Software Quality Assurance & Testing

TOPIC2: LECTURE 2

Seven testing principles

- **Principle 1 – Exhaustive testing is impossible (1)**
  Testing everything (all combinations of inputs and preconditions) is not feasible except for trivial cases. Instead of exhaustive testing, risk analysis, economics and priorities should be used to focus testing efforts.
- **Principle 2 – Defect clustering (2)**
  A small number of modules contain most of the defects discovered during prerelease testing, or are responsible for the most operational failures.
- **Principle 3 – Pesticide paradox(3)**
  If the same tests are repeated over and over again, eventually the same set of test
  cases will no longer find any new defects. To overcome this "pesticide paradox",
  the test cases need to be regularly reviewed and revised, and new and different
  tests need to be written to exercise different parts of the software or system to potentially find more defects.

Seven testing principles

- **Principle 4 – Testing shows presence of defects.**

Testing can show that defects are present, but cannot prove that there are no defects. Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness.

- **Principle 5– Absence-of-errors fallacy**

  Finding and fixing defects does not help if the system built is unusable and does
  not fulfil the users' needs and expectations.

Seven testing principles

- **Principle 6 – Early testing**
  Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives.

- **Principle 7 – Testing is context dependent**
  Testing is done differently in different contexts. For example, safety-critical software is tested differently from an e-commerce site.

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