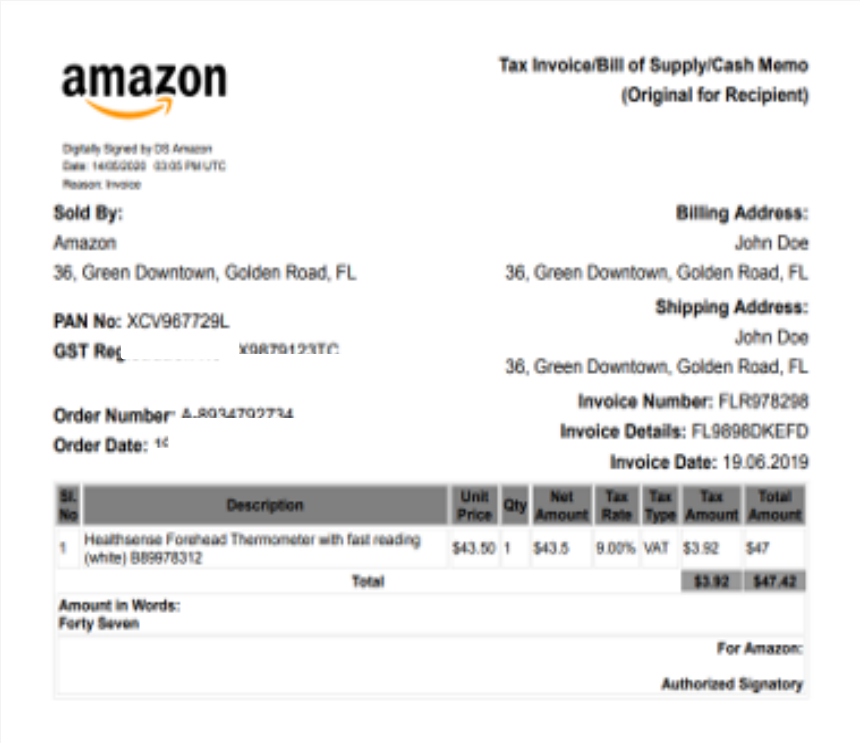
Think and Tell



amazon

Tax Invoice/Bill of Supply/Cash Memo
(Original for Recipient)

Digitally Signed by DS Amazon
Date: 14/05/2020 03:05 PM UTC
Reason: Invoice

Sold By:
Amazon
36, Green Downtown, Golden Road, FL

PAN No: XCV967729L
GST Re[illegible]

Billing Address:
John Doe
36, Green Downtown, Golden Road, FL

Shipping Address:
John Doe
36, Green Downtown, Golden Road, FL

Order Number: [illegible]
Order Date: [illegible]

Invoice Number: FLR978298
Invoice Details: FL9898DKEFD
Invoice Date: 19.06.2019

Sl. No	Description	Unit Price	Qty	Net Amount	Tax Rate	Tax Type	Tax Amount	Total Amount
1	Healthsense Forehead Thermometer with fast reading (white) BB9978312	$43.50	1	$43.5	9.00%	VAT	$3.92	$47
	Total						$3.92	$47.42

Amount in Words:
Forty Seven

For Amazon:

Authorized Signatory

Source: https://bills99.com/

What do you think is wrong with this bill?

Display Screen

Source: https://pholder.com/

Is there any problem with the display of this mobile application?

Credit Card

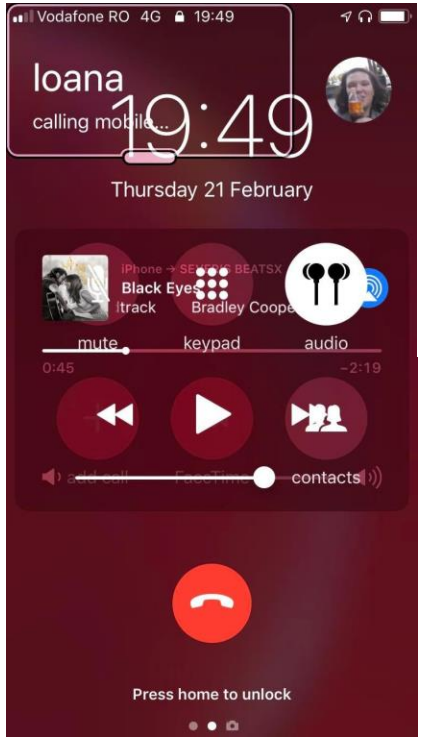
Source: https://www.bajajfinserv.in/

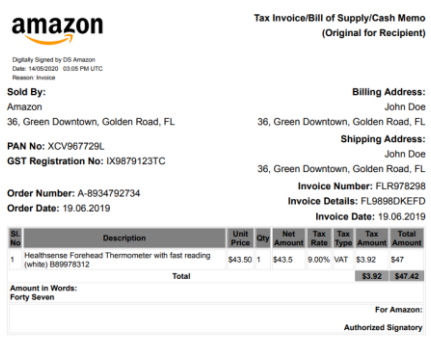
How is this credit card different from others?

How do these problems impact consumers and businesses?

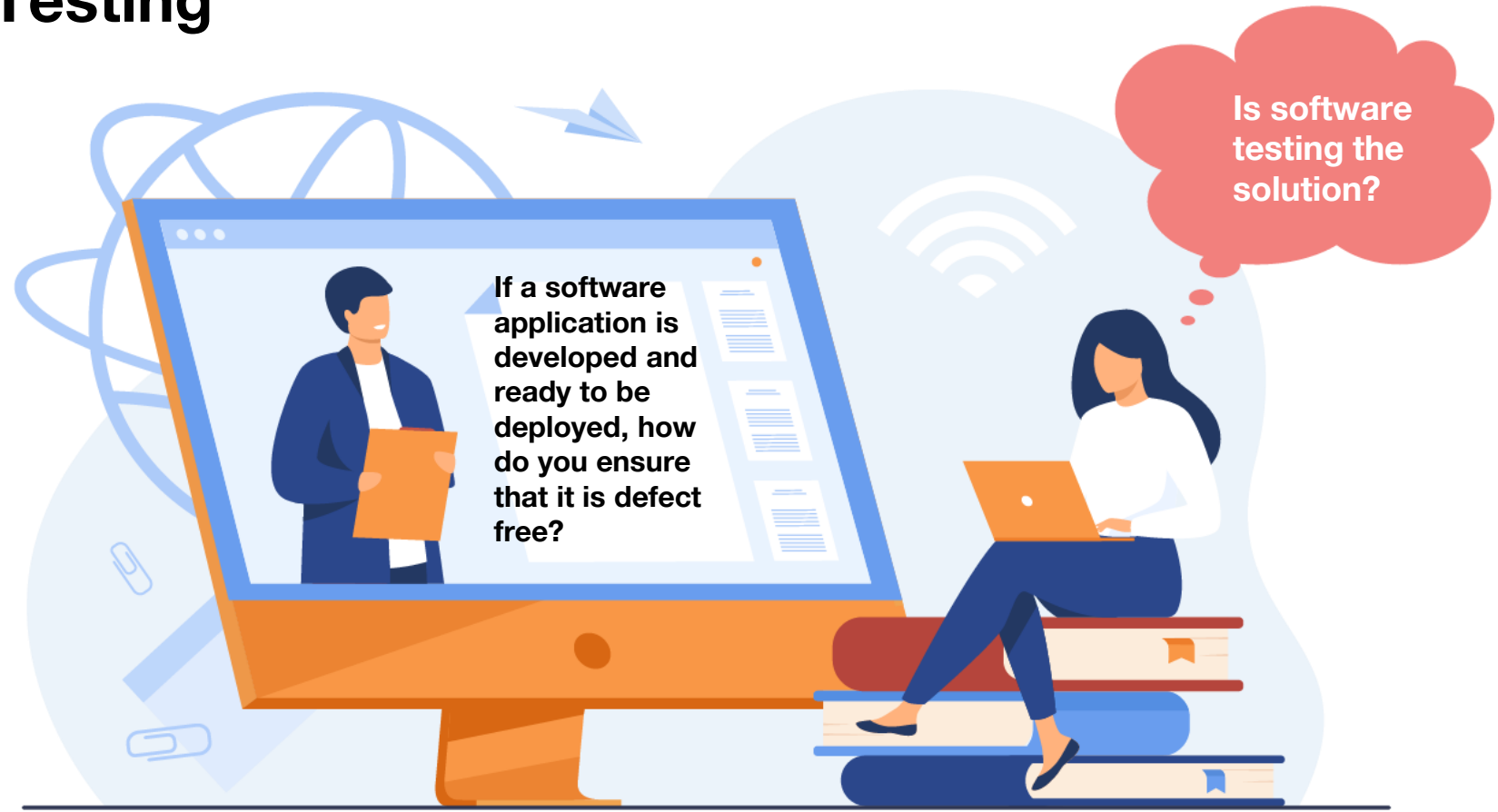
When should these defects be identified?

What steps must companies take to avoid such occurrences?





Testing



Unit Testing with JUnit

Learning Objectives

- Define unit testing and state its importance
- List the features of JUnit framework
- Describe the JUnit architecture
- Write test cases in JUnit
- Use the JUnit annotations and assertions

Unit Testing

- Is one of the best development practices used to test smaller units of code
- It tests the individual units of an application. A unit can be a method of a class, a complete class, a set of inter-related classes, or a module
- It ensures that even the smallest unit of code is bug-free and reusable
- It reduces the chances of encountering errors at a later stage of development
- It ensures that the code functions efficiently on each unit of application
- It is usually created by programmers and occasionally by white box testers

A Testing Tool

Which tool can we use to efficiently perform unit testing on a piece of code?

JUnit

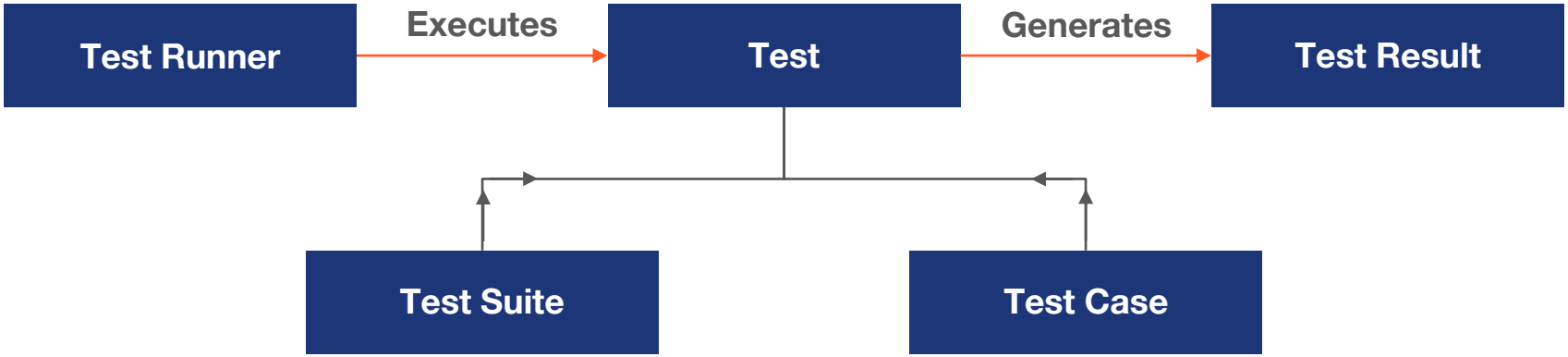
- JUnit is an open-source testing framework
- It is used to write test cases for Java
- It emphasizes the implementation of Test-Driven Development (TDD)

Features of the JUnit Framework

- Provides unit test cases that verify the output of the code under test
- Allows automatic execution of the JUnit test cases
- Shows a progress bar that signifies the success or failure of test cases
- Allows organization of multiple test classes into a test suite
- Allows execution of test cases in a test suite one after another automatically
- Saves time by allowing concurrent execution of multiple tests

The JUnit Architecture

The architecture of JUnit refers to the process used by the JUnit framework to execute the tests and display the results



Elements of the JUnit Architecture

Test Case: It is the smallest unit of any JUnit test that verifies the functionality of the method being tested

Test Suite: Is created when we have multiple test classes and we want to automatically execute these classes one after the other

Test: It is a collection of one or more test cases or test suites that are executed to perform the testing of an application

Test Runner: When you execute a test, the test runner is executed in the background, which displays the result of the test in terms of pass or fail.

Test Result: It refers to the outcome of a test, which is displayed to the user. This test result is collected from the test class.

JUnit Test Case

It carries a set of instructions based on which you can test the functional aspect of the code being tested

The three components of a test case are:

- Input
- Event
- Expected response

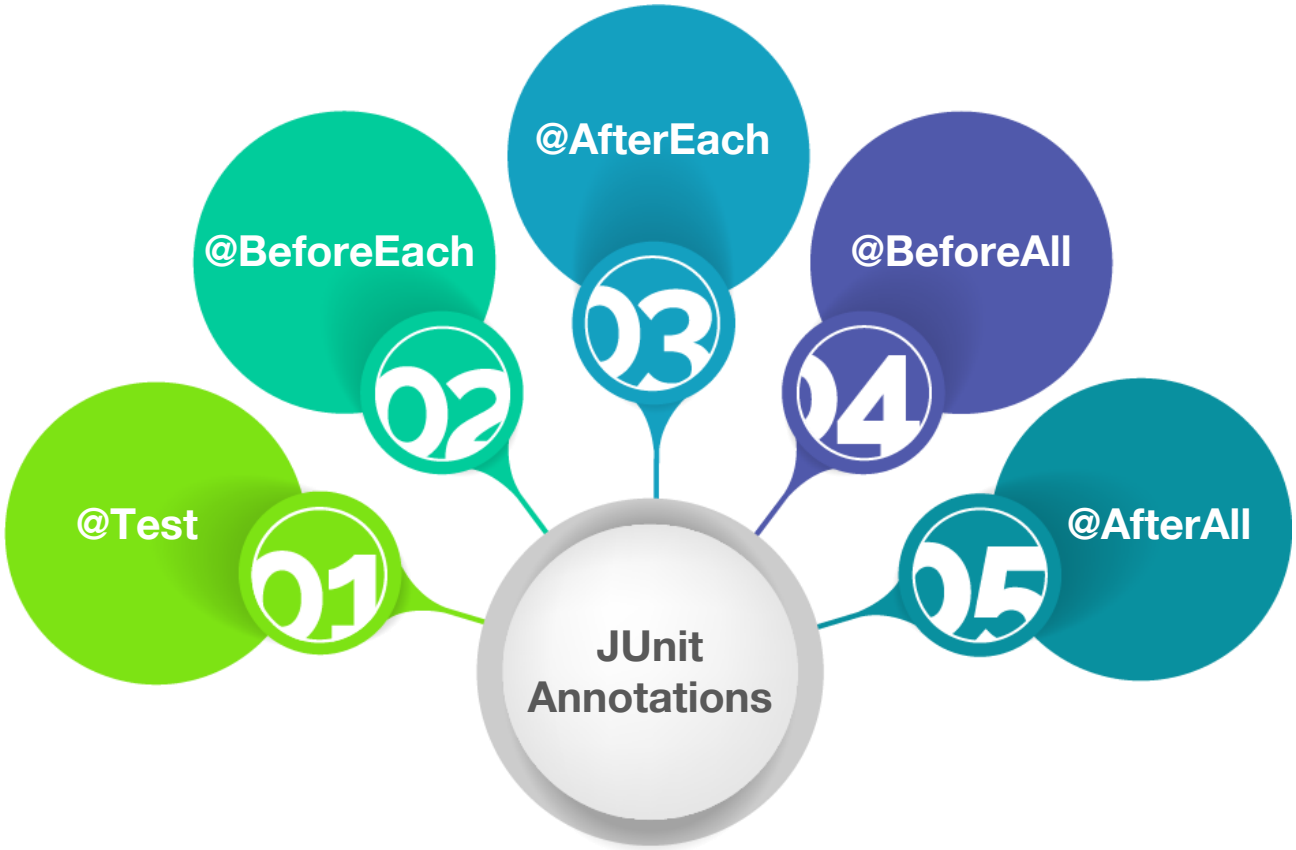
JUnit Test Case (contd.)

To write a JUnit test case, we have to create test classes in a different package so that the test code can be separated from the application code

A test class can have multiple test cases

Multiple test classes containing class-specific test cases can also be created

JUnit Annotations



JUnit Assert Statements

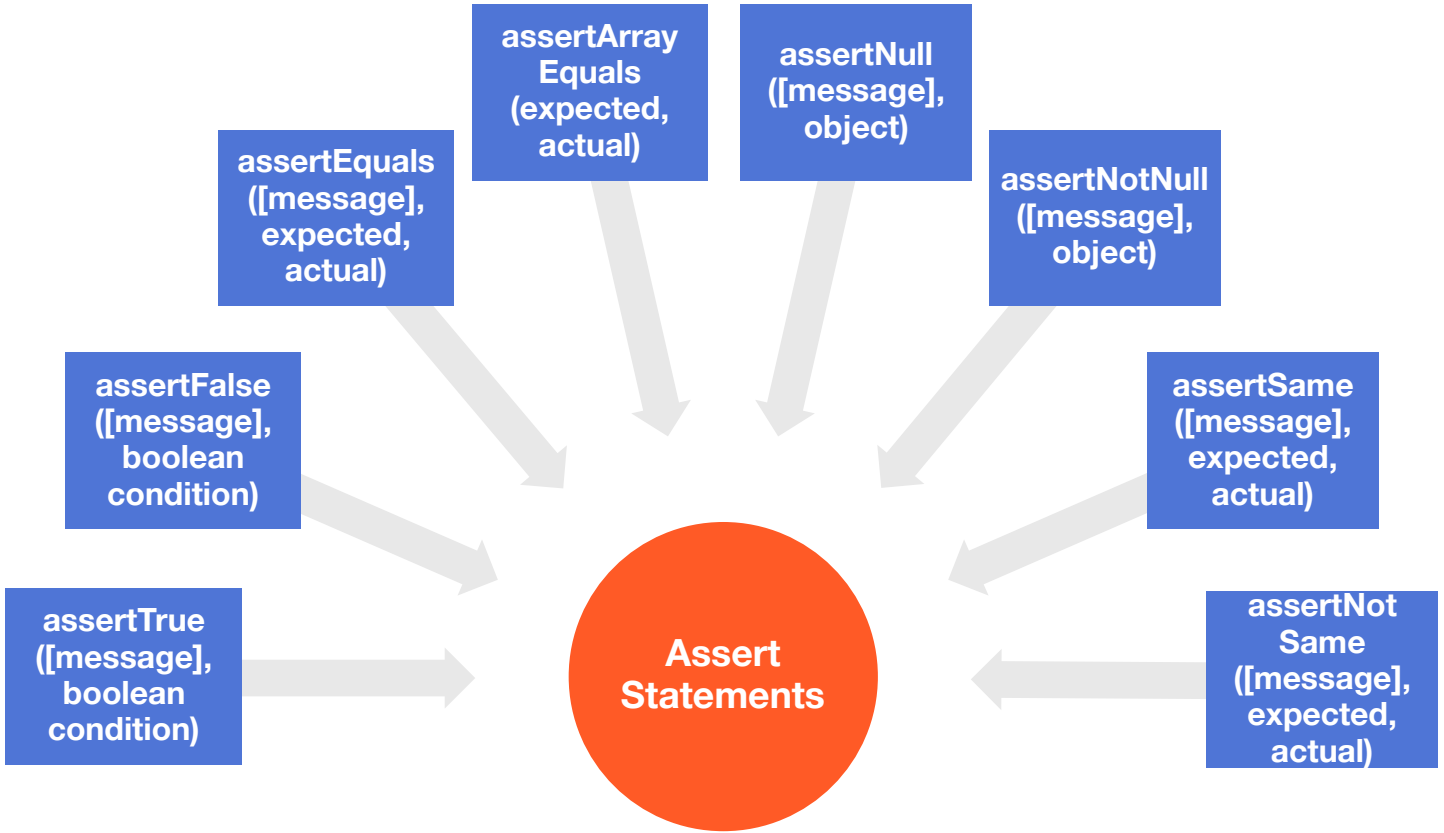
- Test the functionality of a function after a particular function has been annotated using the @Test annotation

- Verify the output of the function being tested with the expected output

Static methods in the Assert class:

  - Verify the expected and actual results
  - Compare the expected value with the actual value returned by the test
  - Throw the exception, "`AssertionException`", if the comparison fails

JUnit Assert Statements (contd.)



Interactive Demo

Write a program to develop a standard calculator that performs basic operations using the following methods:

Add
Subtract
Multiply
Divide

Write test cases to test each method.

Interactive Demo

Create an application to calculate the volume of a cone, cube, and sphere. Also, create test cases to test the functionality of these methods.

Key Takeaways

- Unit testing
- Features of Junit framework
- Elements of JUnit architecture
- JUnit test cases
- JUnit annotations and assert statements