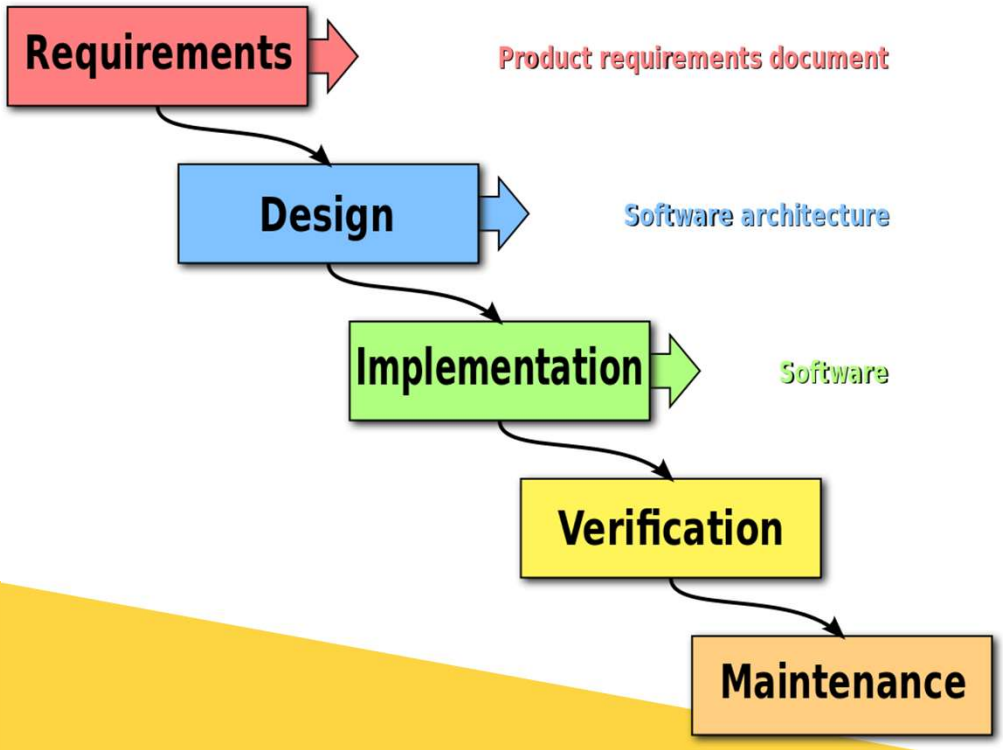
MODULE 1: INTRODUCTION TO PROGRAMMING

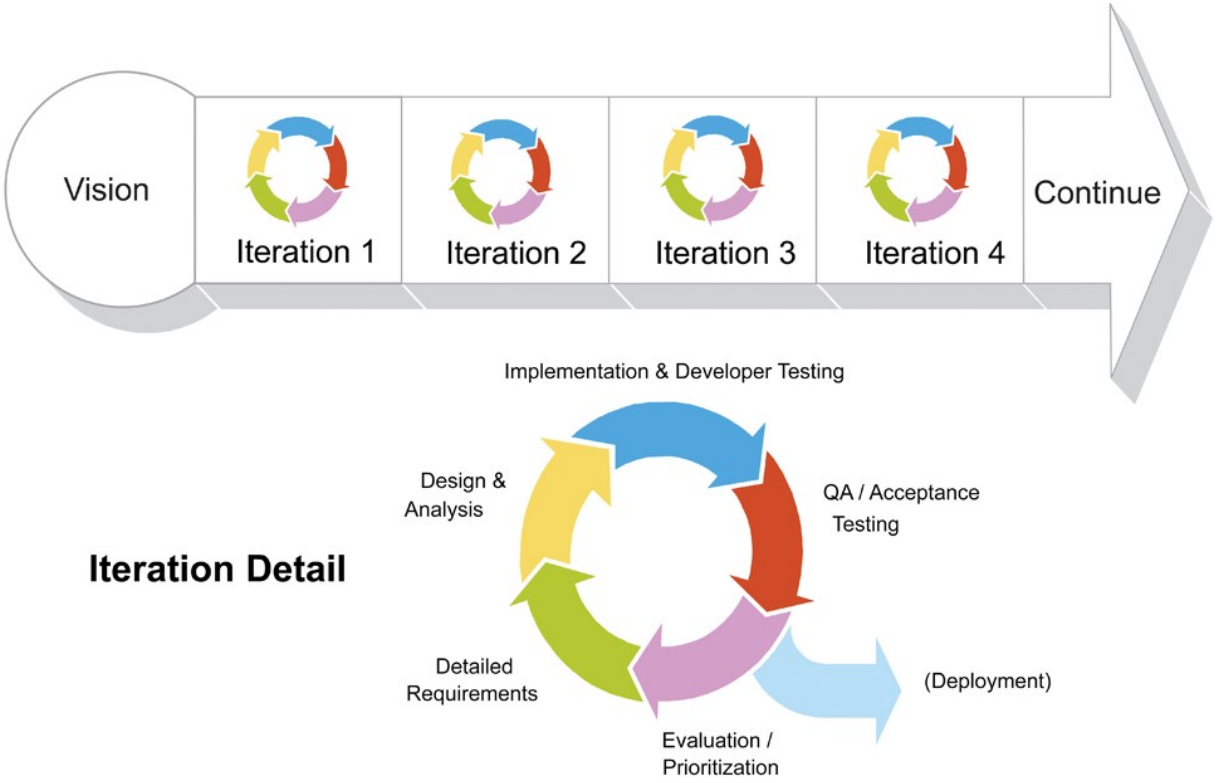
Unit Testing

- *How do we verify that the components of code that we write are correct and that changes don't cause unintended consequences?*

SDLC: Waterfall



SDLC: Agile



SDLC

Waterfall or Agile?

Testing Overview

- Manual Testing
  - Manual tests are no more than the tester using the program as an end user would, and then determining whether or not the program acts appropriately.
- Automated Testing
  - Automated tools run test that repeat predefined actions, comparing a developing program's expected and actual outcomes.

Manual Testing

- Pros:
  - Short-term cost is lower
  - More likely to find real user issues
  - Very flexible
- Cons:
  - Certain tasks are difficult to do manually
  - Not very stimulating
  - Can’t reuse tests

Automated Testing

- Pros:
  - Runs tests quickly and effectively
  - Can be cost effective
  - More interesting
  - Everyone can see results
- Cons:
  - Tools can be expensive
  - Tools still take time
  - Tools have limitations

Testing Motivations

- Exploratory Testing
  - Explores the functionality of the system looking for defects, missing features, or other opportunities for improvement.
- Regression Testing
  - Validates that existing functionality continues to operate as expected.

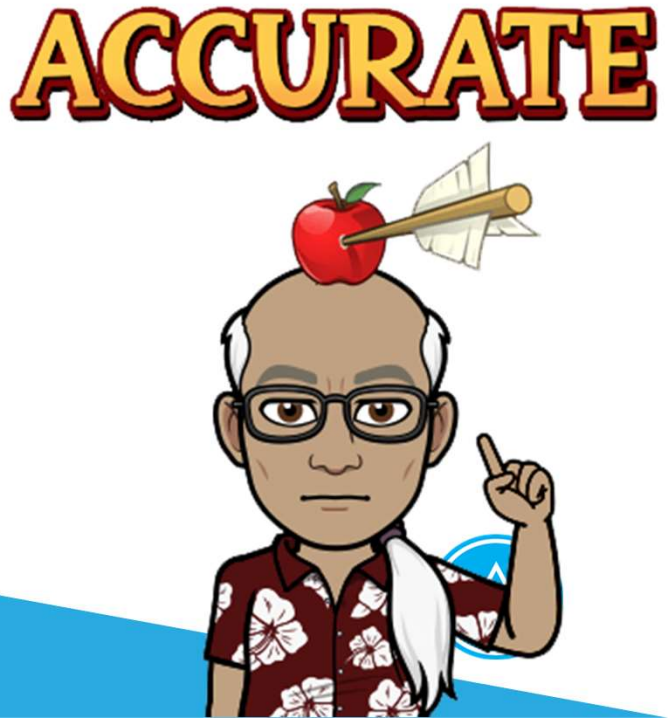
Testing Types

- Unit Testing
  - Low level of testing performed by programmers that validates individual “units” of code function as intended by the programmer.
- Integration Testing
  - Validates the integration between units of code or code and outside dependencies such as databases or network resources.
- Acceptance Testing
  - Performed from the perspective of a user of the system in order to verify that the functionality of the system satisfies user needs.

Testing Overview

Unit Testing -> Integration Testing -> Acceptance Testing:

- longer runtime
- more expensive to write
- harder to troubleshoot



Other Types of Testing

Who Does the Testing?

Dedicated software testers, different skill sets, QA vs. QC

Developers test their own code for correctness

Business people test code for usability and acceptance

Why write unit tests?

ELEVATE YOURSE

Properties of Unit Tests

- Fast - elapsed time of running a unit test should be measured in milliseconds
- Reliable / Repeatable - if a test passed/failed once, it should pass/fail again, assuming no code changes
- Independent - a test should be able to be run independently of other tests and tests should not have interactions with one another
- Obvious - easy to determine why it failed

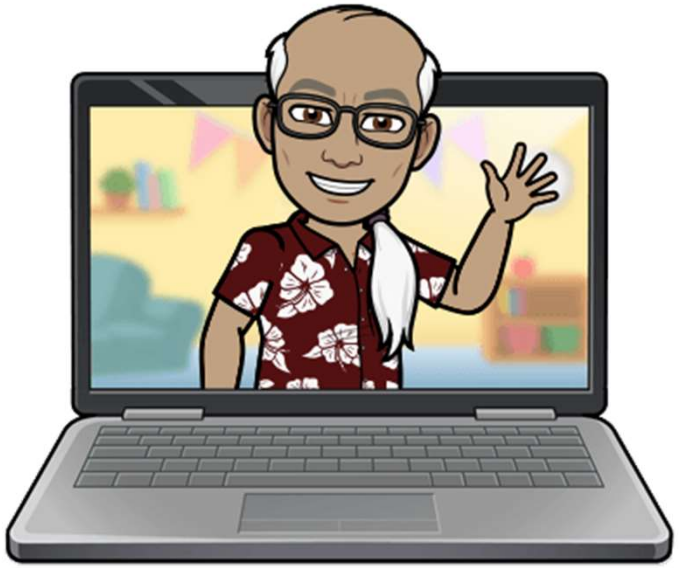
Three Parts to the Test

- **Arrange** - begin by arranging the conditions of the test, such as setting up test data
- **Act** - perform the action of interest, i.e. the thing we're testing
- **Assert** - validate that the expected outcome occurred by means of an assertion (e.g. a certain value was returned, a file exists, etc.)

Unit Test Best Practices

- No external dependencies
- One *logical* assertion per test (i.e. each test should only contain one "concept")
- Test code should be of the same quality as production code
- Test boundary cases
- Test empty arrays, lists, or nulls
- A Test class per class file

LET’S CODE!



WHAT QUESTIONS DO YOU HAVE?

Reading for Monday:

Exception handling

File I/O: Reading Files