Course Description

The Java™ Programming Language course teaches students the syntax of the Java programming language; object-oriented programming with the Java programming language; creating graphical user interfaces (GUI), exceptions, file input/output (I/O), threads and networking. Programmers familiar with object-oriented concepts can learn how to develop Java application. The course uses the Java 2 Software Development Kit (SDK).

Course Content

Module 1: Getting Started

- List the key features of the Java programming language
- Describe the Java virtual machine (JVM™)
- Explain how garbage collection works
- Describe how security features work
- Write a simple Java application, compile and run it

Module 2: Object-Oriented Programming

- Describe the terms class, object, attribute, method, and constructor
- Write code to define a method
- Access the member variables of an object using the dot notation
- Write code to create and initialize an object
- Use the this keyword to access the "current" object
- Use private and public access modifiers
- Write code to invoke a method on a particular object
- Write class constructors and invoke particular constructors using new with arguments
- Understand the use of the package and import statements for library access
- Use the Java Application Programming Interface (API) online documentation

Module 3: Identifiers, Keywords, and Types

- Use comments in a program
- Distinguish between valid and invalid identifiers
- Recognize the keywords in the Java programming language
- List the eight primitive types
• Define literal values for numeric and textual types
• Describe the coding conventions for classes, interfaces, methods, variables, constants, and control structures
• Create a class definition for a simple class containing primitive member variables
• Declare variables of class type
• Describe the significance of a reference variable and state the consequences of assignment between variables of class type

**Module 4: Expressions and Flow Control**

• Distinguish between member and automatic variables
• Describe the initialization of member variables
• Recognize and correct a Possible reference before assignment compiler error
• Recognize, describe, and use operators
• Distinguish between legal and illegal assignments of primitive types
• Recognize Boolean expressions and state the requirement for these in control constructs
• Recognize assignment compatibility and required casts in fundamental types
• Make appropriate use of if, switch, for, while, and do constructions and the labeled forms of break and continue

**Module 5: Arrays**

• Declare and create arrays of primitive, class, or array types
• Explain why and show how to initialize the elements of an array
• Determine the number of elements in any array
• Write code to copy arrays

**Module 6: Class Design**

• Describe encapsulation, polymorphism, and inheritance
• Use subclassing
• Create and use heterogeneous collections
• Create and use methods that accept generic argument types
• Use access control levels
• Invoke overloaded methods and constructors
• Write overriding methods in a subclass and describe execution flow when executing an overridden method
• Invoke overridden methods and describe how the method is chosen
• Invoke overridden constructors
• Control invocation of parent class constructors
• Use wrapper classes
Module 7: Advanced Class Features

- Declare and use static variables and methods
- Declare and use final classes, methods, and variables
- Use abstract methods and interfaces
- Use inner classes

Module 8: Exceptions

- Define exceptions
- Describe the use of the keywords try, catch, and finally
- Describe exception categories
- Identify common exceptions
- Write code to handle your own exceptions

Module 9: Text-Based Applications

- Write code to access command-line arguments and system properties
- Examine and manipulate files and directories
- Read and write text to file streams
- Describe the Collections API
- Use iterators
- Identify deprecated classes and describe how to handle them during migration to Java 2 SDK

Module 10: Building Java GUIs

- Describe the Abstract Windowing Toolkit (AWT) package and its components
- Explain containers, components and layout managers, and how they work together to build a GUI
- Use the flow and border layout managers to achieve a desired dynamic layout
- Use the frame and panel containers
- Place panels inside other containers to build complex layouts

Module 11: GUI Event Handling

- Write code to handle events that occur in a user interface
- Create the appropriate interface and handler method for a variety of event types
- Determine the user action that originated the event from the event object details
- Determine how and when to use the appropriate adapter class to select a subset of event handlers for an event listener

Module 12: GUI-Based Applications
• Identify the key AWT components and the event types that they produce
• Control the colors and font used by an AWT component
• Understand the purpose of the Swing GUI library

Module 13: Threads

• Describe a thread
• Create separate threads, controlling the code and data that are used by that thread
• Control the execution of a thread and write platform-independent code with threads
• Describe some of the difficulties that arise when multiple threads share data
• Use the keyword synchronized to protect data from corruption
• Use wait() and notify() to communicate between threads

Module 14: Advanced I/O Streams

• Use the Streams version of the java.io package
• Construct and use node streams
• Distinguish Readers and Writers from Streams, and select appropriately between them
• Construct and use processing streams
• Understand how to create your own processing stream classes
• Read, write, and update data in random access files
• Use the Serialization interface to encode the state of an object to a stream and to implement object persistence

Module 15: Networking

• Create a minimal TCP/IP client