự động là tốt nhất

**1 C**

**2 B**

**3 B**

**4 D**

**5 A**

**--------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- ---------------**

**Phần 5: kế hoạch kiểm thử**

**Bài 16: Kế hoạch kiểm thử**

**Kế hoạch kiểm thử**

Định nghĩa:

* Test plan là một tài liệu mô tả test scope, chiến được kiểm thử, nguồn lực và lịch trình của các hoạt động kiểm thử cho việc kiểm thử trong một dự án, sản phẩm phần mềm
* Tài liệu này chính là sản phẩm được sử dụng để trao đổi và thống nhất với các stakeholders về kế hoạch, chiến lược, nguồn lực trong việc thực hiện kiểm thử cho dự án

Các mục tiêu của kế hoạch kiểm thử:

* Test scope: mô tả trong dự án này sẽ thực hiện kiểm thử cái gì, không thực hiện kiểm thử cái gì trong giai đoạn nào
* Các rủi ro có thể có trong quá trình thực hiện kiểm thử
* Các tiêu chí để hoàn thành việc kiểm thử, hay các tiêu chí để KH chấp nhận sản phẩm đã kiểm thử
* Lấy được sự đồng thuận về cach thức thực hiện kiểm thử cho sản phẩm từ những nhóm, nx người liên quan đến phát triển sản phẩm
* Mang lại sự hợp tác hiệu quả giữa các nhóm liên quan đến phát triển sản phẩm
* Chiến lược, các tiếp cận để thực hiện kiểm thử cho sản phẩm, dự án dựa trên phạm vi kiểm thử đã xác định
* Nguồn lực để gỗ trợ việc kiểm thử (con người và hệ thống)
* Các chỉ số để đánh giá hoạt động kiểm thử cho sản phẩm, dự án đó
* Các tài liệu, sản phẩm khi thực hiện kiểm thử cho sản phẩm sẽ phải tạo ra như test plan, các test cases hay test reports và lịch trình đưa ra các sản phẩm, tài liệu đó

**Quy trình tạo ra test plan**

Quá trình tạo ra một kế hoạch kiểm thử, người ta gọi là test planning

Đầu vào của quá trình này bao gồm:

* Kế hoạch dự án: tài liệu tổng hợp về các thông tin của dự án
* Yêu cầu của KH vầ các tiêu chí để KH chấp nhận sản phẩm phần mềm

Đầu ra: test plan

**Cấu trúc, thành phần của một sản phẩm, tài liệu kế hoạch kiểm thử**

Thông thường một tài liệu kế hoạchkiểm thử sẽ gồm 7 phần sau:

1. Giới thiệu chung về tài liệu
2. Acceptance criteria: các tiêu chí để chấp nhận sản phẩm
3. Requirement for test: các yêu cầu cần kiểm thử
4. Test strategy: chiến lược kiểm thử cho sản phẩm
5. Resource: các nguồn lực hỗ trợ cho việc kiểm thử
6. Test milestones: các cột mốc, lịch cho các hoạt động kiểm thử
7. Deliverables: các sản phẩm kiểm thử sẽ cần phải bàn giao

**1 D**

**2 D / C**

**3 C / A**

**4 C**

**5 A**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 17: chiến lược kiểm thử**

Định nghĩa:

* Là một mô tả chính thức về cách thức 1 sản phẩm phần mềm sẽ được kiểm thử như thế nào.
  + Ví dụ một chiến lược kiểm thử được trình bày 1 cách đơn giản như: “Chúng ta sẽ sử dụng kiểm thử hộp đen, kiểm thử hộp trắng, kiểm thử biên … để kiểm thử sản phẩm này dựa trên SRS của nó”
* Chiến lược kiểm thử có thể được xây dựng riêng lẻ cho từng giai đoạn kiểm thử nếu có yêu cầu, tương ứng với việc xây dựng các test plan cho từng giai đoạn. Ví dụ có tài liệu cho kế hoạch UT, test plan cho integration test hay test plan cho AC. Thông thường chỉ cần có 1 chiến lược kiểm thử cho toàn bộ dự án, và nằm trong tài liệu kế hoạch kiểm thử

**Các bước để xây dựng, xác định chiến lược kiểm thử**

1. Bước đầu tiên là xác định được các yêu cầu kiểm thử, vì từ những yêu cầu này mới có thể đưa ra cách thức thực hiện, và triển khai việc kiểm thử theo những yêu cầu đó, hay chính là đưa ra chiến lược kiểm thử
2. Bước thứ 2: từ các yêu cầu kiểm thử đã xác định tiến hành xây dựng chiến lược kiểm thử, bao gồm:
   1. Lựa chọn các **loại kiểm thử** thích hợp cho yêu cầu kiểm thử đó
   2. Sau khi đã xác định được danh sách các loại kiểm thử sẽ thực hiện để thỏa mãn tất cả các yêu cầu kiểm thử thì sẽ phải xác định xem loại kiểm thử nào phù hợp thực hiện trong giai đoạn kiểm thử nào. Ví dụ:
      1. Giai đoạn ST thì loại kiểm thử như UI testing, functional testing hay performance testing có thể được thực hiện
      2. Nhưng trong giai đoạn UT thì lại rất khó để thực hiện kiểm thử hiệu năng
   3. Khi đã xác định được loại kiểm thử rồi thì cần xác định tất cả các công cụ hỗ trợ hoạt động kiểm thử cho dự án

**Xác định các yêu cầu kiểm thử**

* Dựa vào các yêu cầu từ KH, SRS, thông qua email, các buổi meeting … người lập ra kế hoạch và chiến lược kiểm thử phải xác định được các yêu cầu kiểm thử
* Thông thường yêu cầu kiểm thử được phân chia 2 loại:
  + Các yêu cầu kiểm thử chức năng
  + Các yêu cầu kiểm thử phi chức năng

**1 C**

**2 C**

**3 A**

**4 C**

**5 A**

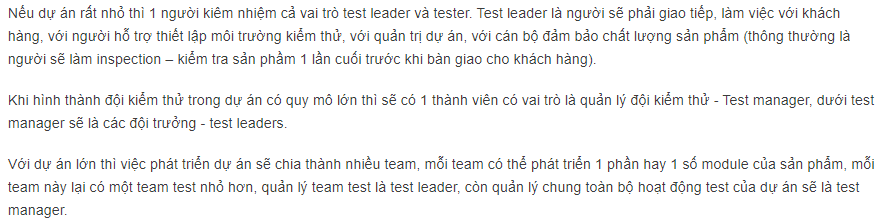
**--------------- --------------- --------------- --------------- --------------- ---------------**

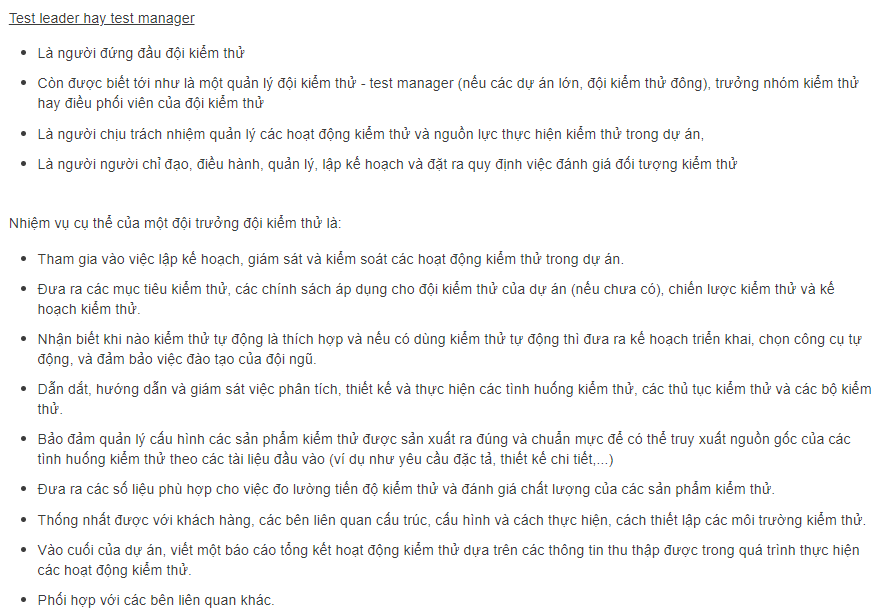
**Bài 18: tổ chức kiểm thử**

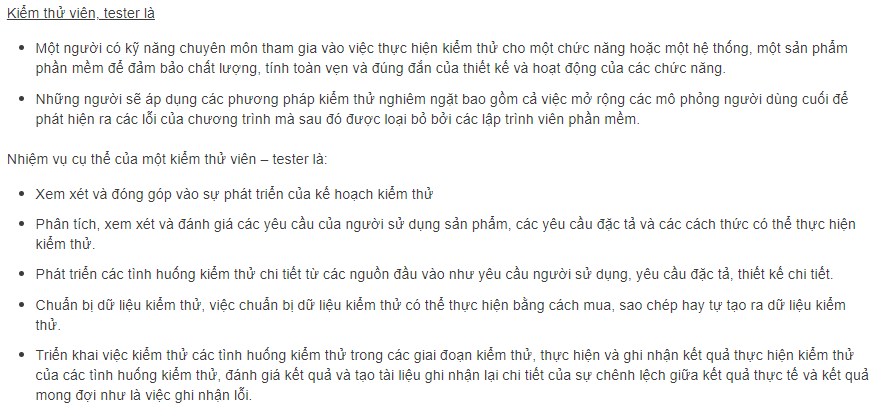
**Tổ chức đội kiểm thử**

Thông thường một đội kiểm thử sẽ bao gồm các thành viên như sau:

1. Quản lý đội kiểm thử (test manager – test leader): quản lý đội kiểm thử, chịu trách nhiệm việc lập ra test plan và quản trị, kiểm soát để test plan đó được thực hiện đúng theo các mục tiêu
2. Kiểm thử viên: là người thực hiện kiểm thử trong dự án, nếu dự án có thực hiện kiểm thử tự động thì tester có thể tách làm 2 vai trò:
   1. Các tester kiểm thử tự động
   2. Các manual tester







**1 A / D**

**2 B / A**

**3 A / C**

**4 D / C / A / B**

**5 C**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 19: Dự toán kiểm thử**

**Dự toán kiểm thử là gì?**

Là quá trình dự toán, tính toán chi phí kiểm thử, công sức dành cho việc kiểm thử cũng như thời gian cần dành hco việc kiểm thử 1 dự án cụ thể về kiểm thử phần mềm

Để có được dự toán chính xác nhất, người dự toán phải:

1. Có kỹ năng dự toán: là một kỹ năng quan trọng của người làm quản lý, trưởng nhóm kiểm thử hay cả đối với một chuyên gia kiểm thử.
2. Đặt câu hỏi cho bản thân và trả lời
   1. Những công việc gì cần thực hiện, cần làm?
   2. Để thực hiện những công việc đó cần bỏ ra công sức như thế nào?

**Phương pháp dự toán kiểm thử**

1. Dựa vào các chỉ số (metrics based): là phương pháp dựa vào việc phân tích các chỉ số của các dự án đã thực hiện trong quá khứ và các chỉ số được thống kê trong lĩnh vực phát triển phần mềm
2. Dựa vào tham vấn các chuyên gia kiểm thử và từ những người sẽ làm, sẽ thực hiện công việc kiểm thử về các công việc sẽ thực hiện rồi đưa ra các con số dự đoán

**Các nhân tố ảnh hưởng đến việc dự toán kiểm thử**

1. Các nhân tố sản phẩm:
   1. Các sản phẩm đầu vào, các tài liệu đầu vào để có thể thực hiện công việc kiểm thử có đầy đủ không
   2. Độ lớn và phức tạp của sản phẩm cần kiểm thử
2. Các nhân tố quy trình:
   1. Sự sẵn sàng của các công cụ kiểm thử
   2. Mô hình phát triển phần mềm hay vòng đời của dự án: ví dụ mô hình phát triển tăng dần thì công sức dành cho việc kiểm thử sẽ nhiều hơn mô hình chữ V do sẽ cần thêm kiểm thử hồi quy
3. Các nhân tố khác
   1. Áp lực thời gian và khả năng của người thực hiện
   2. Số lượng các vòng, lần kiểm thử cho sản phẩm: số lần, số vòng thực hiện kiểm thử sẽ ảnh hưởng để dự toán kiểm thử

**1 B**

**2 B**

**3 A**

**4 C**

**5 A**

**--------------- --------------- --------------- --------------- --------------- ---------------**

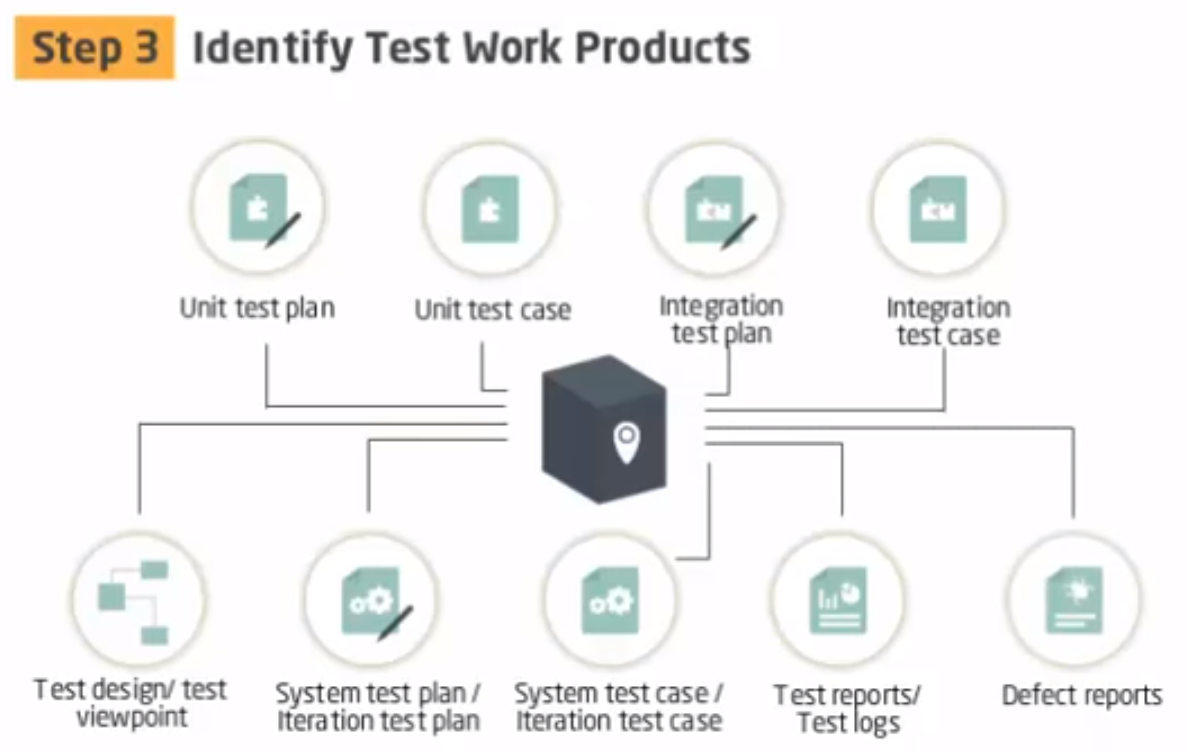
**Bài 20: Mốc kiểm thử (Test Milestones)**

**Mốc kiểm thử**

Mốc kiểm thử chỉ ra một sự kiện, một thành tích kiểm thử quan trọng đạt được hay cần phải đạt được trong 1 dự án

Mỗi mốc kiểm thử sẽ:

* Bao gồm ít nhất 1 hoạt động kiểm thử
* Tạo ra một hoặc nhiều sản phẩm kiểm thử



**Quy trình xác định các mốc kiểm thử**

Để xác định được các mốc kiểm thử phải dựa vào các inputs sau:

* Kế hoạch dự án với các mốc dự án đã xác định và mô hình phát triển phần mềm
  + Bạn sẽ phải quan tâm khi nào sản phẩm hay yêu cầu đặc tả có thế sử dụng
  + Khi nào hoạt động kiểm thử đầu tiên có thể thực hiện
  + Khi nào thì cần bàn giao sản phẩm gì
  + …
* Đầu vào thứ 2 là bản dự toán kiểm thử, dự toán công sức thực hiện cho các công việc kiểm thử cụ thể

Các bước để xác định các mốc kiểm thử

1. Xác định mô hình phát triển phần mềm, hay mô hình triển khai dự án
2. Xác định các mốc dự án
3. Xác định các sản phẩm kiểm thử sẽ phải bàn giao cho mỗi mốc dự án
4. Xác định các mốc kiểm thử và mô tả chúng

**Mô tả, diễn tả các mốc kiểm thử**

Các mốc kiểm thử sẽ được mô tả trong 1 bảng gồm những thông tin sau:

* Tên mốc kiểm thử: thường nên gắn với tên của sản phẩm chính cần hoàn thiện trong mốc đó.
  + Thứ tự của mỗi mốc kiểm thử nên theo quy trình kiểm thử đã được công ty ban hành trước đó, ví dụ không được để ”tạo test case cho hệ thống” trước mốc “tạo kế hoạch kiểm thử”.
  + Kinh nghiệm là chúng ta đặt cột mốc quan trọng trong lịch trình để quản lý kỳ vọng, mong muốn của các bên liên quan
* Công sức thực hiện để hoàn thành mốc kiểm thử: dựa trên số liệu được dự toán, tuy nhiên không phải điền đúng số lượng công sức được điền trong kế hoạch dự án (nếu có) và cũng không được điền số lượng công sức theo lịch
* Ngày bắt đầu mốc kiểm thử: không chỉ dựa vào kế hoạch dự án mà còn phụ thuộc vào quy trình kiểm thử. Dựa vào quy trình để biết được khi thực hiện hoạt động kiểm thử nào trong mốc kiểm thử thì cần inputs là gì. Khi mà các đầu vào được đáp ứng thì mới có thể để là ngày bắt đầu mốc kiểm thử
* Ngày kết thúc mốc kiểm thử: Dựa trên sự thống nhất với KH và các stakeholder về các ngày bắt buộc phải bàn giao sản phẩm kiểm thử cần phải hoàn thành trong mỗi mốc kiểm thử

**1 D / B**

**2 D**

**3 D**

**4 C**

**5 D**

**--------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- --------------- ---------------**

**Phần 6: Thiết kế và phát triển các tình huống kiểm thử**

**Bài 21: mô tả tình huống kiểm thử (test cases)**

**Tình huống kiểm thử**

Tình huống kiểm thử:

* Là một tập các giá trị đầu vào, các điều kiện cần phải có để thực hiện được tính huống kiểm thử đó, các kết quả mong muốn mong muốn, được phát triển cho 1 chức năng của sản phẩm phần mềm để kiểm tra xem chức năng đó có hoạt động đúng như yêu cầu đặc tả hay không
* Để thiết kế được các test cases phải dựa vào các đầu vào sau:
  + SRS
  + Thiết kế chi tiết (screen design, database design, …)
  + Kế hoạch kiểm thử
* Tình huống kiểm thử có thể phần thành
  + High level test cases (logical test case, abstract test case): những test cases này không chứa giá trị chính xác cho các inputs và expected output
  + Low level test cases: là những test cases phải xác định giá trị cụ thể cho input và giá trị cụ thể cho expectation output

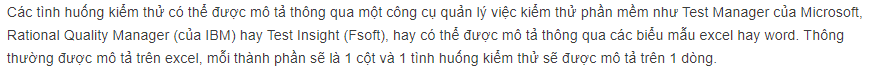
**Mô tả tình huống kiểm thử**

Khi mô tả một test cases thì cần:

* Số hiệu của test case
* Title của test case
* Description của test case
* Expected output
* Kết quả kiểm thử

Ngoài ra có thể cân nhắc các thành phần sau:

* Tên chức năng mà tình huống kiểm thử đó đang mô tả
* Điều kiện trước: điều kiện để tình huống kiểm thử đó có thể thực hiện được
* Dữ liệu kiểm thử
* Kết quả thực tế khi thực hiện kiểm thử
* Mức độ ưu tiên hay độ quan trọng của tình huống kiểm thử, hữu ích khi phải thực hiện kiểm thử trong thời gian ngắn
* …



**Các loại test cases có thể có khi thiết kês các test case cho một chức năng của sản phẩm phần mềm**

Chúng ta cần phải biết cấu trúc hay các loại tình huống kiểm thử có thể có để kiểm thử cho 1 chức năng của sản phẩm phần mềm .

**Các tình huống kiểm thử cho một chức năng gồm:**

* Các tình huống kiểm thử giao diện của chức năng đó
* Kiểm thử chức năng, hay kiểm tra các quy tắc nghiệp vụ của chức năng đó
* Các tình huống kiểm thử kiểm tra dữ liệu đầu vào của chức năng đó
* Các tình huống kiểm thử quyền truy cập trên chức năng đó
* Các tình huống kiểm thử tích hợp với các chức năng hay phần mềm khác nếu có

**Khi thiết kế các tình huống kiểm thử giao diện,** phải dựa vào thiết kế màn hình, hoặc màn hình mẫu. Các tình huống kiểm thử giao diện phải có là:

* Kiểm tra kích cỡ, vị trí, các menu, màu sắc, kiểu chữ … của các đối tượng hiển thị trên màn hình
* Kiểm tra các đối tượng trên màn hình xem:
  + Đối tượng đó kiểu gì
  + Đối tượng có thể nhập hay sửa dữ liệu không
  + Đối tượng đó có phải là input bắt buộc phải nhập không
  + Đối tượng đó khi hiển thị lần đầu thì có giá trị mặc định không
  + Đối tượng đó có chế độ ẩn hiện không
* Các phương thức truy cậpL sử dụng phím **tab** trên bàn phím, chuột, hotkeys …

**Khi thiết kế test cases cho chức năng, quy tắc nghiệp vụ** thì phải đảm bảo có các

* Test cases với dữ liệu đúng, chức năng thực hiện thành công
* Test cases kiểm thử dựa trên các qui tắc nghiệp vụ
* Sử dụng các kỹ thuật kiểm thử hộp đen để thiết kế các tình huống kiểm thử để kiểm tra nghiệp vụ của chức năng, các kỹ thuật có thể dùng:
  + Kỹ thuật phân lớp tương đương
  + Kỹ thuật phân tích giá trị biên
  + Kỹ thuật bảng quyết định
  + Kỹ thuật chuyển dịch trạng thái

**Thiết kế các test cases kiểm tra dữ liệu đầu vào của chức năng:** tùy thuộc vào từng kiểu dữ liệu input để thiết kế test cases, dữ liệu đầu vào có thể là dữ liệu ở:

* Trường text
* Trường kiểu số
* Trường kiểu ngày
* …

**Để thiết kế các tình huống kiểm thử quyền truy cập** thì phải dựa vào bảng mô tả giữa vai trò và quyền trên các chức năng. Thiết kế các tình huống kiểm thử để đảm bảo:

* Mỗi roles, mỗi user truy cập vào hệ thống đều được phân quyền cụ thể và đều được kiểm tra xem có quyền và không có quyền trên các chức năng cụ thể hay không
* Trường hợp người dùng có nhiều vai trò, hay nhiều quyền kết hợp thì hoạt động như thế nào

**1 C**

**2 A**

**3 C**

**4 C**

**5 C / B**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 22: Các kỹ thuật kiểm thử**

**Kỹ thuật kiểm thử**

Ở mức cao nhất, các kỹ thuật kiểm thử được phân ra thành kiểm thử tĩnh (static testss) và kiểm thử động (dynamic tests):

* Kiểm thử tĩnh: là các kiểm thử được thực hiện mà không cần chạy hoặc thực thi các đối tượng kiểm thử, ví dụ hoạt động review documents, source code sẽ là hoạt động kiểm thử tĩnh, sản phẩm cần kiểm thử chỉ là các trang tài liệu tĩnh, tester không cần phải chạy sản phẩm đó
* Kiểm thử động: ngược lại với static test, người kiểm thử cần chạy hoặc thực thi các đối tượng kiểm thử

Static tests có thể được thực hiện bằng cách reviews và static analysis

* Review là hoạt động thực hiện bởi con người, người kiểm tra sẽ đọc kỹ sản phẩm cần kiểm tra để tìm ra lỗi
* Static analysis là hoạt động kiểm tra sản phẩm tĩnh (như tài liệu, source code) bằng 1 công cụ hay tool

Các kỹ thuật kiểm thử động ay cách thức thực hiện kiểm thử động thường được thực hiện theo 5 cách sau:

* Kiểm thử hộp đen: kiểm thử dựa trên đặc tả, hay dựa trên thói quen sử dụng, chúng ta sẽ kiểm thử dựa vào yêu cầu đặc tả, mô tả chi tiết yêu cầu của hệ thống, trên cách mà hệ thống dự định sẽ thực hiện
* Kiểm thử hộp trắng: hay còn gọi là structural testing, là cách chúng ta kiểm thử dựa trên cách hệ thống được phát triển, chỉ được thực hiện tại giai đaonj kiểm thử đơn vị (UT)
* Kiểm thử dựa trên kinh nghiệm, trải nghiệm: kiểm thử dựa trên kỹ năng và trực giác của chính chúng ta, cùng với kinh nghiệm, trải nghiệm chúng ta đã có với các hệ thống cùng lĩnh vực hoặc công nghệ.
* Dynamic analysis: thực hiện việc kiểm thử bằng các công cụ hỗ trợ - test tool
* Kiểm thử dựa trên các lỗi đã biết, đã có từ trước – defect bases: sử dụng sự hiểu biết về các loại lỗi để thiết kế ra các tình hình kiểm thử

**Các kỹ thuật kiểm thử hộp đen**

**Các kỹ thuật kiểm thử hộp đen bao gồm:**

1. Equivalence partitioning – kỹ thuật phân lớp tương đương: kỹ thuật thiết kế các tình huống kiểm thử bằng cách phân chia hệ thống thành nhều nhóm, nhiều lớp mà trong mỗi lớp / nhóm đó hệ thống sẽ được kỳ vọng là được kiểm soát theo cùng 1 cách và có cùng xử lý giống nhau.Các nhóm / lớp này được gọi là các nhóm / lớp ngang bằng hay tương đương
2. Boundary values analysis – kỹ thuật phân tích các giá trị biên:kỳ thuật bổ sung cho kỹ thuật phân lớp tương đường bằng cách kiểm thử các điểm biên của các nhóm tương đương
3. Decision table testing – kỹ thuật bảng quyết định: các test cases được thiết kế dựa trên việc kết hợp các input và / hoặc các nguyên nhân được thể hiện ở trong bảng quyết định.
4. State transition testing – kỹ thuật kiểm thử dựa trên sự thay đổi trạng thái: các test cases được thiết kế dựa vào việc dịch chuyển các trạng thái của hệ thống
5. Use case testing – kỹ thuật kiểm thử dựa trên các tình huống, các use cases của hệ thốn
6. Pairwise testing: các test cases được thiết kế dựa trên tất cả các kết hợp có thể có giữa các cắp biến đầu vào của hệ thống
7. Classification tree method: các test cases được thiết kế dựa trên 1 cây phân loại thể hiện tất cả các kết hợp có thể có giữa ít nhất là 2 đầu vào và / hoặc đầu ra

**Các kỹ thuật kiểm thử hộp trắng**

Các kỹ thuật kiểm thử hộp trắng hay kiểm thử cấu trúc bao gồm:

1. Statement testing – kiểm thử tất cả các câu lệnh: các tình huống kiểm thử được thiết kế dựa trên các câu lệnh trong code đã được lập trình
2. Decision testing – kiểm thử tất cả các quyết định, các rẽ nhánh: các test cases được thiết kế dựa trên kết quả của các quyết định hay rẽ nhánh trong code
3. Condition dêtrrmination testing – kiểm thử điều kiện: các test cases được thiết kế dựa trên sự ảnh hưởng của 1 điều kiện đơn tham gia vào quyết định so với chính quyết định đó
4. Path testing – kiểm thử các đường đi: các tình huống kiểm thử được thiết kế để kiểm tra các đường đi trong một hàm, một thủ tục của code. Gọi là một đường đi vì nó tính từ điểm bắt đầu đến điểm kết thúc

**1 C / A**

**2 A**

**3 D**

**4 B / A / C**

**5 C**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 23: kỹ thuật kiểm thử: phân lớp tương đương và phân tích điểm biên**

**Kỹ thuật kiểm thử: phân lớp tương đương**

Đây là một kỹ thuật kiểm thử hộp đen (black-box test technique) mà các test cases sẽ được thiết kế đại diện cho các class hay vùng tương đương nhau

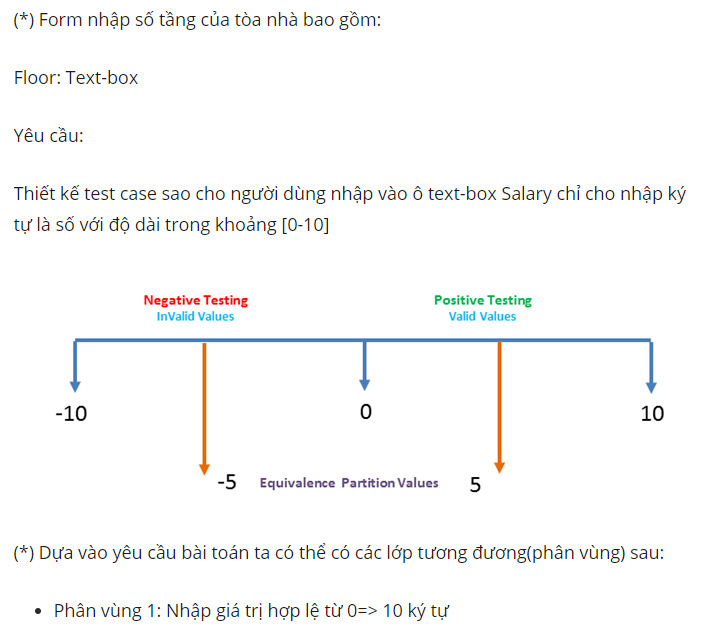
Khi thực hiện kiểm thửm cần xác định được

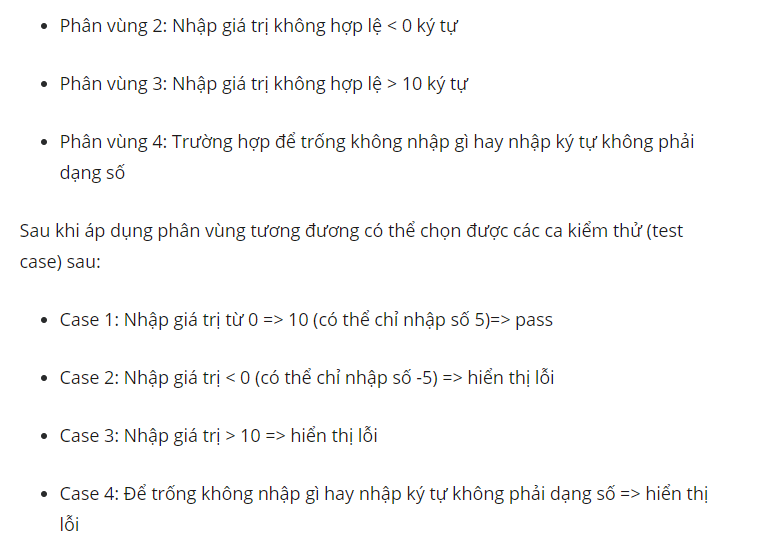
* Dữ liệu đầu vào (inputs)
* Môi trường thực hiện kiểm thử
* Cách hệ thống xử lý
* Đầu ra là gì

Dữ liệu inputs khác nhau, hệ thống xử lý khác nhau và đầu ra cũng sẽ khác nhau 🡺 cần 1 kỹ thuật để giảm thiểu số lượng test cases cần phải thực hiện 🡪 phân lớp tương đương là 1 kỹ thuật như vậy, một lớp / vùng tương đương nghĩa là hệ thống sẽ xử lý giống nhau cho tập đầu vào và đưa ra kết quả giống nhau

Các gợi ý có thểáp dụng khi chia các lớp tương đương:

* Dựa trên giá trị hợp lệ hoặc không hợp lệ của **đầu vào**
* Dựa trên sự giống nhau hoặc khác nhau của giá trị **đầu vào**
* Dựa trên sự giống nhau hoặc khác nhau của giá trị **đầu ra**
* Dựa trên sự giống nhau hoặc khác nhau của quy trình xử lý





**Kỹ thuật kiểm thử: phân tích điểm biên**

Là kỹ thuật thiết kế các test cases để kiểm thử các giá trị biên giữa các vùng, các lớp tương đương. Nói 1 cách khác, thay vì chọn 1 phần tử bất kỳ của 1 lớp để kiểm thử, chúng ta sẽ chọn phần tử bé nhất và lớn nhất của lớp đó để thực hiện kiểm thử.

Không phải lúc nào chúng ta cũng có thể xác định được phần tử min và max của lớp tương đương, ví dụ tập hợp các hình trong toán học, chúng ta có thể chia thành các lớp tương đương như: hình chữ nhật, tam giác, tròn … tuy nhiên trong tập hợp các hình đó, chúng ta không thể xác định được hình nào lớn nhất và hình nào bé nhất

Có 2 cách tiếp cận phân tích giá trị biên:

1. Mô hình 2 giá trị cho mỗi biên: theo mô hình này, biên sẽ nằm giữa giá trị lớn nhất của 1 lớp tương đương va giá trị nhỏ nhất của lớp tương đương ngay trên nó. Ví dụ biên của 3 lớp là 0 & 100 (0 là biên của lớp 1 và 2, 100 là biên của lớp 2 và 3), thì chúng ta sẽ có 4 tình huống cần kiểm thử là 0, 1, 99, 100
2. Mô hình 3 giá trị cho mỗi biên: nếu là giới hạn dưới của 1 lớp thì chúng ta sẽ chọn 3 giá trị: giá trị lớn nhất nằm ngoài lớp đó, giá trị biên, và giá trị ngay trên giá trị biên thuộc về lớp đó. Nếu là giới hạn trên của 1 lớp thì chúng ta sẽ chọn 3 giá trị: giá trị ngay dưới giá trị biên thuộc về lớp đó, giá trị biên và giá trị nhỏ nhất nằm ngoài lớp đó

BVA là kỹ thuật hỗ trợ cho kỹ thuật phân lớp tương đương, 2 kỹ thuật này luôn đi cùng với nhau, vì điểm xác định được các điểm biên chính là xác định được các lớp. Ngoài ra vì đã tìm được các lớp tương đương nên việc chọn 1 giá trị trong lớp đó để thực hiện kiểm thử là đúng nhưng chưa đủ, cần kiểm tra các điểm biên và cận biên nữa

**1 A / C / B**

**2 C / D**

**3 D**

**4 B**

**5 C**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 24: Kỹ thuật kiểm thử: bảng quyết định**

**Khái niệm và các thành phần của bảng quyết định**

Bảng quyết định:

* Là một kỹ thuật kiểm thử hộp đen
* Là một cách tốt để thể hiện sự kết hợp của các đầu vào
* Còn được gọi là bảng nguyên nhân – kết quả (vì trên bảng đó thể hiện rõ các đầu vào và kết quả sau khi các đầu vào được kết hợp với nhau thế nào)

Các thành phân của một bảng quyết định:

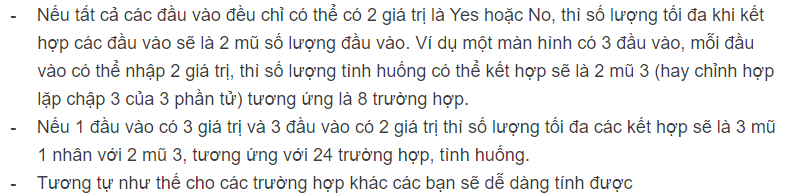
1. Các hàng điều kiện – condition rowss (hay còn gọi là causes): liệt kê tất cả những điều kiện hay đầu vào liên quan đến 1 quyết định
2. Các giá trị - values: là các giá trị tương ứng với các điều kiện, các đầu vào.
3. Các hàng hành động – action rows (hay còn gọi là Effects): liệt kê những hành động hay phản hồi của hệ thống
4. Các luật – Rules (hay còn gọi là combinations): xác định những hành động nào đi kèm với 1 tập hợp các điều kiện cho trước



**Các bước xây dựng bảng quyết định**

Để xây dựng 1 bảng quyết định, chúng ta có các bước sau:

1. Xác định tất cả các điều kiện và giá trị: ở bước này chúng ta sẽ tìm tất cả các điều kiện và các giá trị tương ứng với các điều kiện đó
2. Xác định các hành động, phản ứng hay phản hồi mà hệ thống có thể có, nói cách khác là xác định tất cả các quyết định có thể có của hệ thống
3. Tính số lượng tối đa các kết hợp: nhân **số lượng các giá trị mà mỗi điều kiện có với nhau.**



1. Điền các giá trị của các điều kiện vào từng cột theo thứ tự
2. Xác định các hành động cho từng sự kết hợp của các đầu vào: với mỗi action đánh dấu hành động hay kết quả trả về tương ứng bằng thêm dấu **X** vào bảng quyết định
3. Rà soát lại toàn bộ bảng quyết định
4. Đơn giản hóa bảng quyết định bằng cách giảm bớt những kết hợp có hành động hay kết quả trả về giống nhau

Để thực hiện đơn giản hóa bảng quyết định, có những gợi ý sau:

1. Những cột có action như nhau thì có thể kết hợp với nhau
2. Những cột có thể kết hợp lại với nhau là những cột thường đứng cạnh nhau
3. Tại 1 thời điểm chỉ nên thực hiện 1 sự kết hợp để tránh nhầm lẫn, sau đó lặp lại việc kết hợp trên bảng quyết định đã được đơn giản hóa 1 phần cho đến khi không thể đơn giản hóa được bảng đấy nữa

**Thiết kế các test cases dựa trên bảng quyết định**

Nguyên tắc: mỗi cột – rule ở BQĐ cuối cùng sẽ là 1 test cases tương ứng. Các điều kiện sẽ là các input của test ases đó, và action sẽ là expected output của test case đó

Kỹ thuật BQĐ có thể kết hợp với kỹ thuật lớp tương đương và phân tích điểm biên hoặc với chính 1 BQĐ khác để tạo ra nhiều test cases hơn

**1 B**

**2 A**

**3 C**

**4 A**

**5 D**

**--------------- --------------- --------------- --------------- --------------- ---------------**

**Bài 25: Kỹ thuật kiểm thử: dịch chuyển trạng thái**

**Kỹ thuật kiểm thử dịch chuyển trạng thái là gì?**

* Là một trong các kỹ thuật thiết kế kiểm thử hộp đen: các tình huống kiểm thử được thiết kế để kiểm tra các state được dịch chuyển trong hệ thống phần mềm là hợp lý hay không hợp lý
* Đây là kỹ thuật lý tưởng khi chúng ta có 1 chuỗi các sự kiện xảy ra và các điều kiện áp dụng trên các sự kiện đó, và cách xử lý phù hợp cho 1 sự kiện hay điều kiện cụ thể phụ thuộc vào các sự kiện và các điều kiện đã xảy ra trong quá khứ
* Trong 1 số tình huống, chuỗi các sự kiện có thể là vô hạn và vượt qua khả năng kiểm thử của chúng ta. Nhưng chúng ta vẫn muốn có 1 kỹ thuật cho phép chúng ta kiểm soát tùy ý chuỗi dài các sự kiện.

**Biểu đồ dịch chuyển trạng thái – State transition diagram**

Biểu đồ dịch chuyển state sẽ có 4 thành phần:

* Các trạng thái – state: biểu diện trạng thái của hệ thống, có thể là hình các nút hoặc các vòng tròn
* Các dịch chuyển – transitions: là sự chuyển trạng thái của hệ thống, các mũi tên sẽ thể hiện sự dịch chuyển hợp lệ
* Các sự kiện – events: nguyên nhân gây ra sự dịch chuyển
* Các hành động – actions: là kết quả của 1 sự dịch chuyển (báo lỗi …)

Để có thể bao quát được hết các trạng thái và sự dịch chuyển, các test cases sẽ được thiết kế theo 4 bước sau đây:

1. Thông qua 1 rule để xác định nơi khởi đầu của 1 test case và xác định nơi mà test case đó **có thể** kết thúc hoặc **phải** kết thúc.
2. Từ 1 init state, xác định chuỗi các event hay condition mà sẽ đưa đến được final state. Với mỗi sự dịch chuyển xảy ra, lưu lại action mong muốn mà hệ thống nên làm. Đấy chính là expected output của mỗi test case
3. Đánh dấu các state và sự dịch chuyển đã được duyệt qua để tránh lặp lại.
4. Lặp lại bước 2 và 3 cho đến khi tất cả các state đều đã được ghé thăm và các dịch chuyển đã được duyệt. hay nói cách khác, tất cả các state và dịch chuyển đều đã được đánh dấu

**Bảng dịch chuyển trạng thái**

Bảng dịch chuyển trạng thái được xây dựng theo các bước sau:

1. List ra tất cả các states có được từ mô hình dịch chuyển state
2. List tất cả các cặp events, conditions thể hiện ở mô hình dịch chuyển trạng thái
3. Tạo 1 bảng có 4 cột: current state, event / condition, action, new state với các hàng là sự kết hợp của mỗi state với tất cả các cặp event / condition. Hay nói cách khác chúng ta multiply số state với số cặp / điều kiện thì ra được số lượng hàng của bảng
4. Điền action và state tương ứng với mỗi kết hợp của state với event / condition

**1 C**

**2 A**

**3 C**

**4 A**

**5 D**