# Software Testing

# Outline

- Software testing
- Principles of Software Testing
- Standard Testing Activities
- Testing Life Cycle
- Static Testing
- Dynamic Testing
- Black-Box Testing
- White-Box Testing

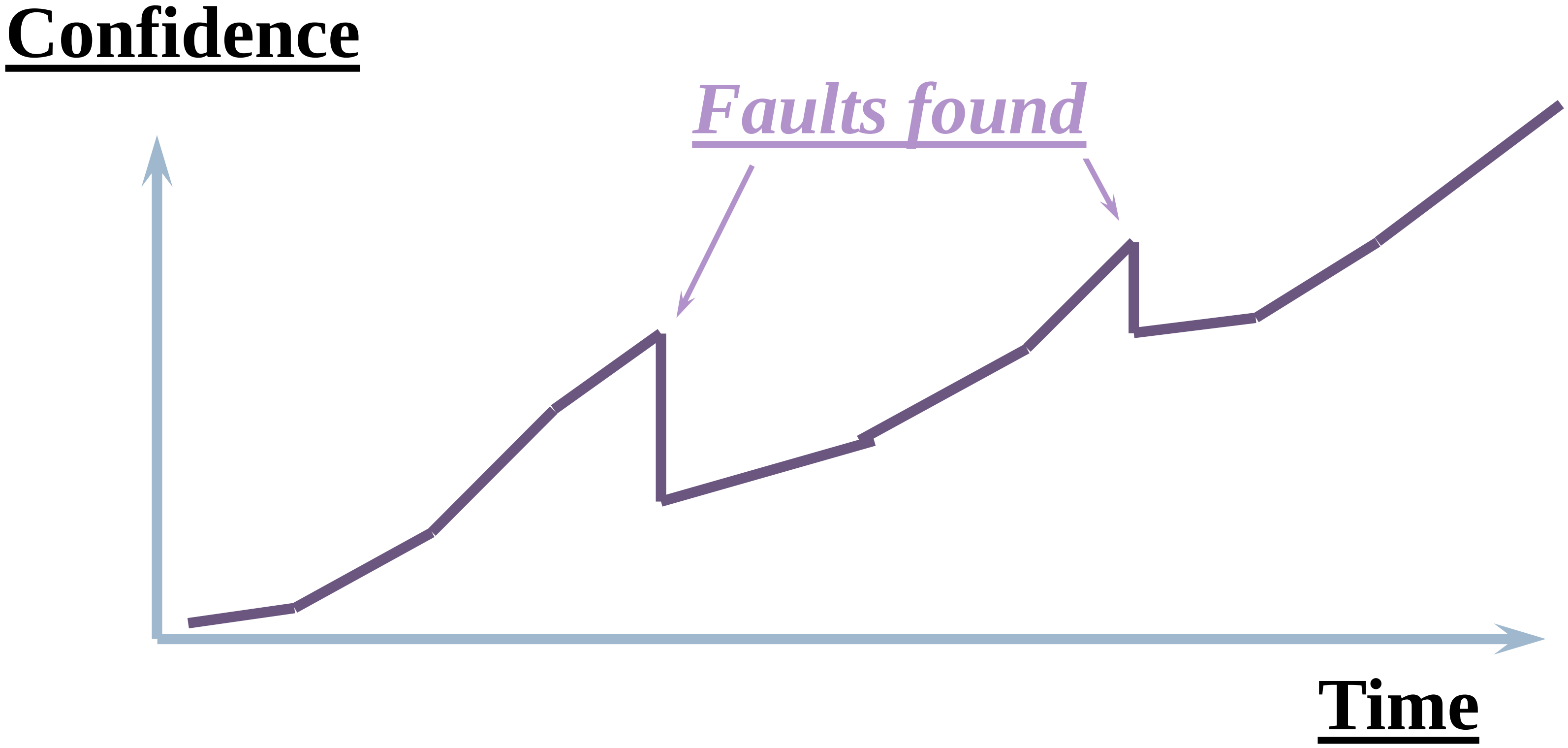
# Software Testing

- **Software testing** is an investigation conducted to provide stakeholders with information about the quality of the product or service under test.
- Software testing can be stated as the process of validating and verifying that a software program/application/product:
  - meets the requirements that guided its design and development;
  - works as expected
  - can be implemented with the same characteristics.

# Why test?

- Build confidence
- Demonstrate conformance to requirements
- Find faults
- Reduce costs
- Show system meets user needs
- Assess the software quality

# Confidence



**No faults found = confidence?**

# Principles of Software Testing

- A good test case is one that has a high probability of finding an undiscovered defect. So, the test cases (the program input) should be selected systematically and with care, both for correct and incorrect behavior.

- A successful test is one that uncovers an undiscovered defect. So, testing is psychologically destructive since it tries to demolish the software that has been constructed.

- Testing can never completely identify all the defects within software

# Principles of Software Testing

- Testing cannot show the absence of defects, it can only show that they are present.
- Testing is quite an ineffective method of quality assurance. [Though, usually the most applicable one.]
- Successful testing shall be followed by a separate debugging phase.
- Testing is also by itself a process that must be systematically managed (and assisted with special testing tools).

# ABCD of TESTING

- Testing Methods: There are many approaches to software testing.
- Reviews, walkthroughs, or inspections are referred to as **static testing,** whereas actually executing programmed code with a given set of test cases is referred to as **dynamic testing.**
- Software testing methods are traditionally divided into **white- and black-box testing.**
- Visual testing is to provide developers with the ability to examine what was happening at the point of software failure.

# Testing Stages

- Unit testing
  - testing of individual components
- Module testing
  - testing of collections of dependent components
- Sub-system testing
  - testing collections of modules integrated into sub-systems
- System testing
  - testing the complete system prior to delivery
- Acceptance testing
  - testing by users to check that the system satisfies requirements.

# Testing Life Cycle

# Testing Life Cycle

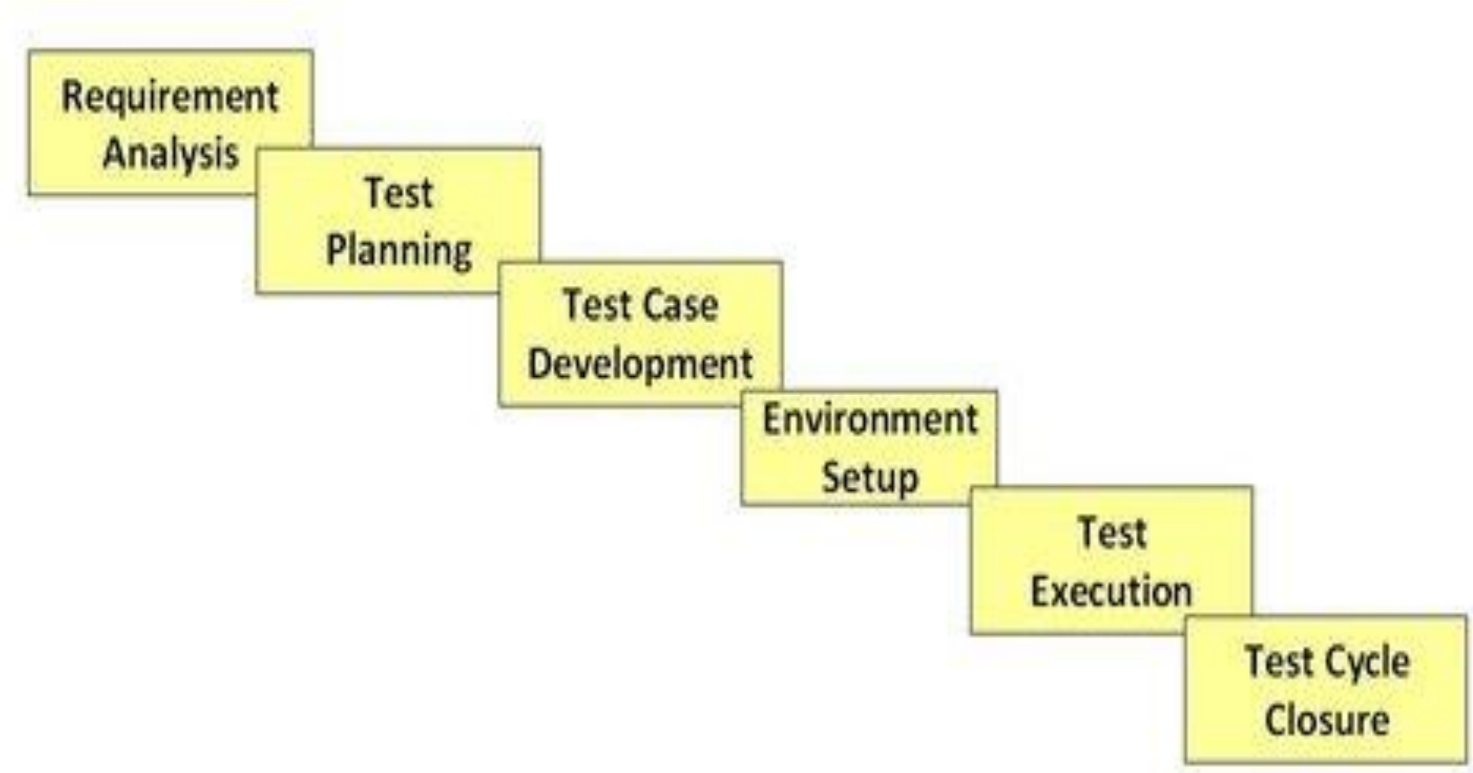
- The time from the initial idea for a product (software) until it is disposed of is called the product life cycle, or software life cycle. Testing is a very important part of the life cycle of any product

- Testing is not an isolated activity, nor is it a development activity.

- Testing is a support activity:
  - nothing developed entails nothing to test

- Testing in a development life cycle is broken down into a number of test levels

# Testing Life Cycle



# Static Testing

# What is Static Testing

- The software is not executed.
- Rather the specifications, documentation and source code that comprise the software are examined in varying degrees of detail.
- Testing of a component or system at specification or implementation level without execution of that software (e.g., reviews or static code analysis)

# Static Testing

- Types of Static Testing:
  - People-based - generally known as “reviews” but there are a variety of different ways in which reviews can be performed.
  - Tool-based - examine source code and are known as “static analysis”.

# Benefits of Static Testing

- Early detection of defects prior to test execution.
- Early warning about suspicious aspects of the code or design, by the calculation of metrics, such as a high complexity measure.
- Identification of defects not easily found by dynamic testing.
- Detecting dependencies and inconsistencies in software models.
- Improved maintainability of code and design.
- Prevention of defects.

# Dynamic Testing

# Dynamic Testing

- Dynamic testing is a term used in software engineering to describe the testing of the dynamic behavior of code.
- Dynamic analysis refers to the examination of the physical response from the system to (change with time).
- The software must actually be compiled and run, it involves working with the software, giving input values and checking if the output is as expected

# Test cases

- Dynamic testing means testing based on specific test cases by execution of the test object or running programs.

- Test cases for Dynamic Testing can be derived:
- Black-Box Technique
- White-Box Technique

# Black-Box Testing

- An approach to testing where the program is considered as a ‘black-box’.
- The program test cases are based on the system specification.
- No knowledge of how it is structured inside the box is required.
- Tester needs to know what the system does and not how it does it.

# Black Box Techniques

- This type of testing attempts to find errors in the following categories:
  - incorrect or missing functions
  - interface errors
  - errors in data structures or external database access, performance errors, and  initialization and termination errors.
- Techniques:
  - Equivalence Partitioning
  - Boundary Value Analysis
  - Use cased based testing
  - Decision Table

# Advantages of Black-Box

- More effective on larger units of code than glass box testing
- Tester needs no knowledge of implementation, including specific programming Languages
- Tester and programmer are independent of each other.
- Will help to expose any ambiguities or inconsistencies in the specifications
- Test cases can be designed as soon as the specifications are completed.

# Disadvantages of Black-Box

- Only a small number of possible inputs can actually be tested.
- Without clear and concise specifications, test cases are hard to design
- May leave many program paths untested
- Cannot be directed toward specific segments of code which may be very complex.

# White Box Testing

- Sometime called whitebox/glass-box testing.

- Derivation of test cases according to program structure. Knowledge of the program is used to identify additional test cases.

- Objective is to exercise all program statements (not all path combinations).

# Path testing

- Structural testing strategy whose objective is to exercise every independent execution path through a component or program.
- If every independent path is executed then all statements in the component must have been executed at least once.
- Path testing is used at the unit testing and module testing stages.
- A flow graph is a skeletal model of all paths through the program. (describes the program control flow)