# ICP-PRG – Agile Programming

## Learning Objectives

1. AUTOMATED TESTS

1.1. Unit Testing

1.1.1. Types of Tests to Automate

1.1.2. Fixture Setup

1.2. Readable Tests

1.2.1. Coding Tests By Intention

1.2.2. Verifying Results

1.3. Test Coverage

1.3.1. Identifying Completeness Conditions

1.3.2. Avoiding Duplication in the Conditions

1.4. Testing Patterns

1.4.1 Listening to Your Tests

1.4.2. Testing the Tests

1.4.3. Refactoring Tests

1.5. Speed of Builds

1.5.1. Test Speed

1.5.2. Test Execution Time

1.6. Test Doubles

1.6.1. Use Test Doubles

1.6.2. Dependency Injection

2. REFACTORING

2.1. Code Smells

2.1.1. Clean Programming

2.1.2. Common Code Smells

2.2. Refactoring with Existing Tests

2.2.1. Principles of Refactoring

2.2.2. Common Refactorings

2.2.3. Refactoring Tools

2.3. Dealing with Legacy Code

2.3.1. Approaching Legacy Code

2.3.2. Refactoring Without Tests

2.3.3. Retrofitting Tests Onto Legacy Code

4. TEST-DRIVEN DEVELOPMENT

4.1. TDD

4.1.1. The Value of TDD

4.1.2. Red-Green-Refactor

4.2. BDD

4.2.1. Identifying Usage Patterns to Define the Object or Function Interface

5. ACCEPTANCE TESTING

5.1. Living Documentation

5.1.1. Tests as Specification and Documentation

5.1.2. ATDD as an Aid to Design Thinking

5.2. ATDD

5.2.1. ATDD Process

5.2.2. ATDD Styles and Tools

6. COLLABORATIVE DEVELOPMENT

6.1. Collective Responsibility

6.1.1. Collective Accountability

6.1.2. Collective Ownership

6.2. Pair Programming

6.2.1. Basics of Pairing

6.2.2. Types of Pairing

7. THE BUILD PROCESS

7.1. Build Tools and Version Control

7.1.1. Build Tools

7.1.2. Version Control

7.2. Continuous Integration

7.2.1. Continuous Integration