

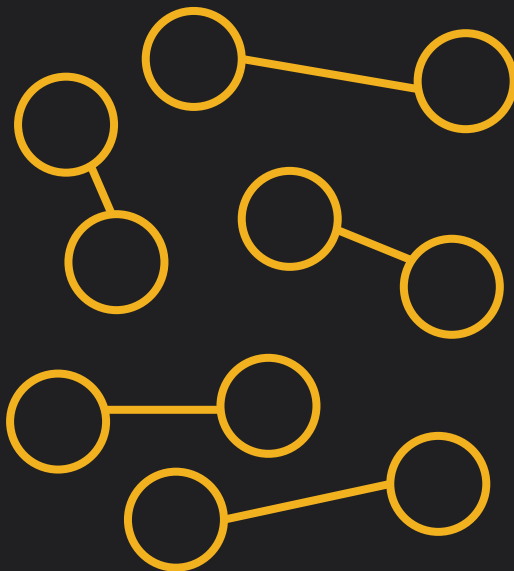
Unit Testing

CS 272 Software Development

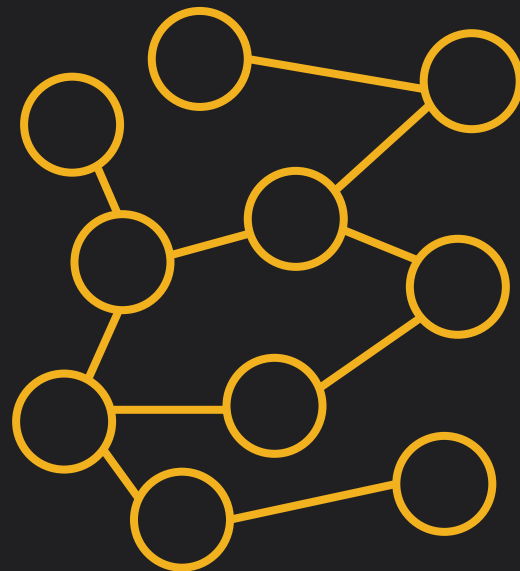
Unit Testing



Integration Testing



System Testing



Unit Testing

Who: Performed by **developers**

What: Improves software **correctness**

When: Usually part of **continuous testing** and/or **test-driven development**

Where: Tests **individual** methods

Why: Tests for **verification** of implementation

How: Tests are **automatable** using different frameworks



JUnit Framework

- Used for primarily for **unit testing**
- Built into Eclipse IDE (and most IDEs)
- Uses custom assertion methods (not **assert** statements)
- Latest versions (v4+) uses **annotations**
 - `@Test`, `@BeforeEach`, `@AfterEach`

<https://junit.org/junit5/>



JUnit Assertion Methods

- `assertFalse()`
- `assertNotNull()`
- `assertNotSame()`
- `assertArrayEquals()`
- `assertTrue()`
- `assertNull()`
- `assertSame()`
- `assertEquals()`

<https://junit.org/junit5/docs/current/user-guide/#writing-tests-assertions>
<https://junit.org/junit5/docs/current/api/org/junit/jupiter/api/Assertions.html>



JUnit Annotations

- Use **@BeforeEach** for code called before every test
 - Used to setup environment before test
- Use **@Test** to indicate a method is a unit test
 - Should contain one assertion method*
- Use **@AfterEach** for code called after every test
 - Used to clean up environment after test

<https://junit.org/junit5/docs/current/user-guide/#writing-tests-annotations>



JUnit @Test Annotations

- Many related test annotations
 - @Test, @RepeatedTest, @ParameterizedTest
- Double check importing right annotations!
 - Similar to annotations in JUnit 4
 - Imports should be from `org.junit.jupiter.api`
- Using `System.exit()` can break unit tests

<https://junit.org/junit5/docs/current/user-guide/#writing-tests>
<https://junit.org/junit5/docs/current/api/org/junit/jupiter/api/Test.html>



Advanced JUnit Tests

- Able to create **parameterized tests**
 - Create single test that expects parameters
 - Create specific list of parameters to test
- Able to **group tests** into nested classes
 - Create multiple nested classes of test cases
 - Can also inherit test cases from other classes

<https://junit.org/junit5/docs/current/user-guide/#writing-tests-parameterized-tests>
<https://junit.org/junit5/docs/current/user-guide/#writing-tests-nested>



Example Normal Test

```
@Test
```

```
public void testNullFlag() {  
    boolean actual = ArgumentParser.isFlag(null);  
    Assertions.assertFalse(actual, "null");  
}
```

<https://github.com/usf-cs272-spring2023/homework-ArgumentParser-template/blob/main/src/test/java/edu/usfca/cs272/ArgumentParserTest.java#L92-L100>



Example Parameterized Test

```
@ParameterizedTest
@ValueSource(strings = { "-a", "-hello", ... })
public void testValidFlags(String flag) {
    boolean actual = ArgumentParser.isFlag(flag);
    Assertions.assertTrue(actual, flag);
}
```

<https://github.com/usf-cs272-spring2023/homework-ArgumentParser-template/blob/main/src/test/java/edu/usfca/cs272/ArgumentParserTest.java#L59-L74>





CHANGE THE WORLD FROM HERE