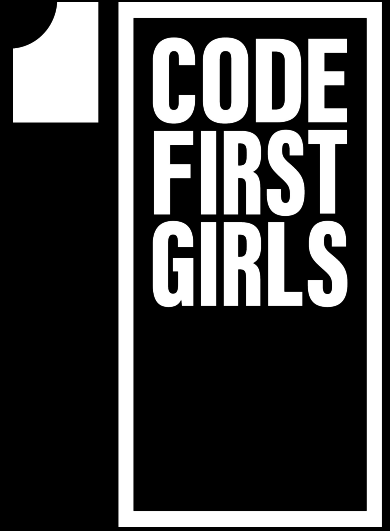


# TESTING

## LESSON 9



CFGDEGREE → FOUNDATION MODULE

# AGENDA



- 01 Introduction to Testing
- 02 Testing Methods
- 02 Types of Testing
- 03 Test Driven Development Overview
- 04 Testing Frameworks Overview
- 05 Mocking
- 06 Unit Tests – Examples and Exercises

# TESTING

## WHY IS IT IMPORTANT

- The testing is important since it discovers defects/bugs before the delivery to the client
- It makes the software more reliable and easy to use.
- Thoroughly tested software ensures reliable and high-performance software operation
- Testing is based on external expectations  
- Internal behavior of the application is unknown.

## #WHO DOES TESTING EXAMPLES



# TESTING

## WHY IS IT IMPORTANT

### # WHEN TO START

**An early start is preferred, but depends on the development model being used.**

- During the requirement gathering phase (analysis and verification of requirements are considered as testing).
- Reviewing the design in the design phase with the intent to improve the design of a system.
- Testing performed by a developer on completion of the code or any increment
- Any feature release

### # WHEN TO STOP

**Testing is a never-ending process, however, the following aspects can be considered:**

- Testing Deadlines
- Completion of test case execution
- Completion of functional and code coverage testing to a certain point
- Bug rate falls below a certain level and no high-priority bugs are identified
- User confirmation
- Management decision

# SOFTWARE TESTING

## QA, QC & TESTING

Quality Assurance	Quality Control	Testing
Focuses on processes and procedures rather than actual testing on the system.	Focuses on actual testing by executing the software with an aim to identify bug/defect.	Focuses on actual testing.
Process-oriented activities.	Product-oriented activities.	Product-oriented activities.
Preventive activities.	It is a corrective process.	It is a preventive process.

**CASE STUDY:** let's discuss how these responsibilities would work in a company that delivers food ordered online

# SOFTWARE TESTING

## TYPES OF TESTING

### # MANUAL

A Tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug:

- Use test plans,
- Test cases
- Test scenarios
- Exploratory testing

### # AUTOMATION

A Tester writes scripts and uses another software to test the product. Automation Testing is used to re-run the test scenarios quickly, and repeatedly:

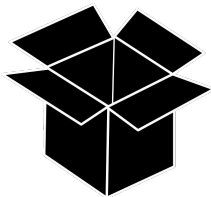
- Identifying areas for automation
- Writing test scripts
- Development of test suits
- Execution of scripts
- Create result reports

# SOFTWARE TESTING

## METHODS

### BLACK BOX TESTING

- The technique of testing without having any knowledge of the interior workings of the application.
- The tester is oblivious to the system architecture and does not have access to the source code.



Advantages	Disadvantages
Well suited for large code segments.	Limited coverage
Code access is not required.	Tester only has limited knowledge about an application.
Separates user's perspective from the developer's perspective	Blind coverage, since the tester cannot target specific code.
Many testers can test the application with no knowledge of programming language (BA, End User)	The test cases are difficult to design.

# SOFTWARE TESTING



## FUNCTIONAL TESTING TYPES

### UNIT TESTING

Ensures that each part of the code delivers the desired output. In unit testing, developers only look at the interface and the specification for a component.

### INTEGRATION TESTING

Testing of combined parts of an application to determine if they function correctly. Checking that modules which are working fine individually do not show bugs when integrated.

### SYSTEM TESTING

Tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets the specified Quality Standards.

### REGRESSION TESTING

Whenever a change is made, it is possible that other areas in the app have been affected by it. RT verifies that a fixed bug hasn't resulted in another functionality or business rule violation.

### SMOKE TESTING

Performed on the 'new' build to verify if the basic functionalities are working or not. The objective is to verify that the critical functionality of the system is working fine.

### USER ACCEPTANCE TESTING (UAT)

UAT is the last phase of the software testing process. In UAT actual software/app users test the software to make sure it can handle required tasks in real-world scenarios.

### NON-FUNCTIONAL TESTING

Tests non-functional but equally important attributes:

- Speed
- Capacity
- Scalability
- Security
- User INterface






# UNIT TESTING

## IN PYTHON

- A unit test is a scripted code level test designed in Python to verify a small "unit" of functionality.
- Python Test cases are comparatively easy to write.
- With the increased use of Python, Python-based test automation frameworks are also becoming popular.

## # POPULAR PYTHON TESTING TOOLS

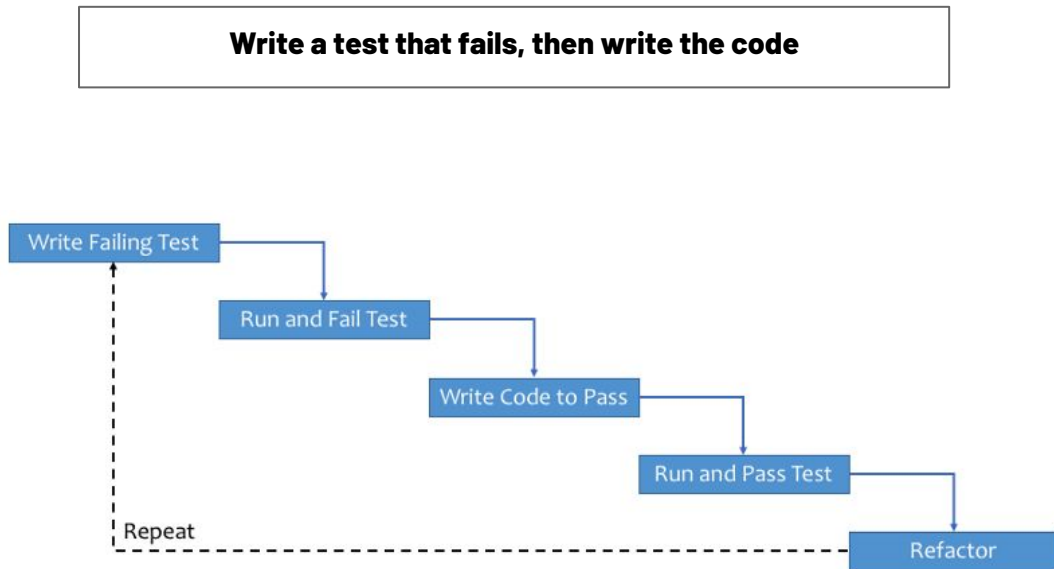
<b>PyTest</b> 	Free software (MIT License)	Stand alone, allows compact test suites.	Unit Testing
<b>unittest</b> 	Free software (MIT License)	Part of Python standard library.	Unit Testing
<b>DocTest</b> 	Free software (MIT License)	Part of Python standard library.	Unit Testing

# TEST DRIVEN DEVELOPMENT

## Σ HOW IT WORKS

- TDD recommends writing tests that would check the functionality of the code **prior** to your writing the actual code.
- Only when you are happy with your tests and the features it tests, you begin to write the actual code
- This process ensures that you carefully plan the code you write in order to pass these tests.

## #CONCEPT

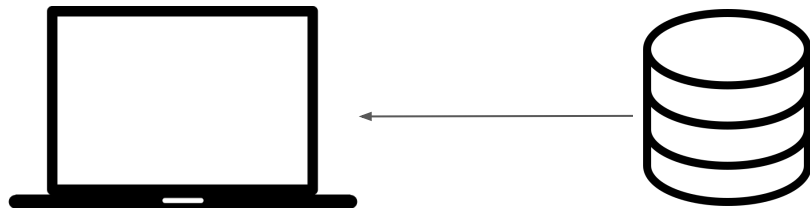


# MOCKING IN PYTHON

## ✂ HOW IT WORKS

- Mock is a 'fake' object used in Python unit testing that allows replacing and mimicking system parts with mock objects.
- We can install **Mock** library to use mocking functionality
- Also we can use *in-built* **unittest.mock** library (most common) to Mock objects,

## #EXAMPLE



**During testing we can mock DB with a 'fake' Mock object to test functionality without calling the actual DB and 'disrupting real data.'**

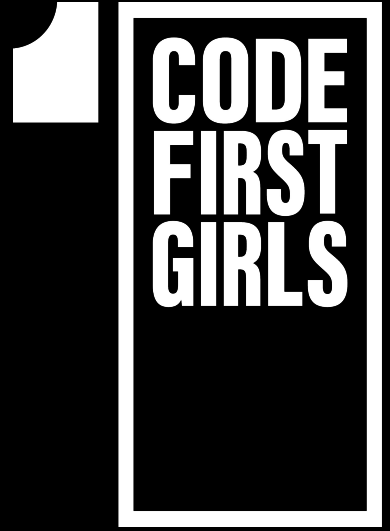
# EXERCISES

## PRACTICE & CODING

WITH YOUR INSTRUCTOR

- **Practice to write unit tests for various functions**
- **Practice to do TDD**
- **Practice to MOCK objects**





**THANK YOU!**