

# Đảm bảo chất lượng phần mềm

# Software Quality Assurance

Đỗ Thị Bích Ngọc

PTIT/FIT/SE

[dothibichngoc@gmail.com](mailto:dothibichngoc@gmail.com)

# Giới thiệu chung

# Giới thiệu



Who are you?

- Đã học rất nhiều môn IT ở PTIT: Nhập môn công nghệ phần mềm, Lập trình hướng đối tượng, Cấu trúc dữ liệu và giải thuật, Phân tích và thiết kế HTTT, Lập trình web...?

What is your plan/dream?

- Thành developer, tester, PM, BA, sale,... ?

Who am I?

- Tiến sĩ về Kiểm chứng phần mềm
- Dạy Đảm bảo chất lượng phần mềm > 5 năm.
- Hoạt động và kinh nghiệm liên quan: kiểm thử thủ công, kiểm thử tự động, rà soát, trưởng nhóm quản lý SQA, nghiên cứu và phát triển tool hỗ trợ kiểm thử tự động

# Vì sao phải học SQA?



PTIT và giáo viên mong muốn gì ở các bạn?

- Phần mềm của các bạn chuyên nghiệp hơn: dễ sử dụng hơn, đáng tin cậy hơn, hiệu năng cao hơn, chính xác hơn... nhưng ít lỗi hơn!

Bạn biết QA, Tester sẽ làm gì với bạn?

- Họ... có phải là kẻ thù của Lập trình viên, người thiết kế, PM?
- Họ sẽ bắt lỗi chương trình của bạn, rà soát thiết kế của bạn...?

*Không hẳn thế!*

Có kiến thức cơ bản để trở thành QA/tester.

# Nghề kiểm thử - Kỹ năng



1. Đọc, hiểu, phân tích Requirement
2. Ngoại ngữ tốt

Requirement study

Test Plan

Test Design/ Test Matrix/ Test case

Execute Test & Bug log

Re-test and Test Report

Test analysis and Close

Kỹ năng:  
- viết tài liệu  
- lập kế hoạch, khả năng bao quát vấn đề  
- Logic, chặt chẽ

IT Background

- Kỹ thuật test
- Phương pháp tạo TC
- Sử dụng Tool test

1. Cẩn thận, tỉ mỉ
2. Rõ ràng, chi tiết
3. Ngắn gọn, dễ hiểu
4. Trung thực

# Nghề kiểm thử - Cấp độ nghề nghiệp



- **Người kiểm thử chưa có kinh nghiệm**
  - sung sướng với số lỗi mà họ tìm ra
- **Người kiểm thử có kinh nghiệm**
  - cũng tìm ra lỗi nhưng làm việc với người phát triển để chắc rằng chúng được sửa
- **Chuyên gia kiểm thử** làm nhiều hơn điều đó:
  - Họ làm việc với **người phát triển** để chắc họ không tạo ra lỗi
  - Họ làm việc với **người dùng** để xác định yêu cầu tốt hơn
  - Họ làm việc với **chuyên gia an ninh** để chắc phần mềm được kiểm thử về an ninh
  - Họ làm việc với **người quản lý dự án** để nhận diện và giảm nhẹ rủi ro;
  - Họ làm việc với **người đảm bảo chất lượng** về sự tuân thủ

# Nghề kiểm thử - Vị trí nghề nghiệp



- Junior Tester
  - Senior Tester:
    - Test Leader
    - Test Manager
    - Expert tester
- 

Chứng chỉ nghề nghiệp:

- ISTQB
- CSTE
- ...

# Nghề kiểm thử - Định hướng



## ● Hướng chuyên gia

- Expert testers: Test tools....
- Tham gia training, làm seminar, workshop...
- Viết sách...

## ● Hướng quản lý

- Director
- PM
- Test Manager/ Test Leader...

## ● Hướng khác:

- BA, Sales, QA, BrSE, Dev...

# Với môn học này, bạn cần...

- Tham gia FB group để lấy:  
Sách, slide, lịch học  
Hướng dẫn cho bài tập lớn  
Các thông tin khác
- Tham gia lớp học (10% điểm )
- Làm bài tập lớn (20% điểm )
- Làm bài tập
- Kiểm tra giữa kì (20% điểm )
- Kiểm tra cuối kì (50% điểm)

# Yêu cầu

Tham gia >80% buổi học

Kiểm tra giữa kì >0

Bài tập lớn >0

Không copy ( điểm sẽ = 0)

## Bài 1

# Vì sao cần đảm bảo chất lượng phần mềm?

# Kiểm thử trong Thế kỉ 21st



Thị trường phần mềm hiện nay:

Lớn hơn

Cạnh tranh hơn

Nhiều người dùng hơn

*Kiểm thử đảm bảo sự thành công  
của sản phẩm phần mềm!*

Các ứng dụng điều khiển nhúng

Máy bay, điều phối trên không

Tàu vũ trụ

- PDAs
- DVD players
- mở cửa gara tự động
- cell phones

Đồng hồ

Lò vi sóng

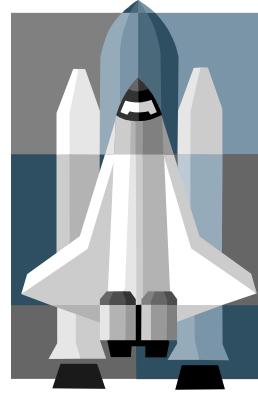
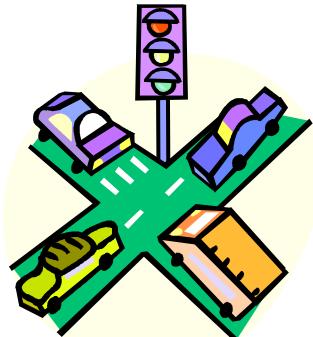
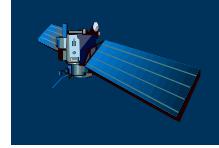
Điều khiển từ xa

Quy trình Agile đặt nhiều áp lực lên testers:

Programmers phải thực hiện unit test – mà không được đào tạo trước!

Kiểm thử là chìa khoá cho yêu cầu về chức năng – nhưng ai thực hiện?

# Phần mềm ở xung quanh chúng ta!



Quote due to Dr. Mark Harman

## Lỗi phần mềm - Software Error

Là các phần code sai do lỗi cú pháp, logic hoặc lỗi do phân tích, thiết kế.

## Sai sót - Software Fault

Là các errors dẫn tới hoạt động không chính xác của phần mềm.

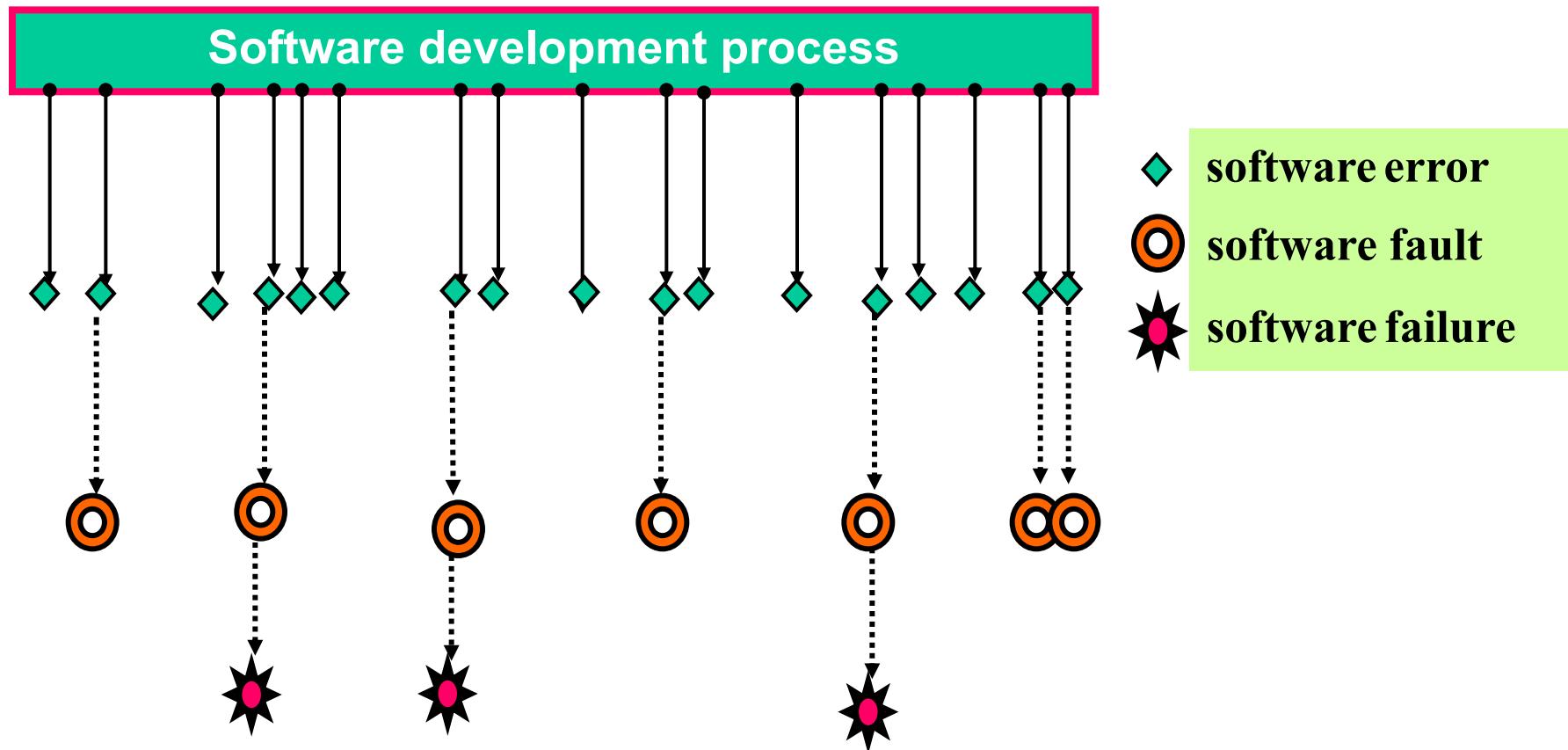
Không phải error nào cũng gây ra fault.

## Hỗng - Software Failures

Fault sẽ trở thành failure khi nó được kích hoạt

Một số đường chạy gây ra failures, một số không

# Software Faults, Errors & Failures



# Ví dụ về Fault và Failure



- Một bệnh nhân nói với bác sĩ các triệu chứng **Failures**
  - Bác sĩ cố gắng chẩn đoán nguyên nhân, bệnh **Fault**
  - Bác sĩ có thể tìm các trạng thái bất thường bên trong (huyết áp cao, nhịp tim không đều, vi khuẩn trong máu)
- Errors**

# Ví dụ

```

public static int numZero (int [ ] arr)
{ // Effects: If arr is null throw NullPointerException
// else return the number of occurrences of 0 in arr
int count = 0;
for (int i = 1; i < arr.length; i++)
{
    if (arr [ i ] == 0)
    {
        count++;
    }
}
return count;
}

```

Fault: phải tìm từ 0,  
thay vì 1

Test 1  
[ 2, 7, 0 ]  
Mong muốn: 1  
Thực tế: 1

Test 2  
[ 0, 2, 7 ]  
Mong muốn: 1  
Thực tế: 0

Error: i là 1, không  
phải 0, ở vòng lặp đầu  
Failure: không

Error: i là 1, không phải 0  
Error lan truyền tới biến count  
Failure: count là 0 ở câu lệnh trả về

# Từ khoá Bug



- Bug thường được dùng
- Đôi khi người nói có ý là fault, đôi khi là error, đôi khi là failure ... thường họ chả biết mình đang có ý nói về gì!
- Môn này sẽ cố gắng dùng từ với ý nghĩa chính xác, được định nghĩa , và không nhầm lẫn

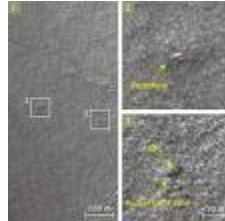


“It has been just so in all of my inventions. The first step is an intuition, and comes with a burst, then difficulties arise—this thing gives out and *[it is]* then that 'Bugs'—as such little faults and difficulties are called—show themselves and months of intense watching, study and labor are requisite. . .” – Thomas Edison

“an analyzing process must equally have been performed in order to furnish the Analytical Engine with the necessary operative data; and that herein may also lie a possible source of error. Granted that the actual mechanism is unerring in its processes, the cards may give it wrong orders.” – Ada, Countess Lovelace (notes on Babbage’s Analytical Engine)

# Software Failures

- NASA's Mars lander: Tháng 9 1999, bị lỗi do lỗi tích hợp units.



- THERAC-25 radiation machine : kiểm thử kém cho phần mềm yêu cầu an toàn cao, dẫn tới chết người : 3 bệnh nhân bị chết
- Ariane 5 explosion : Hàng triệu \$\$
- Tên lửa Patriot: 28 người chết

THERAC-25  
design



Ariane 5:  
exception-handling  
bug : tự nổ ở lượt  
phóng đầu (lỗi chuyển  
đổi 64-bit thành 16-  
bit: thiệt hại  
370 triệu \$)

**Chúng ta muốn phần mềm đáng tin cậy!  
Kiểm thử là một cách để đánh giá độ tin cậy!**

# Ví dụ về software failures

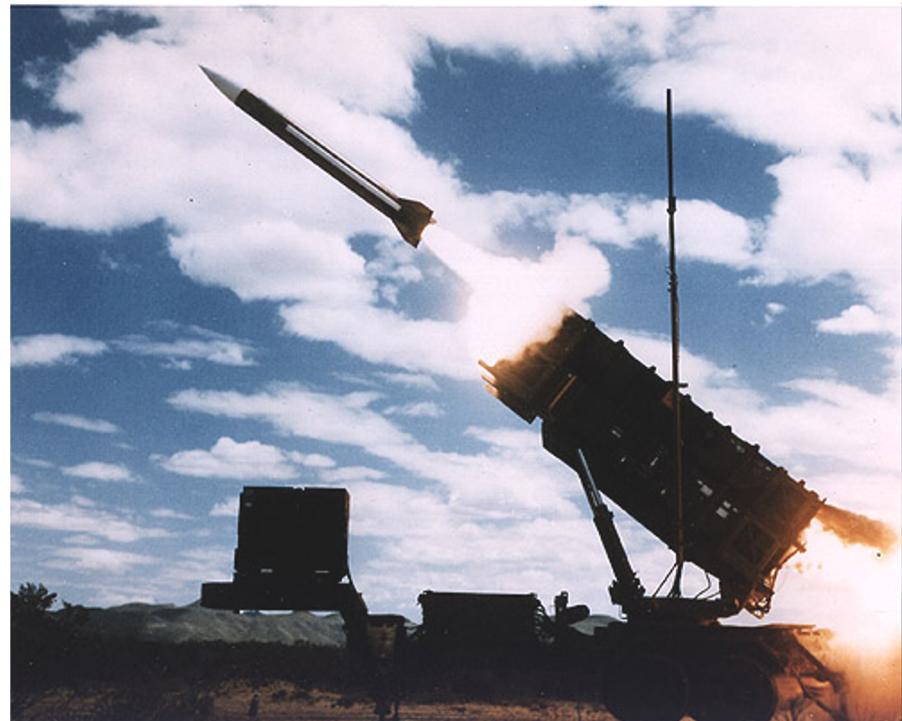
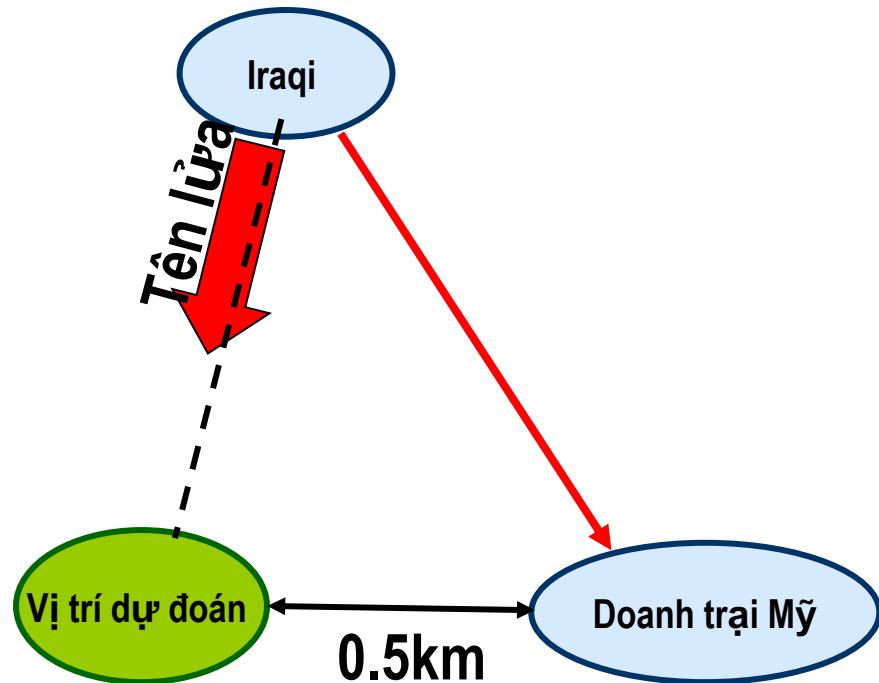


## Vụ thất bại tên lửa Patriot:

xảy ra tại Dharan, Saudi Arabia, vào 25 tháng 2, 1991

làm 28 người chết

nguyên nhân: vì lỗi rounding error (làm tròn)



# Ví dụ về software failures (tiếp)



## Vụ phát nổ Ariane 5:

xảy ra tại Kourou, French Guiana , vào ngày 4 tháng sáu 1996

thiệt hại 500 triệu \$

nguyên nhân: vì lỗi tràn số (**Overflow error**)

(vì biểu diễn số phẩy động (*floating point*) 64-bit  
bởi số phẩy tĩnh (*fixed point*) 16-bit)

The rocket  
exploded just 40  
seconds after its  
lift-off



# Chi phí Software Failures



- NIST báo cáo, “Ảnh hưởng kinh tế của việc không kiểm thử đầy đủ” (2002)
  - Riêng Mỹ từ \$22 đến \$59 tỉ đô mỗi năm
  - Kiểm thử tốt có thể giảm một nửa
- Thiệt hại lớn do ứng dụng web bị hỏng
  - Dịch vụ Tài chính : \$6.5 triệu mỗi giờ (riêng ở USA!)
- 2007 : Symantec nói hầu hết lỗ hỏng bảo mật là do phần mềm hỏng
- Question: Bạn thử tìm hiểu tình hình những năm gần đây?

# What Does This Mean?



Kiểm thử ngày càng quan trọng

Chúng ta làm gì khi kiểm thử?  
Mục tiêu là gì?

# Verification, Validation & Qualification



**Verification** (xác minh) – tiến trình đánh giá một system hoặc component xem sản phẩm của một pha phát triển đã cho có thỏa mãn điều kiện đưa ra ở đầu pha không

**Validation** (xác nhận) – tiến trình đánh giá một system hoặc component trong hoặc sau development process để xác định xem nó có thỏa mãn yêu cầu đã đặc tả hay không

**Qualification** – tiến trình đánh giá một system hoặc component có phù hợp cho operational use

# Verification, Validation & Qualification



Xác minh vs xác nhận:

- Xác minh: có tính kỹ thuật cao hơn, sử dụng những tri thức về các yêu cầu, đặc tả phần mềm.
- Xác nhận: phụ thuộc vào tri thức về lĩnh vực tương ứng. Ví dụ, xác nhận của phần mềm về máy bay yêu cầu tri thức từ kỹ sư hàng không và phi công.

IV & V - independent verification and validation- xác nhận và xác minh độc lập.

- yêu cầu thực hiện việc đánh giá bởi người không phải là lập trình viên, thường là chuyên gia trong lĩnh vực
- thực hiện khi khi dự án kết thúc

# Các mức kiểm thử



Mức 0: testing và debuging là giống nhau

Mức 1: Mục tiêu của kiểm thử là để chỉ ra phần mềm hoạt động

Mức 2: Mục tiêu của kiểm thử là để chỉ ra phần mềm không hoạt động

Mức 3: Mục tiêu của kiểm thử là để giảm các rủi ro khi sử dụng phần mềm

Mức 4: Nhằm trợ giúp các chuyên gia CNTT phát triển các phần mềm có chất lượng cao hơn.

# Mức 0

- Testing và debugging là một
- thường được thực hiện bởi các sinh viên trong các môn học lập trình. Sinh viên viết chương trình, chạy với vài đầu vào, và debug lỗi nếu có.
- không phân biệt giữa hành vi không đúng và lỗi bên trong chương trình.
- chỉ giúp ích đôi chút trong việc phát triển phần mềm chính xác

# Mức 1

- Nhằm để chứng minh tính đúng đắn.
  - Một cách phát triển tự nhiên từ mức 0
- Ta không thể chứng minh tính đúng đắn của phần mềm.
  - Giả sử ta chạy một tập test và không phát hiện ra lỗi nào. Vậy, phần mềm chạy tốt hay tập test kém?
- Việc kiểm thử không có giới hạn dừng cố định, cũng như không có một kỹ thuật test hình thức (formal) nào.
  - Nếu người quản lý hỏi: còn phải thực thi bao nhiêu test nữa? Ta không có cách nào trả lời chính xác câu hỏi này.

# Mức 2



- Nhắm để chỉ ra lỗi.
- Tìm lỗi là một mục tiêu mang tính tiêu cực.
  - Tester có thể vui vẻ khi tìm ra lỗi, nhưng developers thì không muốn vậy - họ muốn phần mềm chạy (mức 1 là suy nghĩ tự nhiên của developers).
- Đặt tester & developers vào quan hệ đối đầu.
  - Điều này có thể ảnh hưởng xấu tới cả nhóm.
- Nếu không tìm thấy lỗi nào thì sao?
  - Phần mềm chạy tốt? hoặc việc kiểm thử còn yếu?

# Mức 3

- Dựa trên nhận định: “Kiểm thử có thể chỉ ra lỗi khi nó xuất hiện, nhưng không thể chứng tỏ phần mềm không có lỗi”.
  - Nghĩa là, ta phải chấp nhận mỗi khi sử dụng phần mềm, ta có nguy cơ gặp lỗi.
- Toàn đội phát triển phần mềm có chung mục tiêu - giảm nguy cơ gặp lỗi khi sử dụng phần mềm.
- Cả tester và developer làm việc cùng nhau để giảm nguy cơ gặp lỗi.

# Mức 4



Khi tester và developers có chung mục tiêu (mức 3), tổ chức có thể chuyển sang mức 4. Kiểm thử nhằm mục tiêu tăng chất lượng.

Có nhiều cách để tăng chất lượng, trong đó tạo ra test có thể dẫn tới lỗi chỉ là một.

Kỹ sư kiểm thử có thể trở thành trưởng nhóm kỹ thuật của dự án. Họ có nhiệm vụ đánh giá, cải thiện chất lượng phần mềm, và sự thẩm định của họ sẽ trợ giúp cho developers.

# Mức 4



- Ví dụ như ta có 1 chương trình spell checker.
  - để tìm những từ sai chính tả (đánh vần sai),
  - để tăng khả năng viết chính tả: mỗi khi spell checker tìm ra một từ sai chính tả, ta có cơ hội để học cách viết đúng.
  - Do vậy, spell checker là “chuyên gia” về chất lượng viết chính tả.
- Tương tự, mức 4 hướng tới mục tiêu kiểm thử để tăng khả năng của developers trong việc phát triển các phần mềm chất lượng cao.
  - Testers có thể đào tạo developers.

# Where Are You?



Mức 0, I, hay 2 ?

Các công ty đang ở mức 0, I, hay 2 ?  
Hay 3?

Sau khoá học này, hi vọng bạn có tư duy ở  
mức 4!

# Why Each Test ?



**Nếu bạn không biết vì sao bạn tạo ra test, việc test sẽ không có ích lắm!**

- Mục tiêu và yêu cầu kiểm thử cần phải viết rõ
- Kế hoạch đạt mức phủ nào?
- Kiểm thử bao nhiêu là đủ?

# Why Each Test ?



*Nếu bạn không bắt đầu lập kế hoạch cho mỗi test khi yêu cầu chức năng được xây dựng, bạn sẽ không bao giờ biết vì sao mình dựng test như vậy?*

Yêu cầu ngưỡng tin cậy là gì?

Mỗi test đang muốn xác minh cái gì?

Đội xây dựng đặc tả cần người kiểm thử!

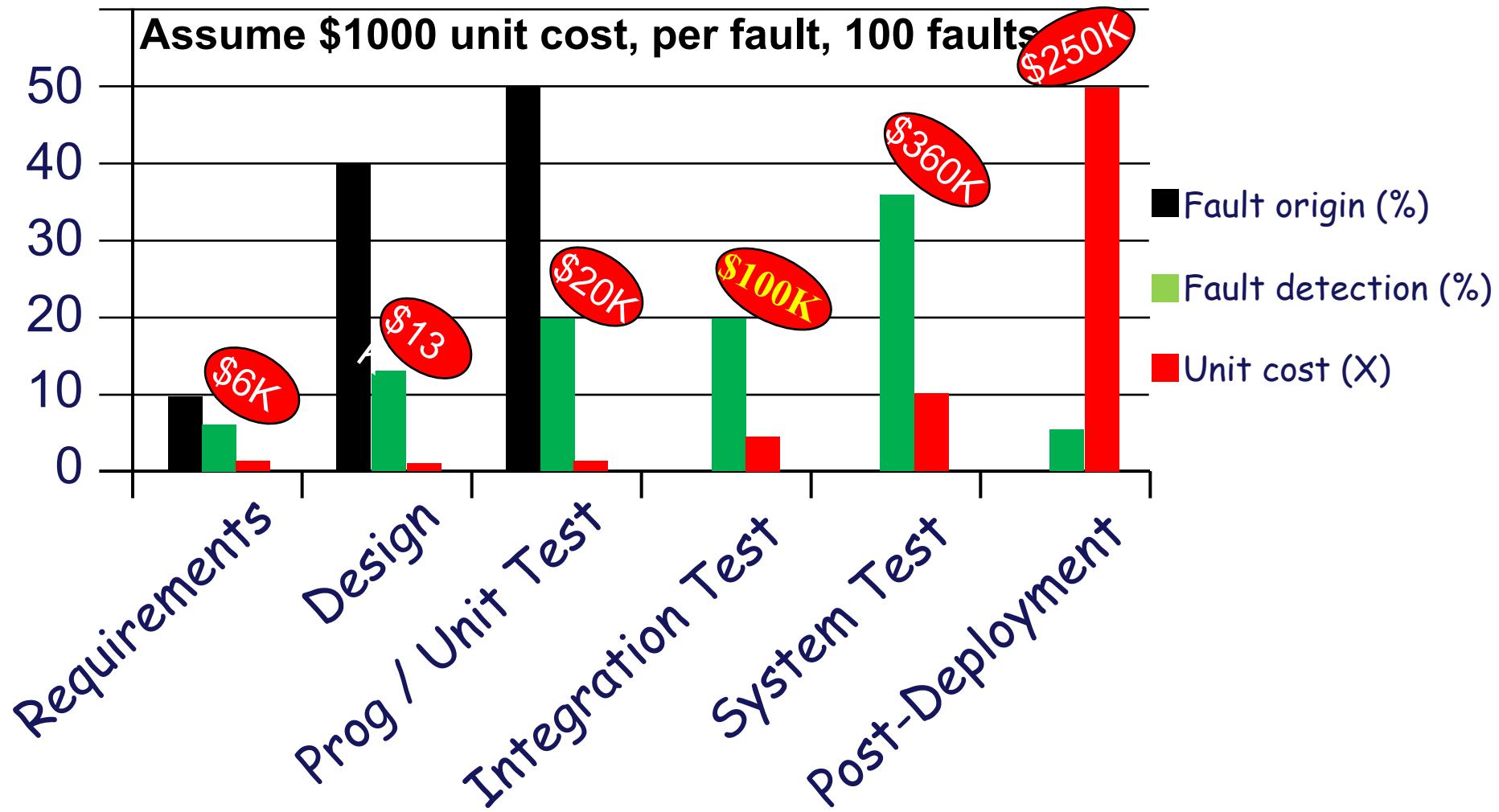
# Nếu không test?



*PM tôi có thể nói: “Kiểm thử quá tốn kém.”*

- Kiểm thử tốn thời gian và là phần tốn kém trong phát triển phần mềm
- Nhưng... KHÔNG kiểm thử còn tốn hơn!
- Nếu ta bỏ ít chi phí kiểm thử cho lúc đầu, chi phí kiểm thử sẽ tang!
- Lập kế hoạch kiểm thử sau khi phát triển sẽ rất tốn kém!

# Nếu kiểm thử quá muộn?



# Tóm tắt: Vì sao cần kiểm thử phần mềm?



Mục tiêu của kiểm thử viên là loại bỏ lỗi  
càng sớm càng tốt!

- Tăng chất lượng
- Giảm chi phí
- Đem lại sự hài lòng cho khách hàng