

# Kiểm thử phần mềm

## Chương 1 - Tổng quan



KHOA CÔNG NGHỆ THÔNG TIN  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

# Nội dung

- ☐ **Tại sao kiểm thử quan trọng?**
- ☐ Kiểm thử phần mềm là gì?
- ☐ Quy trình kiểm thử phần mềm
- ☐ Các nguyên lý tổng quát
- ☐ Vai trò và thái độ
- ☐ Các phân loại liên quan

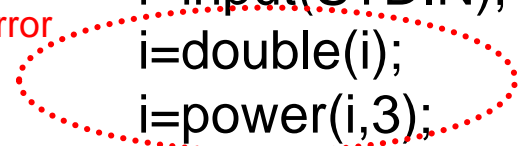
# Lỗi phần mềm



- ☐ Hành vi con người  
→ Error (Mistake)
- ☐ Hệ quả xuất hiện trên chương trình  
→ Fault/Defect
- ☐ Khi thực thi chương trình  
→ Failure
- ☐ Hệ quả không như mong đợi  
→ Incident

# Ví dụ

- **Error** là thuật ngữ của Developer.
- **Bug** là thuật ngữ của Tester

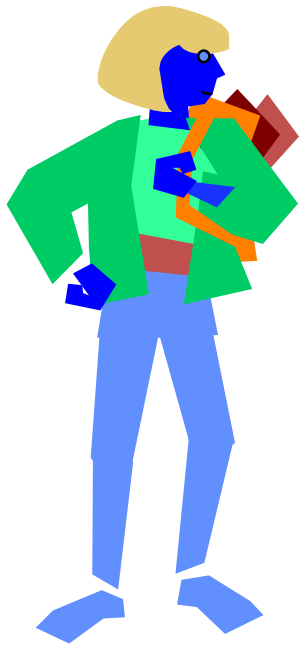
requirements:  
for input  $i$ ,  
give output  $2*(i^3)$   
(so 6 yields 432)

software:  
error  
  
`i=input(STDIN);  
i=double(i);  
i=power(i,3);  
output(STDOUT,i);`

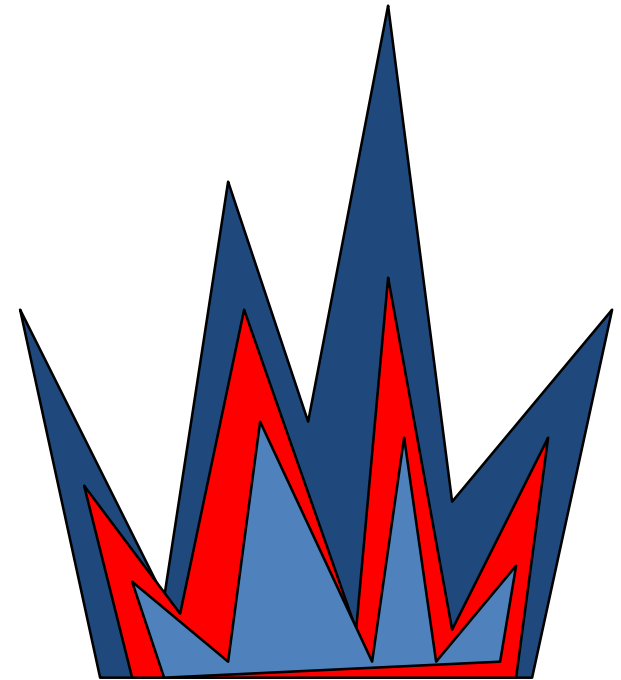
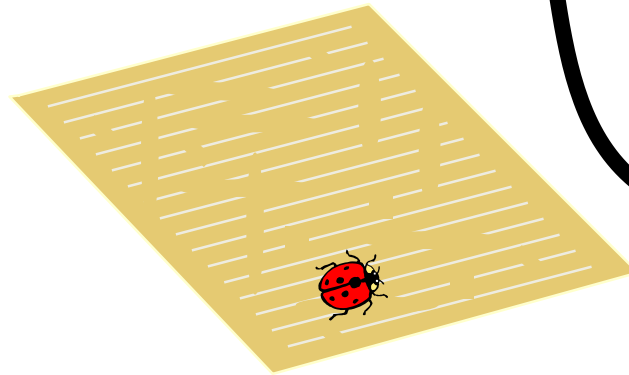
output (verbose):  
input: 6  
fault  
  
doubling input..  
computing power..  
  
output: 1728  
fault + failure

# Lỗi phần mềm

A person makes  
an **error** ...



... that creates a  
**fault** in the  
software ...



... that can cause  
a **failure**  
in operation

# Độ tin cậy (Reliability)

- ☐ Là xác suất phần mềm không phát sinh lỗi trong thời gian và điều kiện xác định
- ☐ Thảo luận
  - ☐ Liệu có một hệ thống không tồn tại lỗi?
  - ☐ Liệu một hệ thống đáng tin cậy nhưng vẫn tồn tại lỗi?
  - ☐ Liệu một hệ thống không lỗi là đáng tin cậy?

# Tại sao có lỗi?

- Phần mềm viết bởi con người
  - Biết nhiều thứ, nhưng không phải mọi thứ
  - Có kỹ năng, nhưng không hoàn hảo
  - Luôn phạm sai lầm
- Làm việc dưới điều kiện căng thẳng để kịp bàn giao đúng tiến độ
  - Không có thời gian kiểm tra, giả định bị sai
  - Hệ thống chưa hoàn chỉnh

# Chi phí lỗi

- Có thể rất lớn
  - ▣ Ariane 5: 7 tỉ USD
  - ▣ Mariner space probe to Venus: 250 triệu USD
  - ▣ American Airlines: 50 triệu USD
- Có thể gây chết người
  - ▣ Therac-25
  - ▣ Airbus & Korean Airlines



# Chi phí lỗi

- Có thể không đáng kể
  - Bất tiện khi sử dụng
  - Ảnh hưởng không nhìn thấy được
- Không tuyến tính: 1 lỗi nhỏ nhưng có hậu quả lớn

# Tại sao kiểm thử cần thiết?

- Không vì:
  - Lắp khoản thời gian giữa ngày hoàn thành và ngày bàn giao
  - Chứng minh là phần mềm không lỗi
  - Kiểm thử là một phần của kế hoạch dự án

# Tại sao kiểm thử cần thiết?

## □ Vì

- Phần mềm luôn tồn tại lỗi
- Đánh giá độ tin cậy
- Chi phí lỗi có thể rất cao
- Tránh bị kiện từ khách hàng
- Giữ uy tín trong kinh doanh

# Nội dung

- ☐ Tại sao kiểm thử quan trọng?
- ☐ **Kiểm thử phần mềm là gì?**
- ☐ Quy trình kiểm thử phần mềm
- ☐ Các nguyên lý tổng quát
- ☐ Vai trò và thái độ

# Kiểm thử phần mềm là gì?

- Kiểm thử phần mềm là quá trình thực thi một chương trình với mục đích tìm lỗi

## *The Art of Software Testing*

- Là quá trình kiểm tra xem phần mềm có chạy chính xác hay không (Verification) và có thoả mãn yêu cầu của khách hàng hay không (Validation) nhằm hướng tới mục tiêu Chất lượng cho phần mềm.

# Xác minh và thẩm định

- ☐ Xác minh (Verification)
  - ☐ Có đúng đặc tả, có đúng thiết kế
  - ☐ Phát hiện lỗi lập trình
- ☐ Thẩm định (Validation)
  - ☐ Có đáp ứng nhu cầu người dùng
  - ☐ Phát hiện lỗi phân tích, thiết kế

# Xác minh và thẩm định

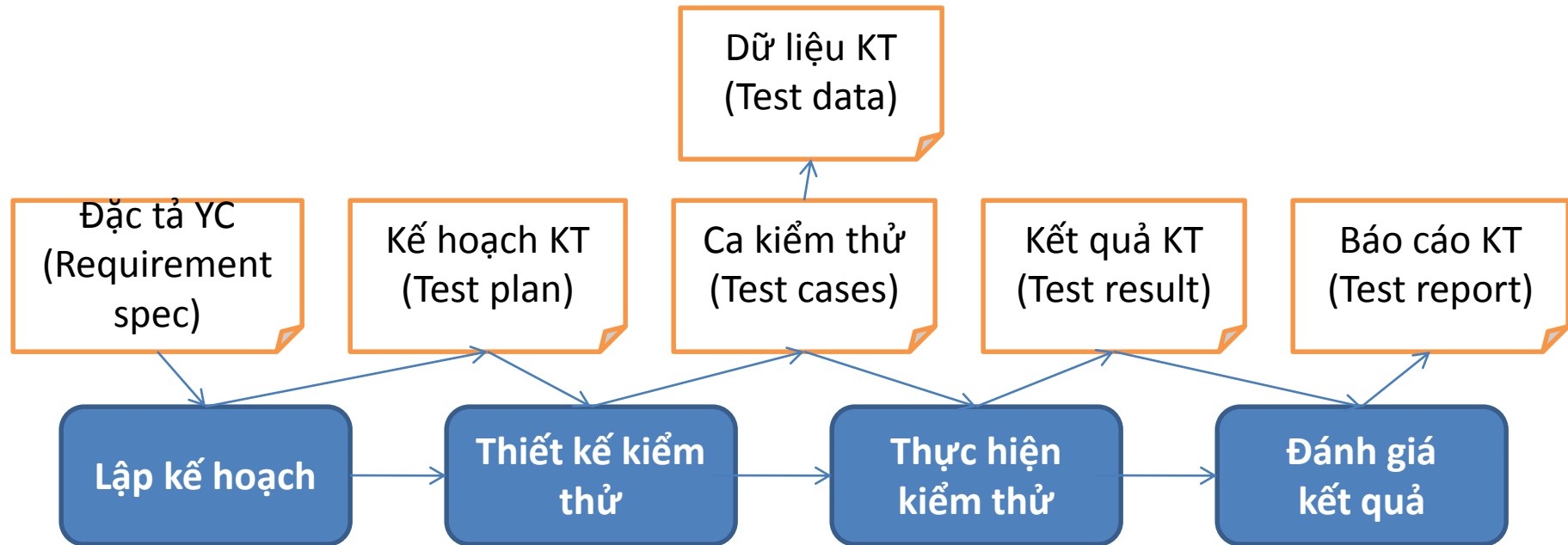
- V & V = Verification and Validation
  - Mục tiêu là phát hiện và sửa lỗi phần mềm, đánh giá tính đúng đắn được của phần mềm
  - Thứ tự thực hiện: Verification → Validation
  - Verification chiếm 80%, Validation chiếm 20% công việc
- Validation tác động 80% hiệu quả chung

# Nội dung

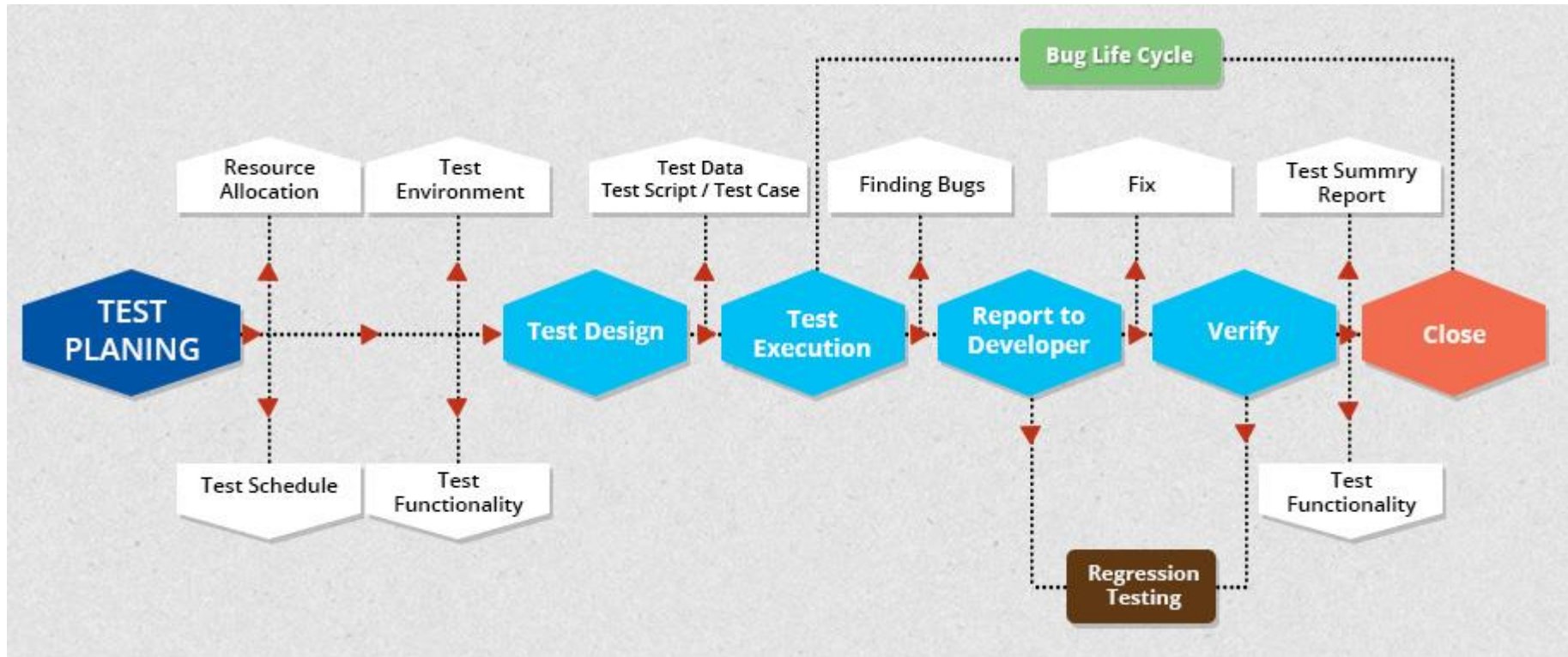
- ☐ Tại sao kiểm thử quan trọng?
- ☐ Kiểm thử phần mềm là gì?
- ☐ **Quy trình kiểm thử phần mềm**
- ☐ Các nguyên lý tổng quát
- ☐ Vai trò và thái độ



# Quy trình kiểm thử đơn giản



# Quy trình kiểm thử chi tiết



# Lập kế hoạch

- ☐ Mục đích: chỉ định, mô tả các chiến lược kiểm thử
- ☐ Kết quả: bản kế hoạch kiểm thử (Test plan)
- ☐ Nội dung:
  - ☐ Giới thiệu
  - ☐ Yêu cầu
  - ☐ Chiến lược
  - ☐ Thời gian
  - ☐ Tài nguyên

# Lập kế hoạch

Test  
Policy

Test  
Strategy

Company level

High Level  
Test Plan

Project level (IEEE 829)  
(one for each project)

Detailed  
Test Plan

Test stage level (IEEE 829)  
(one for each stage within a project,  
e.g. Component, System, etc.)

# Lập kế hoạch

- Các bước lập kế hoạch
  - Xác định yêu cầu kiểm thử
  - Khảo sát rủi ro
  - Xác định chiến lược kiểm thử
  - Xác định nhân lực, thiết bị
  - Lập kế hoạch chi tiết
  - Tổng hợp và tạo các bản kế hoạch kiểm tra
  - Xem xét các kế hoạch kiểm tra

# Thiết kế

- Mục đích: bảo đảm tất cả các tình huống kiểm tra “quét” hết tất cả yêu cầu cần kiểm tra
- Kết quả: ca kiểm thử (Test cases), dữ liệu kiểm thử (Test data)

# Thiết kế

- Các bước thiết kế ca kiểm thử
  - Xác định điều kiện cần thiết lập, mô tả dữ liệu đầu vào, kết quả mong chờ
  - Mô tả các bước chi tiết
  - Xem xét và khảo sát độ bao phủ
  - Xem xét test cases và các bước kiểm tra

# Ca kiểm thử (test case)

- ☐ Test case: dữ liệu để kiểm tra hoạt động của chương trình
- ☐ Test case tốt: được thiết kế để phát hiện một lỗi của chương trình
- ☐ Kiểm thử thành công: phát hiện ra lỗi
- ☐ Mục đích
  - ☐ Chứng minh sự tồn tại của lỗi
  - ☐ Không chứng minh sự không có lỗi



# Nội dung của test case

- Mô tả
  - Chức năng muốn kiểm thử
  - Dữ liệu đầu vào
  - Môi trường thử nghiệm
  - Thứ tự thao tác
- Kết quả mong muốn
  - Dữ liệu đầu ra
  - Màn hình, thời gian phản hồi
- Kết quả thực tế

# Thực hiện

- ☐ Mục đích: thực hiện các ca kiểm thử, ghi nhận kết quả
- ☐ Kết quả: bảng báo cáo (Test result)

# Thực hiện

- Các bước thực hiện kiểm thử
  - Xác lập và khởi động môi trường
  - Thực hiện các bước (bằng tay hoặc script)
  - Đánh giá quá trình kiểm thử
    - Hoàn tất chu kỳ → Thẩm định kết quả
    - Bị dừng hoặc treo → xác định nguyên nhân lỗi, khắc phục và lập lại
  - Thẩm định kết quả: bảo đảm kết quả nhận được là đáng tin cậy

# Đánh giá quá trình kiểm thử

- Mục đích: xem xét và đánh giá kết quả kiểm tra, liệt kê lỗi, chỉ định các yêu cầu thay đổi, thống kê số liệu
- Kết quả: báo cáo kiểm thử (Test report)
- Lưu ý: bước đánh giá mang tính toàn cục

# Nội dung

- ☐ Tại sao kiểm thử quan trọng?
- ☐ Kiểm thử phần mềm là gì?
- ☐ Quy trình kiểm thử phần mềm
- ☐ **Các nguyên lý tổng quát**
- ☐ Vai trò và thái độ

# Các nguyên lý tổng quát

## 1. Phơi bày sự hiện diện của lỗi

- ❑ Cho thấy lỗi đang tồn tại
- ❑ Giảm xác suất lỗi chưa phát hiện

## 2. Không thể vét cạn hết các trường hợp

- ❑ Không thể kiểm nghiệm triệt để một phần mềm
- ❑ Thay vào đó:  
Phân tích rủi ro  
Độ ưu tiên

# Các nguyên lý tổng quát

## 3. Kiểm tra sớm

- Nên bắt đầu sớm nhất có thể trong chu kỳ phát triển

## 4. Gom nhóm lỗi

- Nguyên lý Pareto: 20% module gây ra 80% lỗi
- → cô lập, tập trung những module khả nghi nhất

# Các nguyên lý tổng quát

## 5. Nghịch lý thuốc trừ sâu (Pesticide paradox)

- ❑ Sử dụng cùng 1 kỹ thuật, ca kiểm thử nhiều lần → không tìm được lỗi mới
- ❑ Ca kiểm thử phải được xem xét và thay đổi thường xuyên

## 6. Phụ thuộc ngữ cảnh

- ❑ Thực hiện khác nhau trong các ngữ cảnh khác nhau



# Các nguyên lý tổng quát

## 7. Ảo tưởng “không lỗi” (Absence-of-errors fallacy)

- ❑ Việc tìm và sửa chữa lỗi sẽ vô nghĩa nếu hệ thống được xây dựng xong vô dụng

# Các nguyên tắc cơ bản

- ☐ Test case phải luôn có kết quả mong đợi
- ☐ Lập trình viên không nên kiểm tra chương trình của chính mình
- ☐ Phải xem xét kỹ lưỡng kết quả của mỗi trường hợp kiểm tra
- ☐ Phải luôn có trường hợp kiểm tra cho các trường hợp ngoại lệ, những trường hợp ít ai ngờ tới
- ☐ Không nên bỏ test case sau khi kiểm tra
- ☐ Đừng bao giờ giả định phần mềm không có lỗi
- ☐ Xác suất còn lỗi trong phần mã nguồn chưa kiểm tra tương ứng với tỷ lệ đã được kiểm tra

# Nội dung

- ☐ Tại sao kiểm thử quan trọng?
- ☐ Kiểm thử phần mềm là gì?
- ☐ Quy trình kiểm thử phần mềm
- ☐ Các nguyên lý tổng quát
- ☐ **Vai trò và thái độ**

# Vai trò QA, QC

## ☐ QC – Quality Control

- ☐ Những hoạt động, những kỹ thuật nhằm đảm bảo chất lượng sản phẩm.

## ☐ QA – Quality Assurance

- ☐ Những kế hoạch, những hoạt động mang tính hệ thống nhằm đảm bảo quá trình sản xuất sẽ tạo ra những sản phẩm có chất lượng.

*Định nghĩa của ISO 9000*

# Vai trò QA, QC

## ☐ QC

- ☐ Sản phẩm
- ☐ Phản ứng
- ☐ Tìm lỗi

## ☐ Ví dụ

- ☐ Kiểm duyệt
- ☐ Kiểm thử
- ☐ Thanh tra
- ☐ Kiểm tra lại

## ☐ QA

- ☐ Tiến trình
- ☐ Tiên đoán, ước tính
- ☐ Ngăn ngừa lỗi

## ☐ Ví dụ

- ☐ Đảm bảo chất lượng
- ☐ Định nghĩa tiến trình
- ☐ Chọn lựa công cụ
- ☐ Huấn luyện

# Thái độ của Tester

## ☐ Cẩn thận (Cautious)

- ☐ Phỏng đoán chứ không kết luận
  - ☐ Tập thừa nhận “Tôi không biết”
  - ☐ Có người khác kiểm tra lại
- Good testers are hard to fool.

## ☐ Tò mò (Curious)

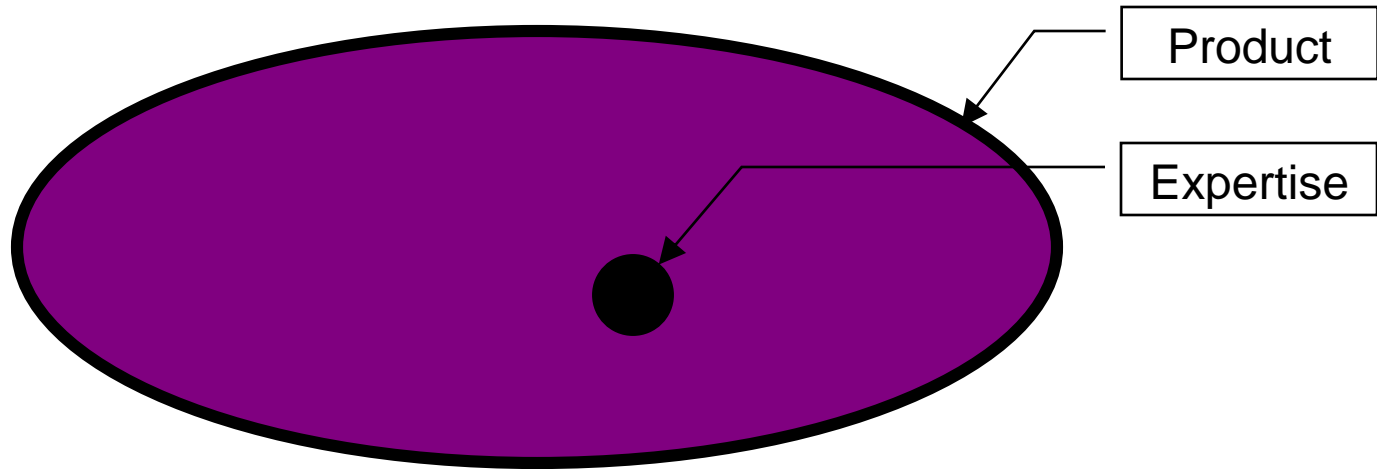
- ☐ Điều gì xảy ra nếu ...?
- ☐ Nó hoạt động như thế nào?
- ☐ Tại sao nó xảy ra?

# Thái độ của Tester

- ☐ Chỉ trích, phê phán (Critical)
  - ☐ Tiến hành phỏng đoán và bác bỏ
  - ☐ Tích cực tìm kiếm phản chứng
- ☐ Can đảm (Courageous)

→ Good testers are hard to fool.

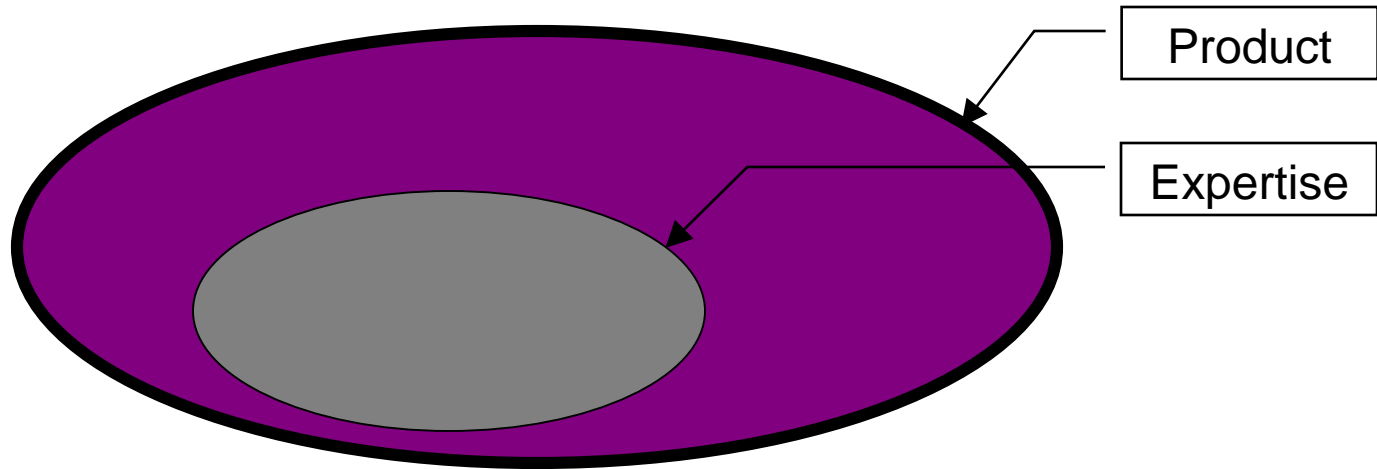
# Developer's Mindset



- ☐ A good developer is clever
- ☐ Emphasis on depth of expertise



# Tester's Mindset



- ☐ A good tester is mischievous
- ☐ Emphasis on breadth of expertise

# Các phân loại liên quan

- ☐ Các cấp độ kiểm thử
- ☐ Các phương pháp kiểm thử
- ☐ Các loại/kỹ thuật kiểm thử
- ☐ Các tài liệu kiểm thử
- ☐ Các công cụ kiểm thử
- ☐ Các bằng cấp liên quan đến kiểm thử

# Software Testing Levels (cấp độ)

- ☐ Unit Testing
- ☐ Integration Testing
- ☐ System Testing
- ☐ Acceptance Testing
- ☐ Other Testing Levels

# Software Testing Methods (phương pháp)

- ☐ Black Box Testing
- ☐ White Box Testing
- ☐ Gray Box Testing
- ☐ Manual Testing
- ☐ Automated Testing
- ☐ Other Testing Methods



# Software Testing Types (loại/kỹ thuật)

- ☐ Desktop Applications Testing
- ☐ Web Applications Testing
- ☐ Performance Testing
- ☐ Smoke Testing
- ☐ Installation Testing
- ☐ Look & Feel Testing
- ☐ Usability Testing
- ☐ Regression Testing
- ☐ Exploratory Testing
- ☐ Other Testing Types

# Software Testing Artifacts (tài liệu)

- ☐ Test Plan
- ☐ Test Cases
- ☐ Bug Reports
- ☐ Test Reports
- ☐ Other Testing Artifacts

# Software Testing Tools (công cụ)

- ☐ Test Management Tools
- ☐ Functional Test Automation Tools
- ☐ Performance Test Automation Tools
- ☐ Bug Tracking Tools
- ☐ Other Testing Tools

# Một số công cụ thường dùng

- ❑ **Automated Functional Testing:** Selenium IDE/RC, Rational Robot, Functional Tester, OTP, OC, FitNesse, TestComplete, SilkTest
- ❑ **Automated performance testing:** LoadRunner, Rational Robot, ATS
- ❑ **Secure Testing:** Web Vulnerability Scanner, WebInspect, Rational AppScan, Tenable Network Security, GUI Zenmap, Web Application Security, Paros, Wikto, Grendel-Scan, MaxPatrol
- ❑ **Accessibility Testing:** OTP, Watchfire Bobby
- ❑ **Issue Management Tools:** Atlassian, Bugzilla, Edgewall, VersionOne, Mantis, FogBugz





# Software Testing Certifications (chứng chỉ của công ty)

- ☐ Vendors who offer Testing Certifications:
  - ☐ Mercury
  - ☐ Segue
  - ☐ Rational
  - ☐ Empirix



# Software Testing Certifications (chứng chỉ phổ dụng)

- Vendor-Neutral Testing Certifications:
  - ▣ Certified Software Quality Analyst (CSQA)
  - ▣ Certified Software Test Engineer (CSTE)
  - ▣ ISTQB Certified Tester, Foundation Level (CTFL)
  - ▣ Quality Improvement Associate Certification (CQIA)
  - ▣ Certified Test Manager (CTM)
  - ▣ Certified Software Test Professional (CSTP)
  - ▣ Six Sigma Black Belt Certification (SSBB).

# Thảo luận

□ <http://www.c-sharpcorner.com/UploadFile/51e7af/basics-of-manual-testing/>

