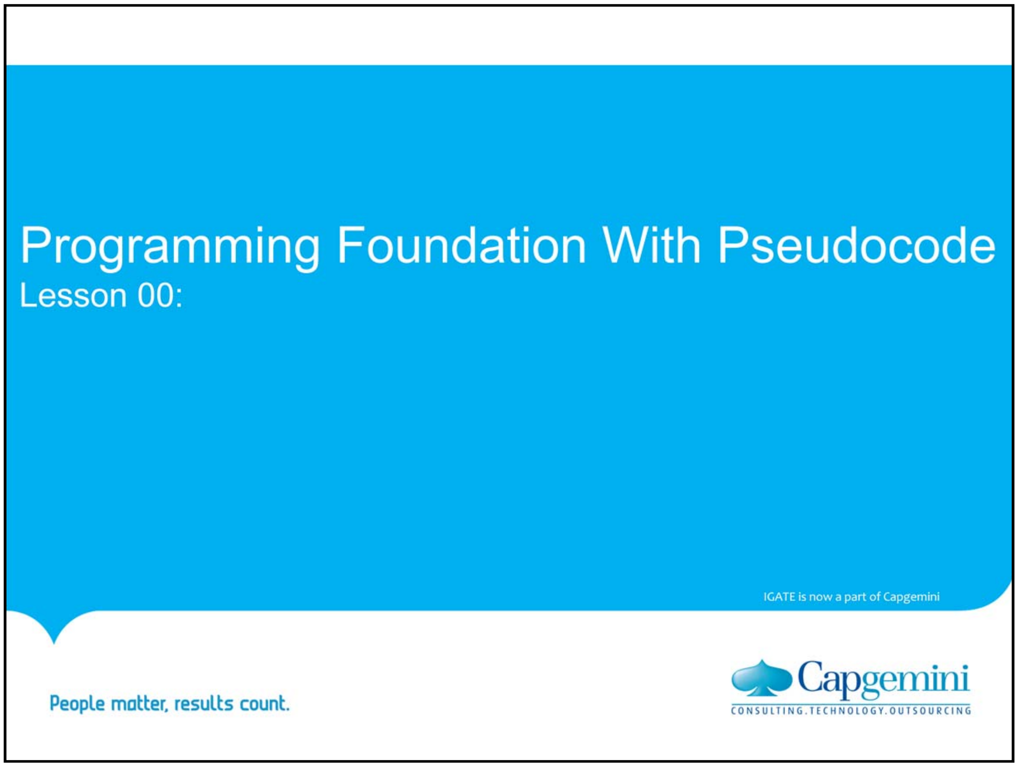


Programming Foundation With Pseudocode

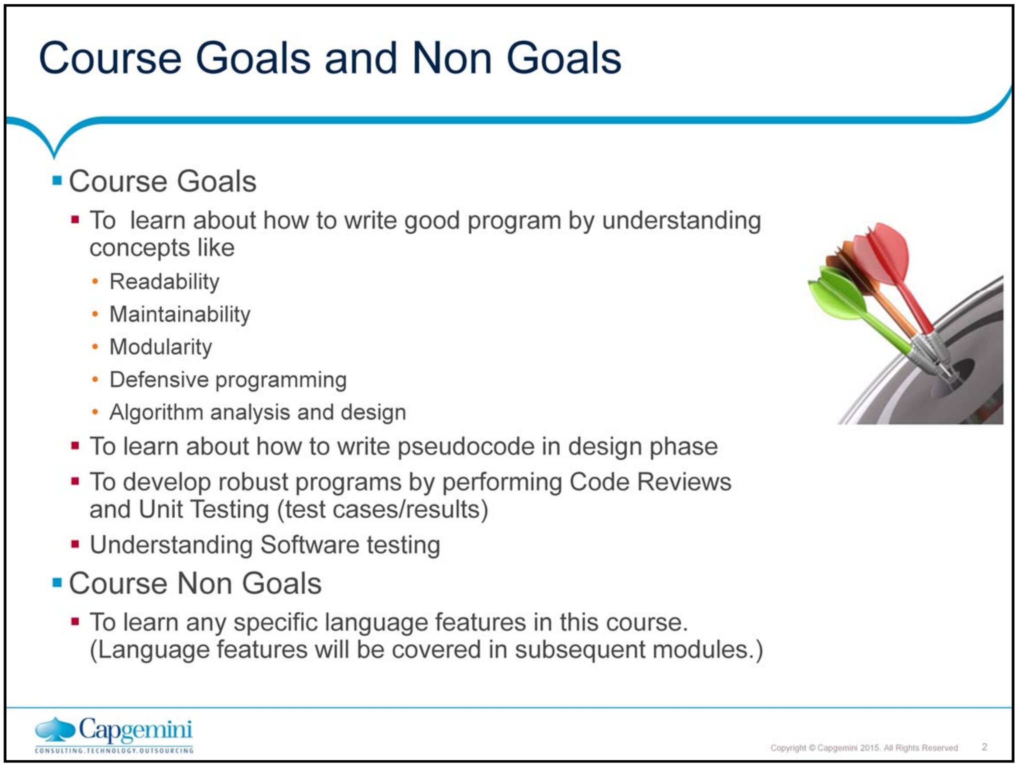
Lesson 00:

IGATE is now a part of Capgemini

People matter, results count.

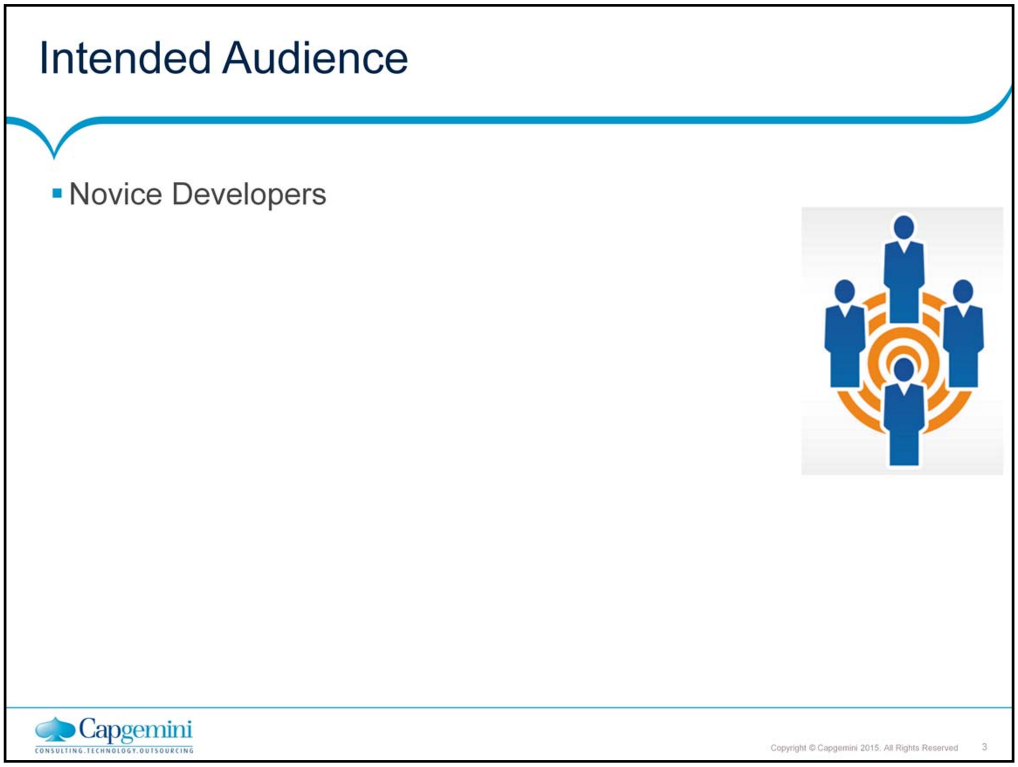
Capgemini

CONSULTING.TECHNOLOGY.OUTSOURCING



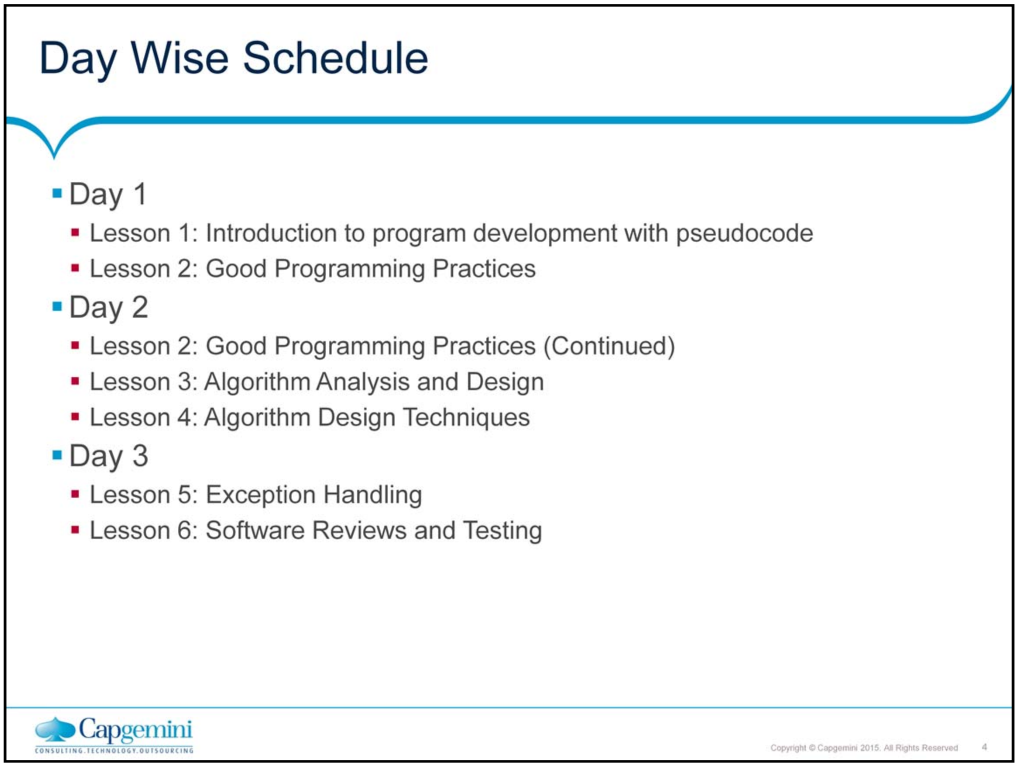
Course Goals and Non Goals

- Course Goals
  - To learn about how to write good program by understanding concepts like
    - Readability
    - Maintainability
    - Modularity
    - Defensive programming
    - Algorithm analysis and design
  - To learn about how to write pseudocode in design phase
  - To develop robust programs by performing Code Reviews and Unit Testing (test cases/results)
  - Understanding Software testing
- Course Non Goals
  - To learn any specific language features in this course. (Language features will be covered in subsequent modules.)



Intended Audience

- Novice Developers



Day Wise Schedule

- Day 1
  - Lesson 1: Introduction to program development with pseudocode
  - Lesson 2: Good Programming Practices
- Day 2
  - Lesson 2: Good Programming Practices (Continued)
  - Lesson 3: Algorithm Analysis and Design
  - Lesson 4: Algorithm Design Techniques
- Day 3
  - Lesson 5: Exception Handling
  - Lesson 6: Software Reviews and Testing

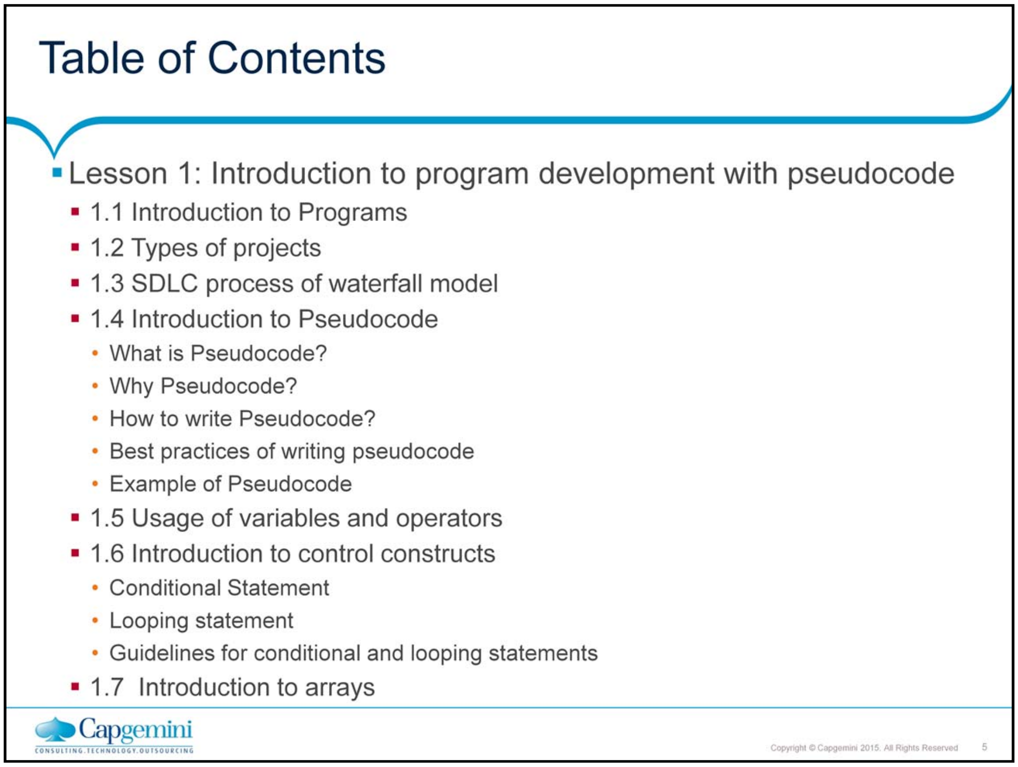


Table of Contents

- Lesson 1: Introduction to program development with pseudocode
  - 1.1 Introduction to Programs
  - 1.2 Types of projects
  - 1.3 SDLC process of waterfall model
  - 1.4 Introduction to Pseudocode
    - What is Pseudocode?
    - Why Pseudocode?
    - How to write Pseudocode?
    - Best practices of writing pseudocode
    - Example of Pseudocode
  - 1.5 Usage of variables and operators
  - 1.6 Introduction to control constructs
    - Conditional Statement
    - Looping statement
    - Guidelines for conditional and looping statements
  - 1.7 Introduction to arrays

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

5

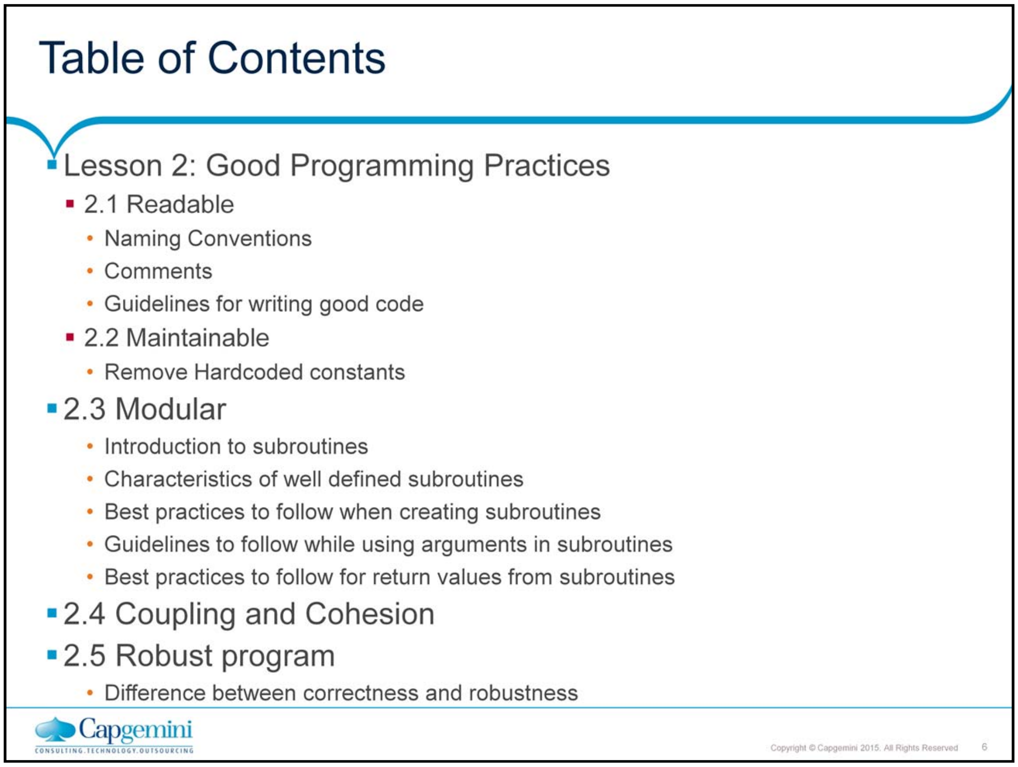


Table of Contents

- Lesson 2: Good Programming Practices
  - 2.1 Readable
    - Naming Conventions
    - Comments
    - Guidelines for writing good code
  - 2.2 Maintainable
    - Remove Hardcoded constants
- 2.3 Modular
    - Introduction to subroutines
    - Characteristics of well defined subroutines
    - Best practices to follow when creating subroutines
    - Guidelines to follow while using arguments in subroutines
    - Best practices to follow for return values from subroutines
- 2.4 Coupling and Cohesion
- 2.5 Robust program
    - Difference between correctness and robustness

6

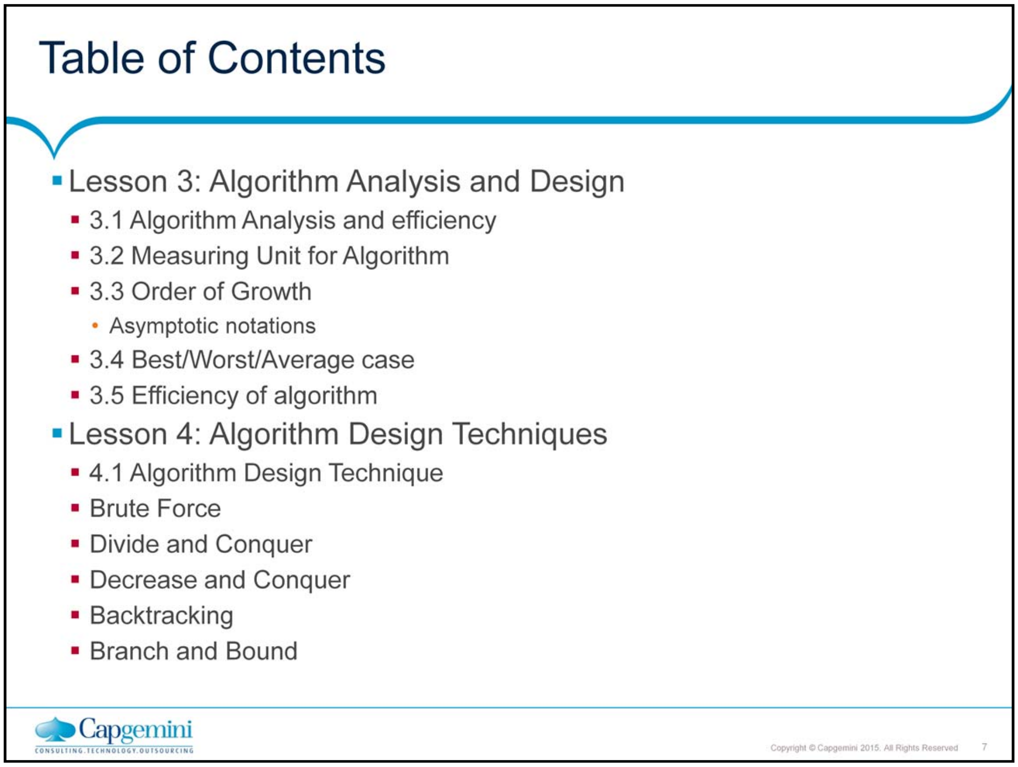


Table of Contents

- Lesson 3: Algorithm Analysis and Design
  - 3.1 Algorithm Analysis and efficiency
  - 3.2 Measuring Unit for Algorithm
  - 3.3 Order of Growth
    - Asymptotic notations
  - 3.4 Best/Worst/Average case
  - 3.5 Efficiency of algorithm
- Lesson 4: Algorithm Design Techniques
  - 4.1 Algorithm Design Technique
  - Brute Force
  - Divide and Conquer
  - Decrease and Conquer
  - Backtracking
  - Branch and Bound

7

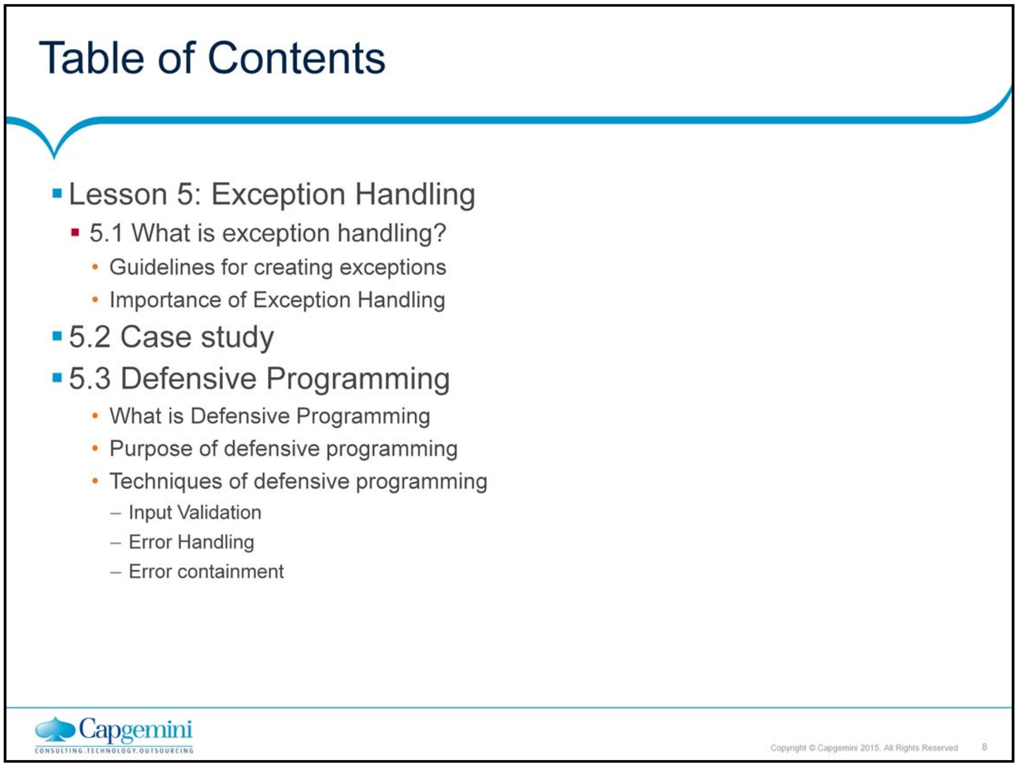


Table of Contents

- Lesson 5: Exception Handling
  - 5.1 What is exception handling?
    - Guidelines for creating exceptions
    - Importance of Exception Handling
- 5.2 Case study
- 5.3 Defensive Programming
    - What is Defensive Programming
    - Purpose of defensive programming
    - Techniques of defensive programming
      - Input Validation
      - Error Handling
      - Error containment

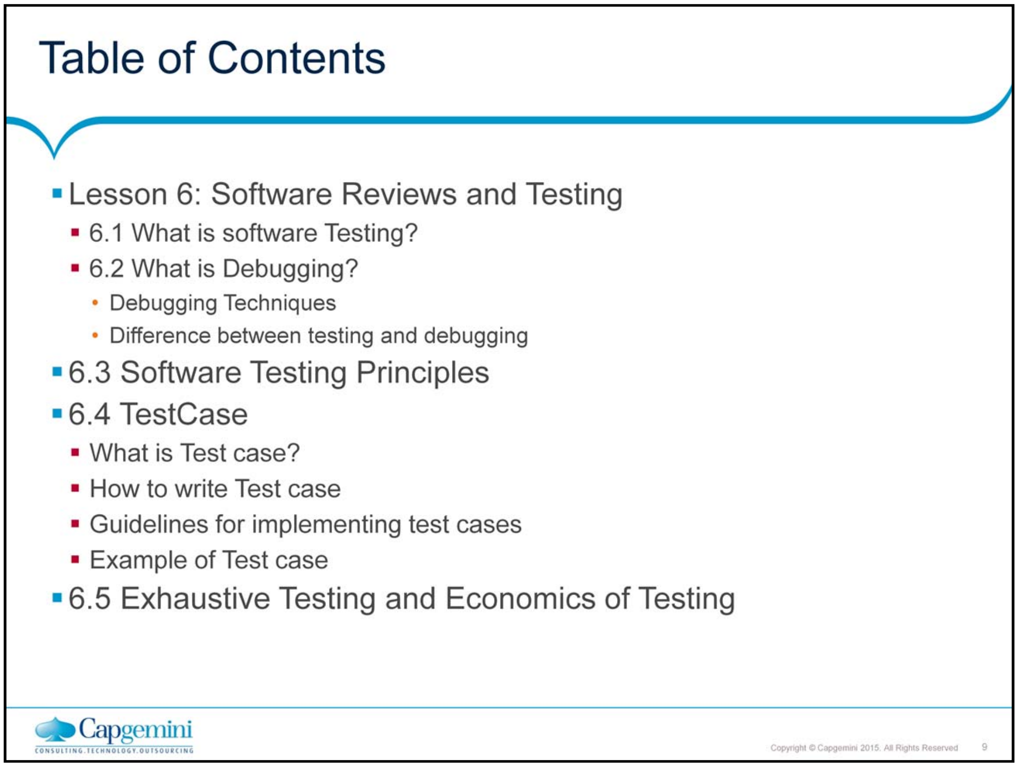


Table of Contents

- Lesson 6: Software Reviews and Testing
  - 6.1 What is software Testing?
  - 6.2 What is Debugging?
    - Debugging Techniques
    - Difference between testing and debugging
- 6.3 Software Testing Principles
- 6.4 TestCase
  - What is Test case?
  - How to write Test case
  - Guidelines for implementing test cases
  - Example of Test case
- 6.5 Exhaustive Testing and Economics of Testing

9

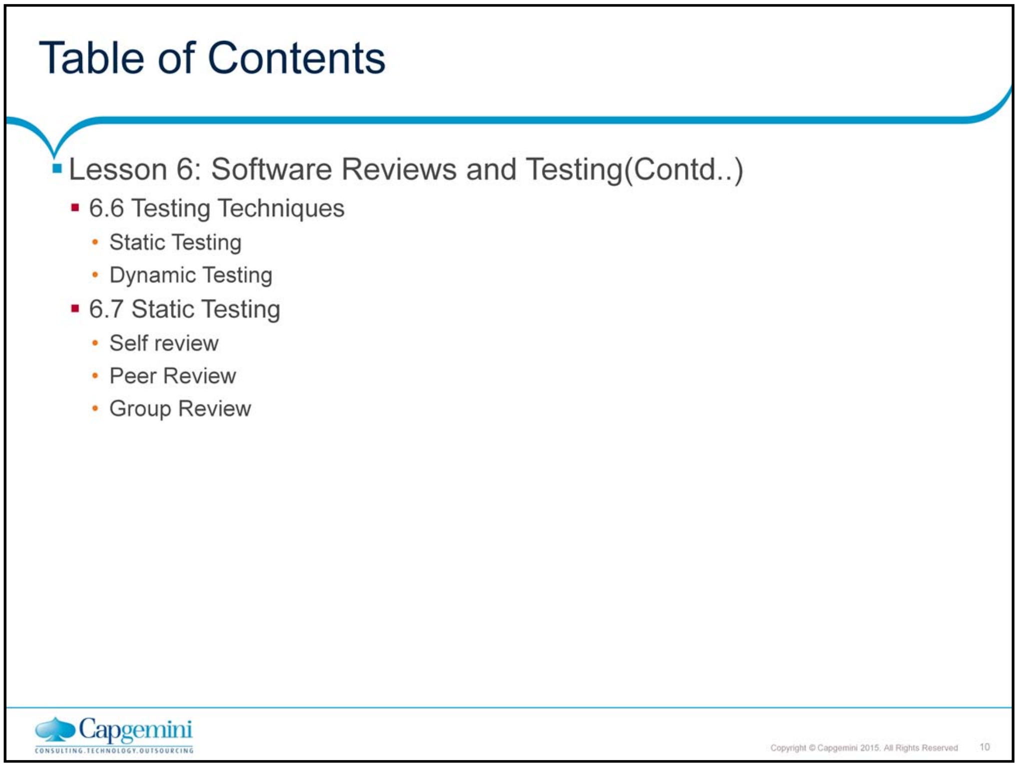


Table of Contents

- Lesson 6: Software Reviews and Testing(Contd..)
  - 6.6 Testing Techniques
    - Static Testing
    - Dynamic Testing
  - 6.7 Static Testing
    - Self review
    - Peer Review
    - Group Review

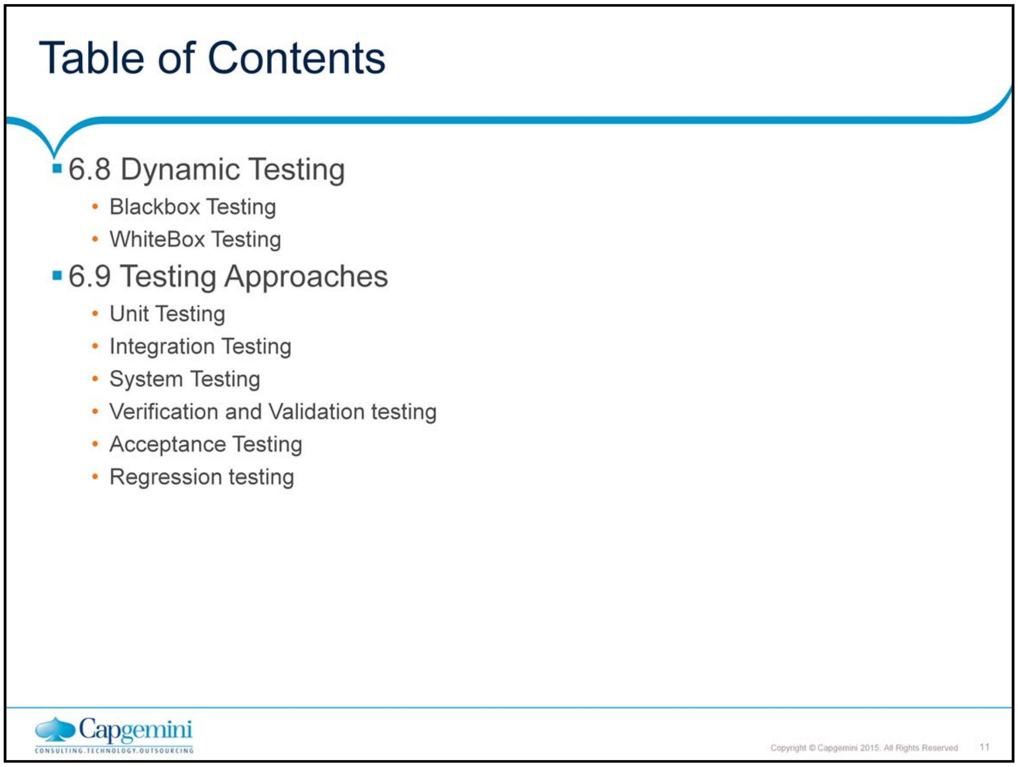
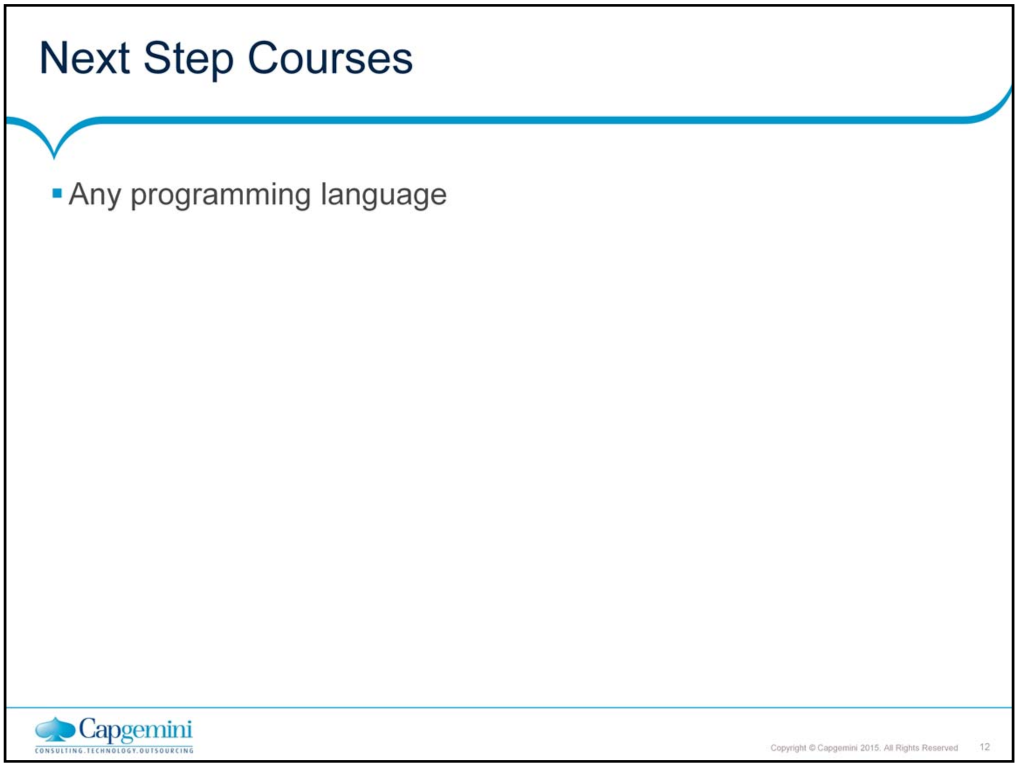


Table of Contents

- 6.8 Dynamic Testing
  - Blackbox Testing
  - WhiteBox Testing
- 6.9 Testing Approaches
  - Unit Testing
  - Integration Testing
  - System Testing
  - Verification and Validation testing
  - Acceptance Testing
  - Regression testing



Next Step Courses

- Any programming language