



SYNTAX
TECHNOLOGIES

SDLC DAY 4

Introduction to Software Testing
Testing Applications & Hierarchy



Why do we Test?





We Test to

- To eliminate the bugs
- Reduce the risk of error
- Continuously improve the functionalities of software and its quality so that the product works as expected before it reaches to the customers.

Main Goal is make sure that the end-user is as comfortable as possible.



Software Testing is necessary because we all make mistakes. Some of those mistakes are not important, but some of them are expensive or dangerous. We need to check everything and anything we produce because things can always go wrong – humans make mistakes all the time.

What is Quality?

- Dictionary Definition: “The standard of something as measured against other things of a similar kind”.
- **Quality is Perception** - Good quality for you may not be good quality for me, so it differs from person to person.
- **Our main goal as a tester** is to make better products by trying to know what makes end user more excited and what will make them come back to our product.
- Links to check out <https://www.youtube.com/watch?v=RzCsBxqmmys>
<https://www.youtube.com/watch?v=G8TPZR6O4eo>
<https://www.youtube.com/watch?v=ZwZTmkqQOFM>





What is Software Testing?

- Software testing is a process of executing a program or application with the intent of finding the software bugs.
- Process of validating and verifying that a software program or application or product: Meets the business and technical requirements that guided its design and development
- Works as expected
- Make sure 100% customer and end-user satisfaction

Software testing is a process to make sure that the application developed will satisfy and meet end-users and customers expectations.



What is a Bug?

Wikipedia Definition:

A **software bug** is an error, flaw, failure or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways



What is a Bug?

- It is an **unexpected result** you find during testing when you are comparing expected result with actual result.
- Bug can also be a tester finding software difficult to understand, hard to use, or slow from an end-user point of view.





Bug Examples

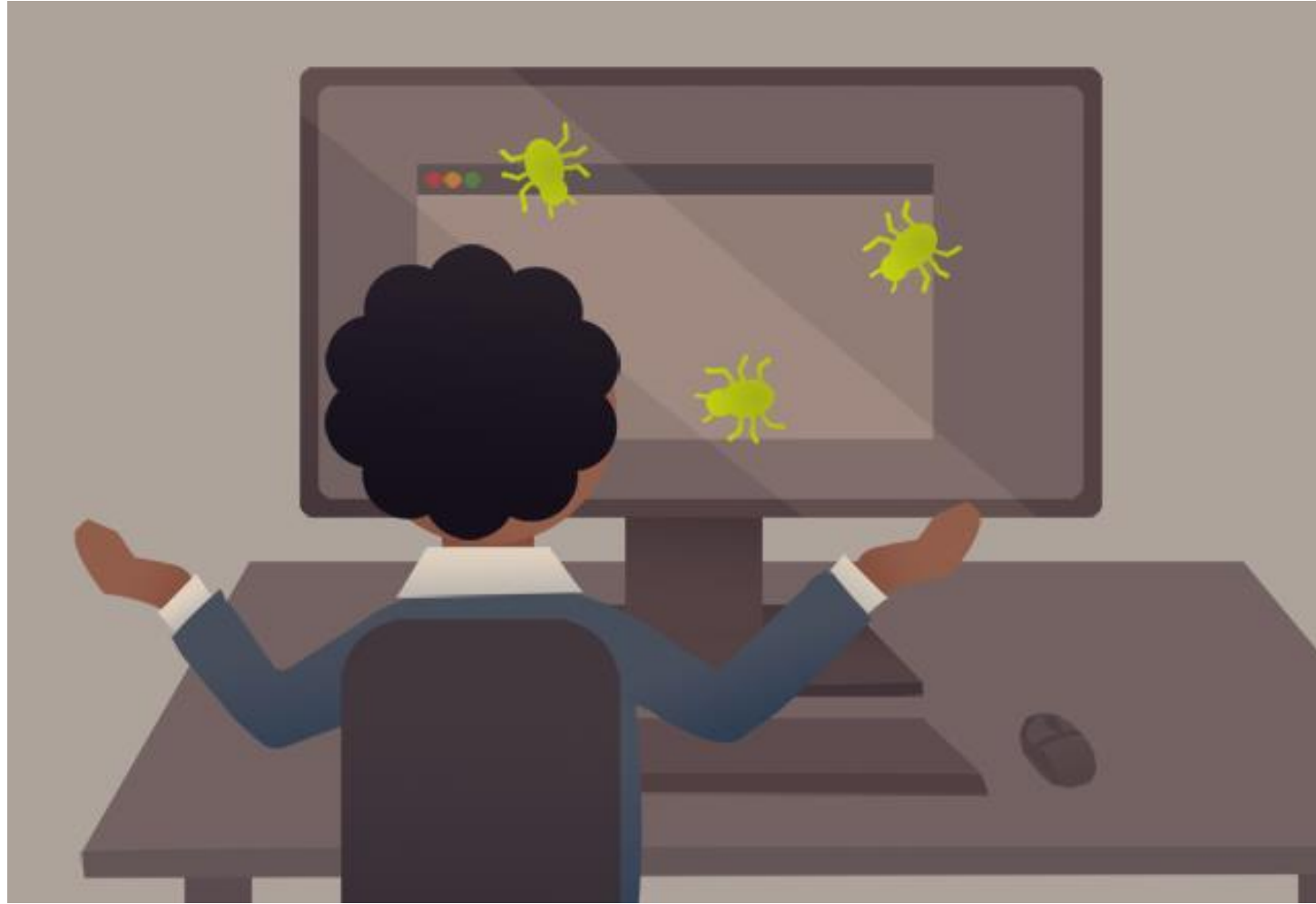


- **Software Doesn't do something that specification says it should do**
 - Example: User was not able to login with valid credentials.
- **Software Does something that specification says it should NOT do.**
 - Example: User was able to login with invalid credentials.
- **Software Does something that specification does not mention.**
 - Example: Software allowed user to add 2 spouses to the policy
- **Software Doesn't do something that specification doesn't mention but it should.**
 - Example: Login button is in-between User Name and Password field.

<https://www.youtube.com/watch?v=UlhGhj1tj-A>

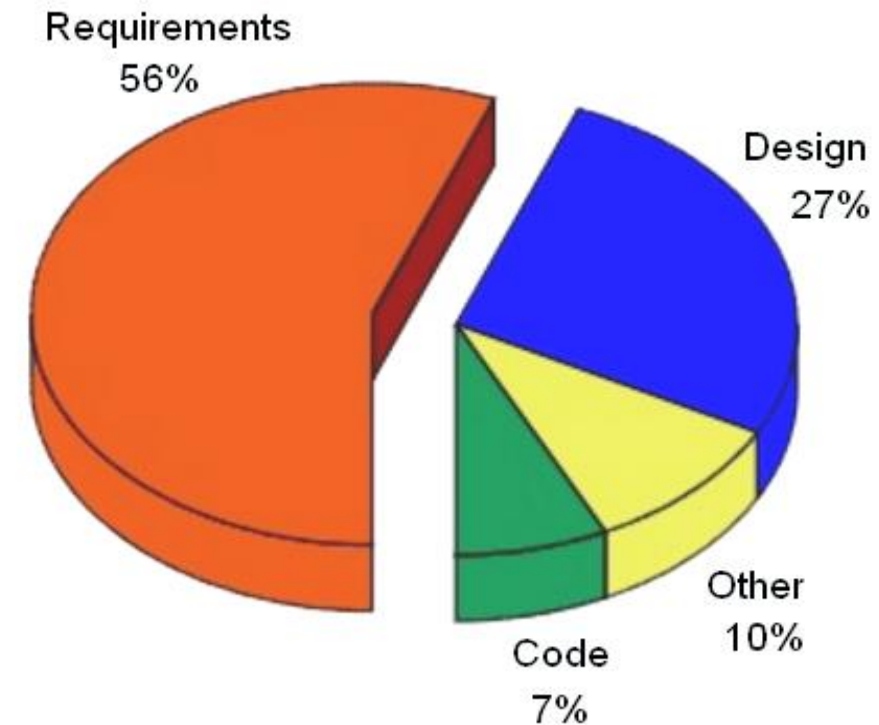


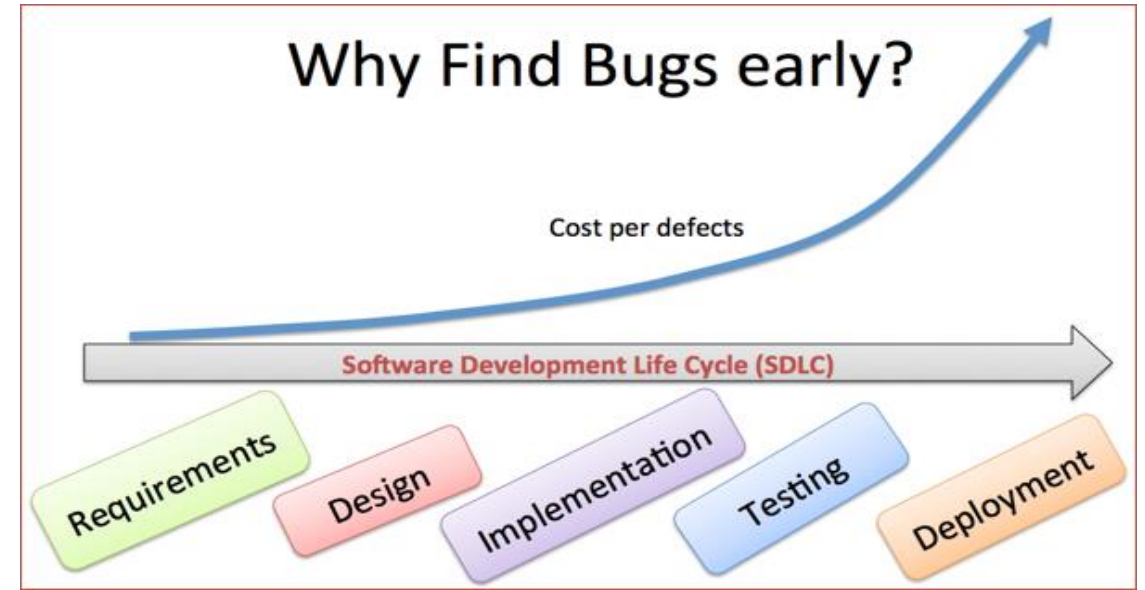
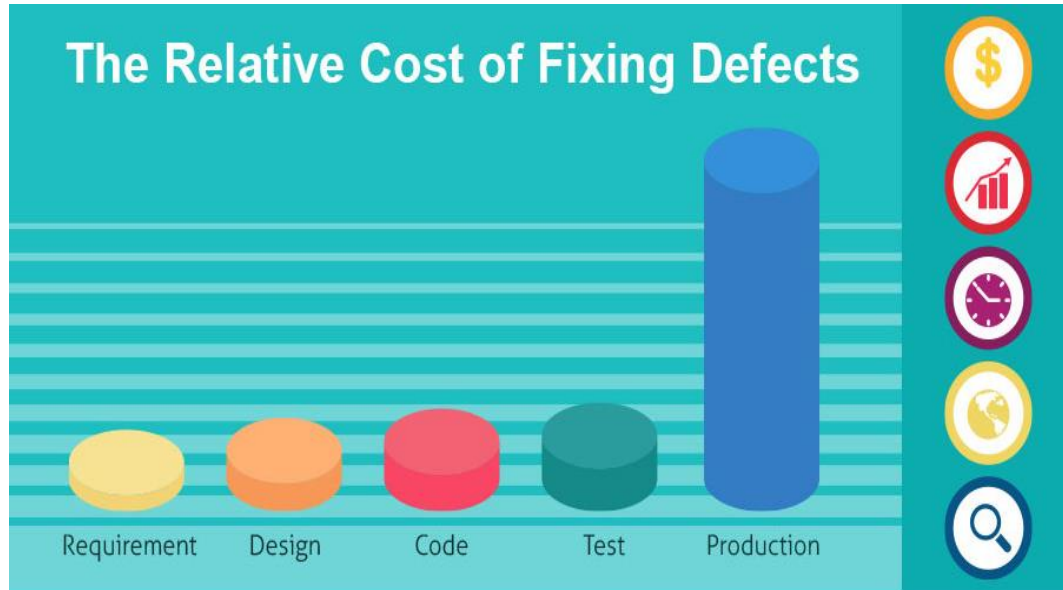
Why Bugs Occur?



Why Bugs Occur?

It has been shown that some 56% of all software defects emerge during the **requirement phase**, 27% in the **design phase**, and only 7% during the **development phase**.







3 Types of Application We Test



Web Application

- A simple definition of a web application is a software program which the user accesses over a network with a web browser.
- Web applications requires only a web browser enabling the users to access the application from almost any computer. Some web applications requires Internet Explorer , chrome, Firefox .Data can be shared with everyone in an organization regardless of location.





Desktop Application

- Desktop application runs on personal computers and work stations, so when you test the desktop application you are focusing on a specific environment. You will test complete application broadly in categories like GUI, functionality, Load, and backend .They are standalone applications and does not depend on internet connectivity.

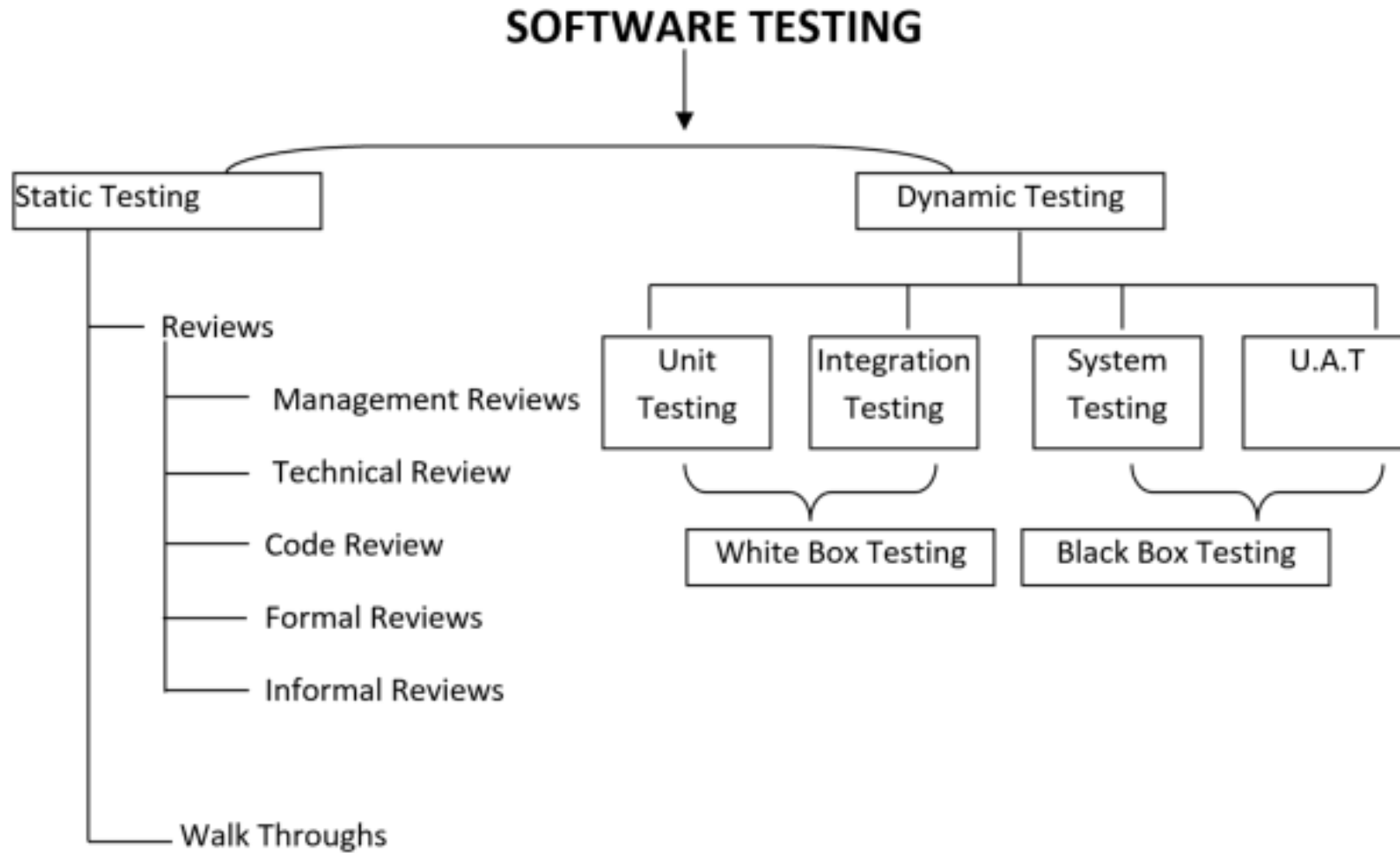




Client Server Application

- In client server application you have two different components to test. Application is loaded on server machine while the application exe on every client machine. You will test broadly in categories like, GUI on both sides, functionality, Load, client server interaction, backend. This environment is mostly used in Intranet networks. You are aware of number of clients and servers and their locations in the test scenario.

Examples: Internal Hospital System, Internal Human Resources software, IT HelpDesk software etc.





STATIC TESTING / TECHNIQUES

STATIC TESTING:

It is a process of verifying are we developing the right system or not, this static testing will be carried out with the help of Reviews and Walkthroughs.

REVIEWS:

Examining a project related work or a process related work is called a Review.

For Ex: Examining requirements, Design, Code etc...

“STATIC TESTING DOES NOT EXECUTE THE CODE “

TYPE OF REVIEWS:

- 1)Management Review
- 2)Technical Review
- 3)Formal Review
- 4)Informal Review

MANAGEMENT REVIEW:

This review will be conducted by top level or middle level management to monitor the project status.

These reviews are help full for the management to take the necessary corrective actions if there are any Slippages.

TECHNICAL REVIEWS:

These reviews will be conducted among the technical people to decide the best approach of implementation, for there are any ambiguities while implementing a technical job.

FORMAL REVIEWS:

If a review is carried out with a prior plan by following a systematic procedures and proper documentation then this reviews are called Formal Reviews

INFORMAL REVIEWS

If a review is conducted with out following any procedures and documentation then these reviews are called informal reviews



Peer Reviews & Code Reviews are the best examples for informal Reviews.

PEER REVIEWS

Reviews conducted among colleagues are called as **PEER REVIEWS**.

CODE REVIEWS

Review among colleagues where they look at each others code to make sure we follow coding standard and any flaws etc.

OBJECTIVES OF REVIEWS

- To find defects in requirements
- To find defects in design
- To identify the deviations in any process
- To provide valuable suggestions to improve the process

WALK THROUGHS

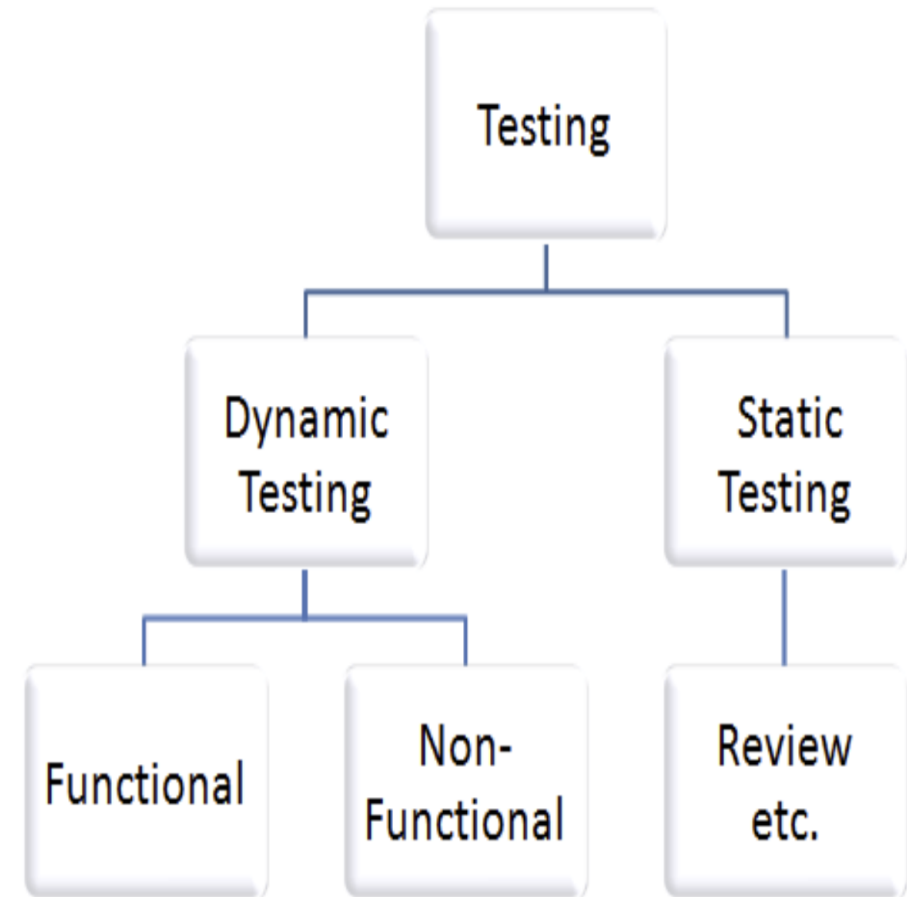
A step by step presentation conducted by the author or by the domain expert about a subject. KTS (Knowledge Transfer Sessions) are best example of walkthroughs.

DYNAMIC TESTING / TECHNIQUES

Dynamic Testing code is executed. It checks for functional behavior of software system , memory/cpu usage and overall performance of the system. Hence the name "Dynamic"

Main objective of this testing is to confirm that the software product works in conformance with the business requirements. This testing is also called as Execution technique or validation testing.

Dynamic testing executes the software and validates the output with the expected outcome. Dynamic testing is performed at all levels of testing and it can be either **black** or **white box testing**.





Two Types of Testing

Functional Testing

- It tests the functionality of the application.
- First thing we do when a code or new functionality comes from developers so make sure that the functionality is working fine according to business requirements, and doesn't have any major bugs.
- Functional testing is performed by manual testers and by automation team.
- Examples:
 - Can user login? Can user logout?
- Functional testing is designed to determine that the application's features and operations perform the way they should.

Non-functional Testing

- Testing software application or system for its non-functional requirements: The way a system operates, rather than specific behaviors of that system.
- Non-functional testing is designed to figure out if your product will provide a good user experience.
 - For example, non-functional tests are used to determine how fast the product responds to a request or how long it takes to do an action (performance testing)
- Non-functional testing wants to see if the product stands up to customer expectations.
- Non-functional testing wants to know that the product "behaves" correctly.
- Look and Feel are also part of non-functional testing



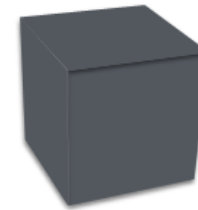
White-box testing

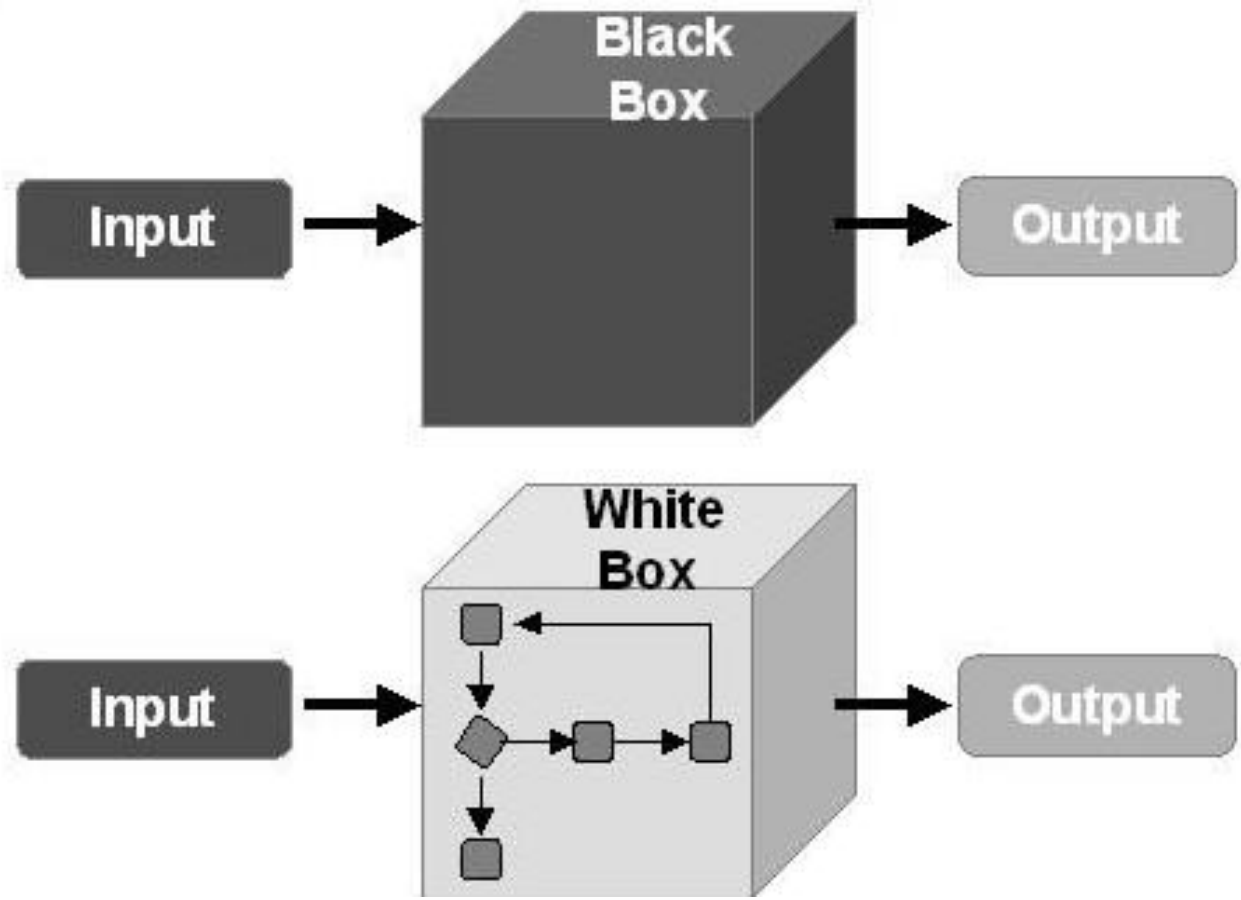
- Requires detailed programming skills.
- Mostly done by developers where they execute their code before it goes to QA. environment like unit testing
- Internal structure is known to the Developer who is going to test the software.



Black-box testing

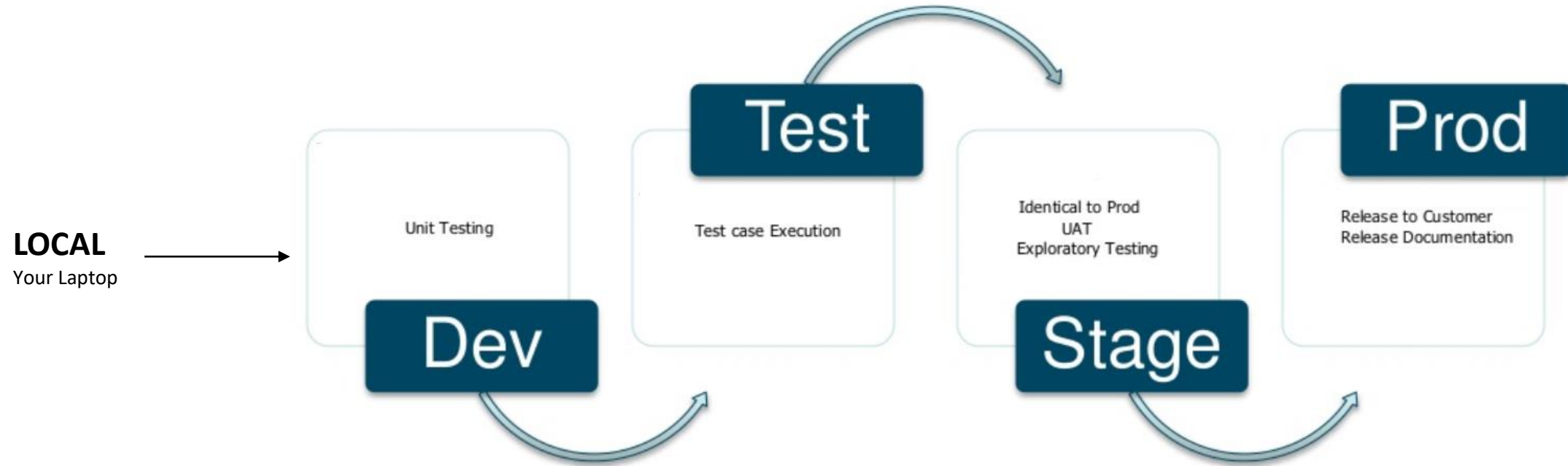
- Requires no knowledge of internal paths, structures, or implementation of the software being **tested**.
- Tester doesn't know what is inside.
- All the tester knows is what the software is supposed to do and if the software is doing its job correctly
- Black-box testing only cares about input / output







DEVELOPMENT & TEST FLOW





DEV,TEST,STAGE are all environment that are similar and contains dummy data.

Production is only environment which is live where we usually don't have access as testers or only have read-only access.


Next slides have real examples of actual application and you can see how all environment practically looks similar .DEV ,TESTER and UAT do their testing in their environments after Deployment occurs. Others names are **DEV,SIT,UAT**



AgAdvantage Acrea x AgAdvantage - Powe x

dev.alisdev.com/Legacy/Login.aspx?ReturnUrl=/

THIS IS DEV ENVIORMENT WHERE
DEVELOPERS CHECK WHAT THEY
DEVELOP AND TEST



Release 7.19 - Powered by AgriLogic

Please sign in

Enter email address (required)

alexvald@agrilogic.com

Enter password (required)

☐ Show password

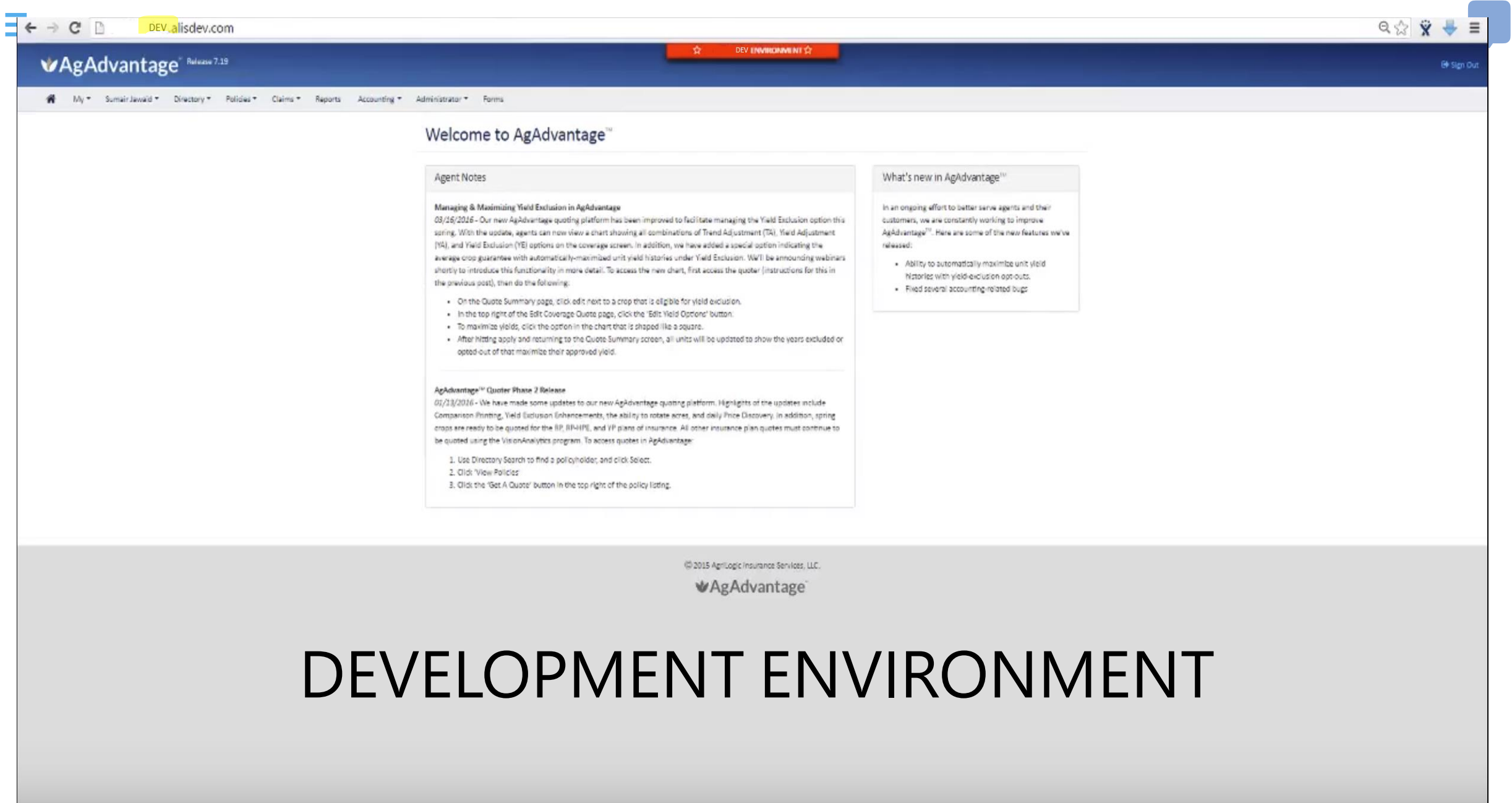
Sign in

[Forgot your password?](#)

WARNING: Using the "Back" button on your internet browser may cause unforeseen consequences with the behavior of the system.

© 2015 AgriLogic Insurance Services, LLC.


DEVELOPMENT ENVIRONMENT



DEVELOPMENT ENVIRONMENT



test.alisdev.com/Legacy/Login.aspx?ReturnUrl=/


Release 7.19 - Powered by AgriLogic

Please sign in

Enter email address (required)

sjawaid@agrillogic.com

Enter password (required)

☐ Show password

Sign In

[Forgot your password?](#)

WARNING: Using the "Back" button on your internet browser may cause unforeseen consequences with the behavior of the system.

© 2015 AgriLogic Insurance Services, LLC.

TEST ENVIRONMENT



Welcome to AgAdvantage™

Agent Notes

Managing & Maximizing Yield Exclusion in AgAdvantage

03/15/2016 - Our new AgAdvantage quoting platform has been improved to facilitate managing the Yield Exclusion option this spring. With the update, agents can now view a chart showing all combinations of Trend Adjustment (TA), Yield Adjustment (YA), and Yield Exclusion (YE) options on the coverage screen. In addition, we have added a special option indicating the average crop guarantee with automatically maximized unit yield histories under Yield Exclusion. We'll be announcing webinars shortly to introduce this functionality in more detail. To access the new chart, first access the quote (instructions for this in the previous post), then do the following:

- On the Quote Summary page, click edit next to a crop that is eligible for yield exclusion.
- In the top right of the Edit Coverage Quote page, click the 'Edit Yield Options' button.
- To maximize yields, click the option in the chart that is shaped like a square.
- After hitting apply and returning to the Quote Summary screen, all units will be updated to show the years excluded or opted-out of that maximize their approved yield.

AgAdvantage™ Quoter Phase 2 Release

01/13/2016 - We have made some updates to our new AgAdvantage quoting platform. Highlights of the updates include Comparison Printing, Yield Exclusion Enhancements, the ability to rotate acres, and daily Price Discovery. In addition, spring crops are ready to be quoted for the RP, RP-HPE, and YP plans of insurance. All other insurance plan quotes must continue to be quoted using the VisionAnalytics program. To access quotes in AgAdvantage:

1. Use Directory Search to find a policyholder, and click Select.
2. Click 'View Policies'
3. Click the 'Get A Quote' button in the top right of the policy listing.

What's new in AgAdvantage™

In an ongoing effort to better serve agents and their customers, we are constantly working to improve AgAdvantage™. Here are some of the new features we've released:

- Ability to automatically maximize unit yield histories with yield-exclusion opt-outs.
- Fixed several accounting-related bugs

© 2015 AgriLogic Insurance Services, LLC.



TEST ENVIRONMENT




Release 7.19 - Powered by AgriLogic

Please sign in

Enter email address (required)

Enter password (required)

☐ Show password

[Forgot your password?](#)

WARNING: Using the "Back" button on your internet browser may cause unforeseen consequences with the behavior of the system.

© 2015 AgriLogic Insurance Services, LLC.

STAGE/UAT ENVIRONMENT



Policy

Print

Policy #:
Reinsurance Year:
Crop Year:
State:
Added Counties Option:
Name on Policy:
In Care Of:
Entity Type:
Agency:
Agent:
Underwriter:
Power of Attorney:
Status:
Old Policy #:
Previous Carrier:
Limited Resource Farmer:
Claim Authorization Offset:

New
2017
2017
-Select-
☐ (Category B Crops Only)
PAULA J TAYLOR

Associations
SILVEUS INSURANCE GROUP-Dumas
Select
ELIZABETH MENDOZA

Incomplete

Select
☐
☐
Save

Prepare for Printing

© 2015 AgriLogic Insurance Services, LLC.

STAGE/UAT ENVIRONMENT



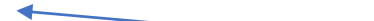
Software Testing Hierarchy

USERS-
STAGE/UAT
Environment



acceptance

QA Manual/Automation-
TEST/SIT Environment



system

integration

Developer –
DEV Environment

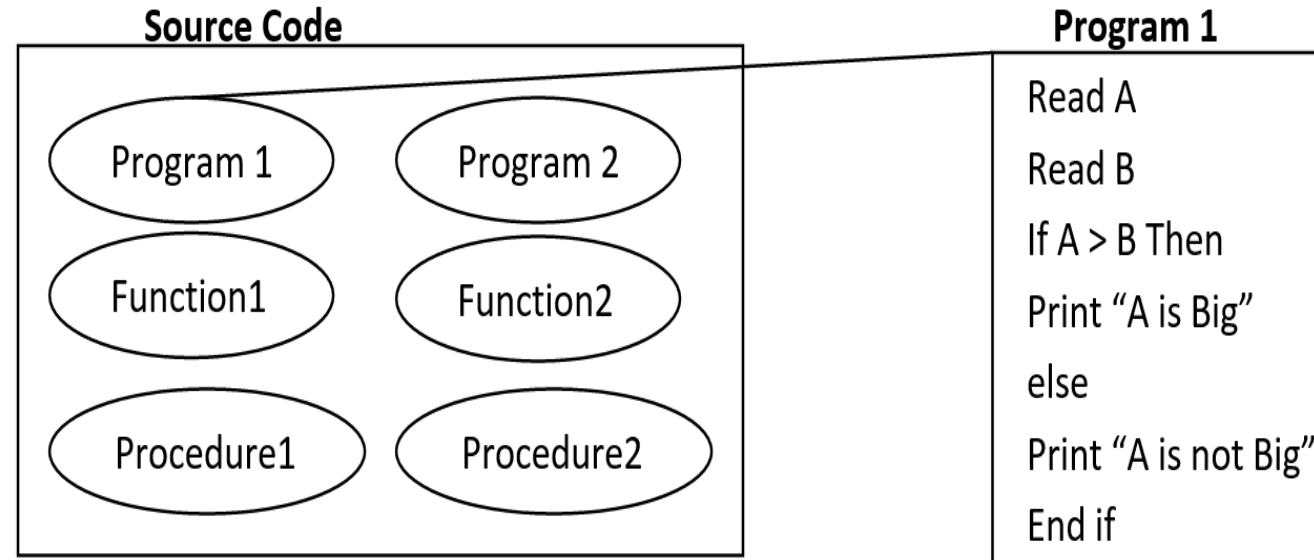


unit

Unit Testing

- Unit is smallest testable part of an application like functions, classes etc. A smallest testable portion in the source code of the application is called Unit Testing.
- In the source code of the application are called units, testing conducted on this unit to check does the code is working as expected or not is called unit testing.
- Its part of the white box testing.
- It's done by developers. They execute and evaluate their code and if they find any bugs they fix it before it come to QA environment and testers finds it.
- As a tester we have never done unit testing





Testing conducted on Program 1 by developer to check that the code behind program 1 is working as expected or not is called **UNIT TESTING**.

Integration Testing

- Once the unit testing is completed developers will integrate all source code units and checks interactions among all these units which are called integration testing.
- Validate the interaction of related components, modules and systems to ensure they work together.
- Test individual software components to verify interaction between various software components and detect interface defects. After the integration testing has been performed on the components, they are readily available for system testing.
- For Ex. Amazon video will work fine as a standalone component. But when you put Amazon video code together with the rest of the website it is no longer working. We need to make sure that all the components when integrated together are able to interact with each other and work properly.



System Testing

- System testing is testing complete and fully integrated software product. Everything together like integrated component, Front end, back end, database, server etc.
- Test that the behavior and functionality of the complete and fully integrated application complies with specified requirements.
- Tests the combination of software and hardware as The software works on top of hardware
- Also called End to End scenario testing.
- It also tests overall user experience.





Acceptance Testing

- Once the application has completed it is delivered to user or customer and tested by them for acceptance testing.
- Acceptance testing is basically done by the user or customer although other stakeholders may be involved as well. Work with subject matter experts and business stakeholders to guide them through the user acceptance testing process, ensuring that the application meets business needs
- Also known as UAT User Acceptance Testing.
- Companies can also have UAT Team to perform User Acceptance Testing. Once QA team and Dev team completes the testing then UAT team tests the product. UAT team assures that the product works fine, as expected before it goes to business.