For Updated version, please click on this
http://ocw.ump.edu.my

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

BCS3323 – Software Testing and Maintenance

Overview of Testing

Editors
Prof. Dr. Kamal Z. Zamli
Dr. AbdulRahman A. Alsewari
Faculty of Computer Systems & Software Engineering
alswari@ump.edu.my

Chapter 1.2 Description

- Aims is to discover
  - The risk
  - How to calculate the risks
  - Software Quality
  - Difference between Testing vs Debugging
  - Seven Testing Principles

- Expected Outcomes
  - Students can explain the risk, calculation of the risk
  - The purpose of the risk calculation
  - Differentiated between Testing vs Debugging
  - Knowledge about the testing principles

- References
  - ISTQB
  - MSTB/GTB
  - http://www.softwaretestingclass.com/software-testing-tools-list/
  - http://www.softwaretestinggenius.com/articalDetails.php?qry=572#commentsList
  - https://www.guru99.com/software-testing-seven-principles.html

How much testing is required ?

- **Risk** – '*A factor that could result in future negative consequences; usually expressed as impact and likelihood"*
- It depends on RISK
  - risk of missing important faults
  - risk of incurring failure costs
  - risk of releasing untested or under-tested software
  - risk of losing credibility and market share
  - risk of missing a market window
  - risk of over-testing, ineffective testing

So little time, so much to test ..

- test time will always be limited
- use RISK to determine:
  - what to test first
  - what to test most
  - how thoroughly to test each item
    - what not to test (this time)

} i.e. where to place emphasis

- use RISK to
  - allocate the time available for testing by prioritising testing ...

So little time, so much to test ..

- RISK = Impact * Likelihood

		Impact				
Probability		Trivial	Minor	Moderate	Major	Extreme
	Rare	Low	Low	Low	Medium	Medium
	Unlikely	Low	Low	Medium	Medium	Medium
	Moderate	Low	Medium	Medium	Medium	High
	Likely	Medium	Medium	Medium	High	High
	Very Likely	Medium	Medium	High	High	High

Testing and quality

- Software quality can be measured by testing
- With testing can detect defects; software quality (and possibly reliability) is improved by fixing the these defects.
- what does testing test?
  - Functionality of the system, correctness of operation
  - non-functional qualities: reliability, usability, maintainability, reusability, testability, etc.

- Quality software is known as **reasonably bug-free**, can be **delivered on time** within budget, **meets requirements and/or expectations**, and is **maintainable**

Quality Products

VS

VS

ISO/IEC 9126: Internal and External Quality

Internal Quality

- Functionality
- Reliability
- Usability
- Efficiency

External Quality

- Maintainability
- Portability

Other factors that influence testing

- contractual requirements
- legal requirements
- industry-specific requirements
  - e.g. pharmaceutical industry (FDA), compiler standard tests, safety-critical or safety-related such as railroad switching, air traffic control

It is difficult to determine

how much testing is enough

but it is not impossible

Why do we test then?

- Specified requirements
- The software products are fit for purpose
- Defects deduction

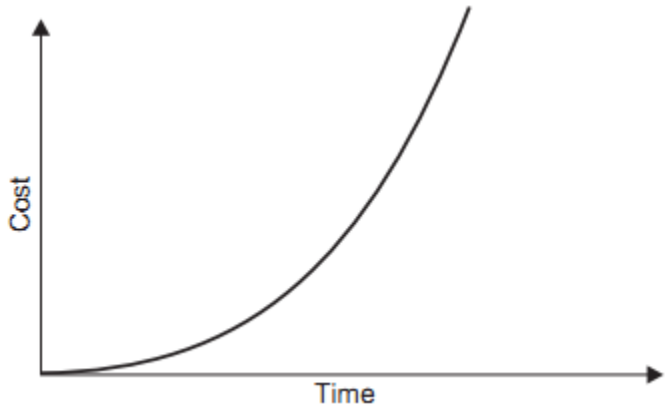
Testing vs Debugging?

- Testing != Debugging
- Testing finds defects. When a tester (i.e. via a set of test cases) finds a defect to be fixed, a programmer must do some work to locate the defect in the code and make the fix, termed debugging.
- The programmer examines the code and repair and checks the code for conformance.
- The tester then re-checks the fix through so-called regression testing.

Cost of Testing – Cost Escalation Model

Comparative cost to correct errors

Stage error is found	Comparative cost
Requirements	$1
Coding	$10
Program testing	$100
System testing	$1,000
User acceptance testing	$10,000
Live running	$100,000



Seven Testing Principles

General Testing Principles

The Seven Key Principles

1. Testing shows presence of Defects
2. Exhaustive Testing is Impossible!
3. Early Testing
4. Defect Clustering
5. The Pesticide Paradox
6. Testing is Context Dependent
7. Absence of Errors Fallacy

https://www.guru99.com/software-testing-seven-principles.html

General Testing Principles

The Seven Key Principles

1.Testing shows the presence of Defects

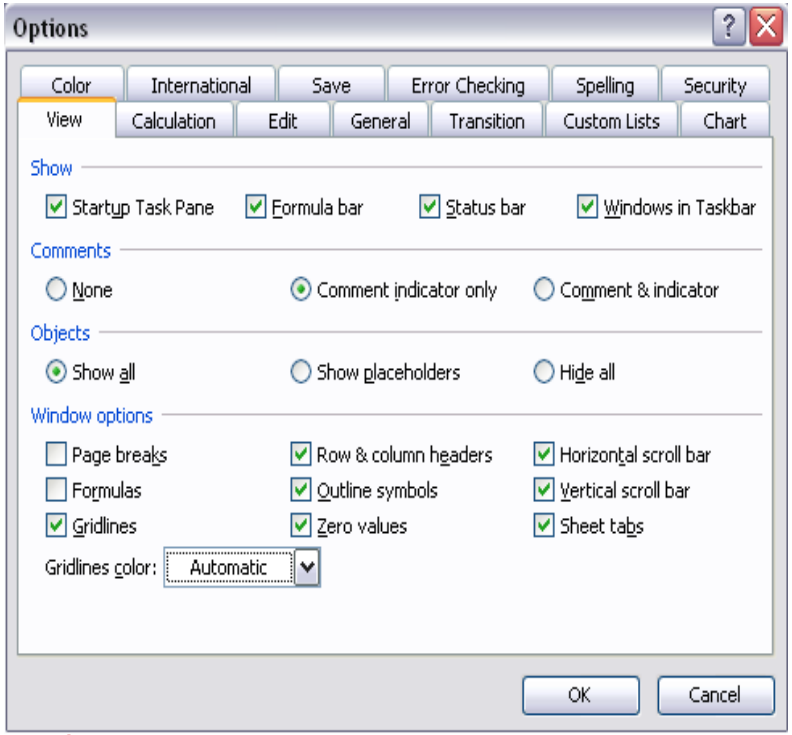
- We test to find Faults (a.k.a Defects)
- As we find more defects, the probability of undiscovered defects remaining in a system reduces.
- However Testing cannot prove that there are no defects present

General Testing Principles

The Seven Key Principles

2. Exhaustive Testing is Impossible!

- We have learned that we cannot test everything (i.e. all combinations of inputs and pre-conditions).
- That is we must Prioritise our testing effort using a Risk Based Approach.



There are 2^{20}x56 = 58,720,256 combinations to be tested.... resulting into combinatorial explosion of test cases.

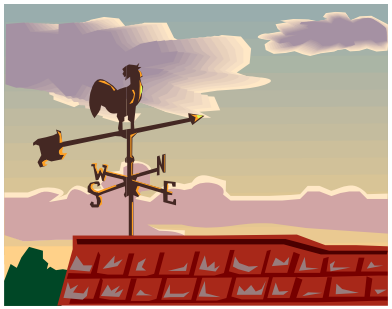
If 5 minutes is required to execute one test case, then it would require 559 years to complete all tests merely for View tab option!

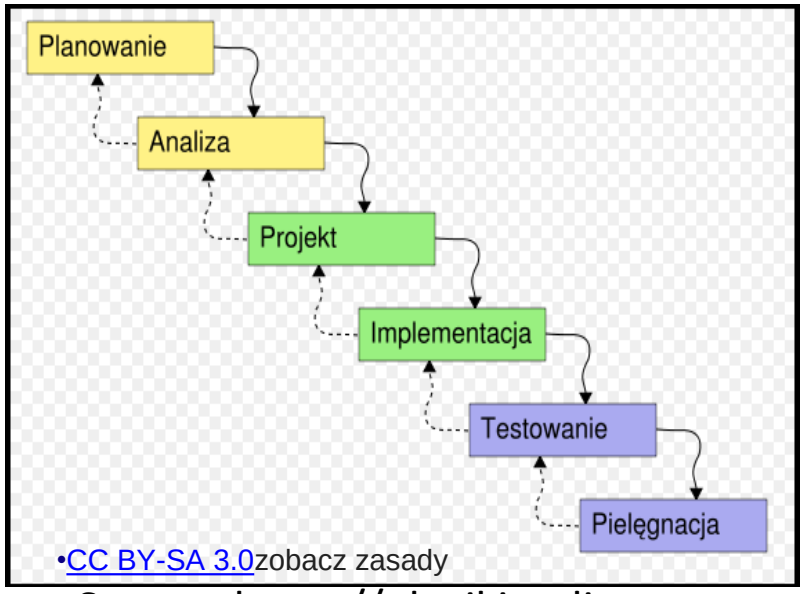
General Testing Principles

The Seven Key Principles

3.Early testing

Check the SRS, SDS, Code, etc.





Source: https://pl.wikipedia.org

General Testing Principles

The Seven Key Principles

4. Defect Clustering

- Defects are not evenly spread in a system
- They are ‘clustered’
- In other words, most defects found during testing are usually confined to a small number of modules
- Similarly, most operational failures of a system are usually confined to a small number of modules
- An important consideration in test prioritisation!

General Testing Principles

The Seven Key Principles

5.The Pesticide Paradox

- Testing identifies bugs, and programmers respond to fix them
- As bugs are eliminated by the programmers, the software improves
- As software improves the effectiveness of previous tests erodes
- Therefore we must learn, create and use new tests based on new techniques to catch new bugs

- *N.B It's called the "pesticide paradox" after the agricultural phenomenon, where bugs such as the boll weevil build up tolerance to pesticides, leaving you with the choice of ever-more powerful pesticides followed by ever-more powerful bugs or an altogether different approach.' – Beizer 1995*

General Testing Principles

The Seven Key Principles

6.Testing is Context Dependent

- Testing is depending on context. For example, safety-critical software systems are tested differently (more intensely and using other techniques) from commercial applications.

Game applications Vs Bank system

General Testing Principles

The Seven Key Principles

7. Absence of Errors Fallacy

- If we build a system and, in doing so, find and fix defects ….

  It doesn't make it a good system

- Even after defects have been resolved but the system or application unusable and/or does not fulfil the users' requirements and expectations

Who Tests Software?

- Programmers
  - Unit testing
  - Testing buddies can test other's programmer's code
- Users
  - Usability and acceptance testing
  - Volunteers are frequently used to test beta versions
- Quality assurance personnel
  - All testing types except unit and acceptance
  - Develop test plans and identify needed changes

Tester's Attitude

- Absolute pessimists
- People tend to see what they want to see, hence, they will miss defects
- Tester's aim is to break the software
  - Reveal all relevant defects
  - Find out problems real user would experience
- Testing Is all about exceptions, special cases, error situations, and complicated unexpected combination.

Tester's Contribution

- Explore, investigate, and measure
- Provide quality related information to stakeholders
- Destructive towards software under test but highly constructive towards people