Course Description

Migrating to OO Programming With Java™ Technology provides students with the necessary concepts and skills to move to Java technology. This course teaches basic object-oriented (OO) concepts and object-oriented analysis and design as they relate to Java technology, as well as introductory Java programming language constructs.

This is not a programming course. It is a preparatory course so that students can learn the necessary background in preparation for taking SL-275: Java Programming Language.

Course Content

Module 1: Objects
- Describe abstraction and how it is used in object orientation
- Identify objects and non-objects from a problem domain
- Describe object encapsulation

Module 2: Classes
- Group objects with similar attributes and common operations in classes
- Explain how classes are used to define objects
- Define inheritance and explain how it relates to software reuse
- Define generalization and specialization and how they relate to inheritance
- Define polymorphism and explain how inheritance promotes polymorphism
- Define abstract classes

Module 3: Using Java Classes
- Code class declarations
- Define the primitive types
- Declare class variables using primitive types
- Declare class variables using reference variables
- Use inheritance correctly
- Use abstract classes
- Use the import statement to include packaged classes in a program
- Use the package statement to group classes in a class hierarchy
- Understand the structure of a Java program
• Name Java classes and other identifiers according to Java programming language guidelines

Module 4: Using Java Methods

• Define class behavior using methods
• Declare a public method for encapsulation
• Pass arguments to a method
• Describe a method's signature
• Explain how method overloading works
• Use a constructor to instantiate an object

Module 5: Object Interaction

• Explain how objects interact with each other through object messaging
• Define association and composition
• Decide whether a relationship between two objects should be association or composition
• Define the lifetime of an object with regard to association and composition
• Define the custody of an object with regard to association and composition

Module 6: Object-Oriented Analysis and Design Using UML

• Create a set of use cases to describe a problem domain
• Create a sequence diagram for a use case
• Create a class diagram for a problem domain
• Create an activity diagram for a use case
• Code class declarations for the class diagram

Appendix A: Java Programming Constructs

• Use casting and promotion
• Apply operators in a statement
• Code looping constructs
• Code an if statement.
• Use switch correctly
• Describe how break affects a loop