Introduction to Software Testing
Software Testing

- Testing is the process of detecting errors by running the actual software and verifying that it works as it should
  - Test cases, Expected results, Actual results

- Testing is by far the most popular QA activity (but not the most effective)

- Technical reviews (design reviews, code reviews, etc.) are cheaper and more effective than testing, but are often not done

- Research has shown that all forms of testing combined usually find less than 60% of the errors present
Software Testing

There are many different types of testing. Three of the most important are:

- **Unit Testing**: testing individual modules (e.g., classes) to make sure they work in isolation before combining them with the rest of the system

- **Integration Testing**: testing the combination of multiple modules after they have been integrated together
  - If the individual modules work in isolation, can there possibly be defects in their combination?
  - YES! The interactions between the modules can contain defects

- **System Testing**: testing done on the entire program, after it is completely integrated
Software Testing

• Exhaustively testing software is not feasible
  – The number of possible input combinations is effectively infinite
  – The number of unique paths through the code is effectively infinite
  – You might not live long enough to exhaustively test a non-trivial software system

• We must do partial testing because we only have enough resources (time and money) to run relatively few test cases

• Partial testing can never prove the absence of defects
  – If the system passes all your test cases, there could still be defects, you just need more or better test cases to find them
Software Testing

• Effective testing lies in intelligently choosing the relatively few test cases that will actually be executed
  
  – Test all requirements and features defined in the requirements spec. and functional spec.
  
  – Test cases should not be redundant (i.e., each one should follow a different path through the code)
  
  – Focus on scenarios that users are likely to encounter in practice
  
  – Analyze the program’s design and code to find potential weak areas
  
  – Analyze all points at which data enters the system and look for ways to attack it
Software Testing

• Approaches to test case design are generally divided into two broad categories: Black Box Testing and White Box Testing

• **Black Box Testing**
  – The tester has limited knowledge of the inner workings of the item being tested
  – Test cases are based on the specification of the item's external behavior
  – Can be done at the Unit, Integration, and System levels

• **White Box Testing**
  – The tester has knowledge of the inner workings of the item being tested
  – Test cases are based on the specification of the item's external behavior AND knowledge of its internal implementation
  – Most commonly done at the Unit level
Software Testing

• Testing is unlike other software development activities because the goal is to break the software rather than to create it.

• Effective testing requires an assumption that defects actually exist, and a desire to find them.

• If you think you won't find defects, or you don't want to, you won’t be effective in your testing.

• Testing by both developers and an independent testing group are essential
  – They have different perspectives and motivations.
  – They do different kinds of tests (developers do white box, test team does black box), which tend to discover different types of defects.
Software Testing

• Defects are not evenly distributed (i.e., they tend to cluster)

• Research has shown that:
  – 80% of a system's defects are found in 20% of its code
  – 50% of a system's defects are found in 5% of its code

• There is a high correlation between bugs and complex code.
  – Use tools to measure code complexity, and focus testing on those modules with the most complex code

• One goal of testing is to identify the most problematic modules
  – Redesign may be needed if there is an inherent design flaw
  – Or, replace buggy module with a third-party library/product
Software Testing

• Automation of test cases is essential to make frequent re-running of test cases feasible

• Create programs whose purpose is to test other programs

• Inventing ways to automate test cases can be interesting and challenging work that requires lots of software design and coding (sometimes called “Test Engineering”)

• Some tests are difficult to automate and must be run manually