CORE JAVA

Duration: 5 Days

EVOLUTION OF JAVA

- Evolution of Java and forces that shaped it
- Java Architecture
- Defining a class and an object
- Defining a function
- Writing the first java program
- Language basics
- Primitive data types
- Arrays

INTRODUCTION TO OBJECT ORIENTED PROGRAMMING TECHNIQUES, INHERITANCE

- Implementing Encapsulation and Abstraction
- Introducing access specifiers “private” and “public”
- Understanding constructors
- Use of “this” operator
- Static variables
- Introduction to garbage collection
- Describe Java’s inheritance model and its language syntax
- Demonstration of examples related to inheritance
- Use of “super” keyword
- Defining multilevel hierarchy
- Method overriding
- Implementing Runtime polymorphism
- Abstract classes
- Need for abstract classes
- Examples demonstrating abstract classes
- final methods and final classes
- The cosmic class - Object

INTERFACES, PACKAGES, INSTANCEOF OPERATOR AND FINALIZE METHOD

- Understanding packages
- Need for packages
- How to create packages and store classes in it
- Implementing access control in java using different access specifiers
- Understanding classpath
- Example demonstrating the use of package
- Importing classes from different packages
- Understanding interfaces
- Interfaces as elegant alternative to multiple inheritance
- How to define an interface and how to implement it
- Extending interfaces
- Describe the context in which to use the instanceof operator
- Describe finalization
- Example demonstrating the use of finalize method

EXCEPTION HANDLING

- Describe the exception handling mechanism of Java
- Describe the use of the keywords that comprise Java’s exception handing mechanism
- Checked and Unchecked Exceptions
- Creating user defined exceptions.

UTILITY CLASSES AND INTERFACES

- Describe the need for wrapper classes
- Define wrapper classes
- Describe the context of using the Cloneable interface
- Facilitate the processing of arrays and collections through the Enumeration interface
- Define dynamic arrays in the form of Vector objects

MULTITHREADING

- Define multithreading
- Differentiate between multitasking and multithreading
- Describe Java’s multithreading mechanism in the form of the Thread class and the Runnable interface
- Describe some key methods of the Thread class
- Describe Thread priorities
- Describe race conditions that are likely to occur between threads that are not synchronized
- Define thread synchronization through the use of the synchronized keyword
- Facilitate inter-thread communication with the help of wait( ), notify(), and notifyAll( ) methods
- Examples demonstrating the key concepts of multithreading

I/O STREAMS

- Define input and output streams in Java
- Define Byte streams and Character streams
- Define the predefined stream objects defined in the System class, namely in, out, and err
- Describe the need for stream wrapping or constructor wrapping
- Examples demonstrating the use of the following streams
- BufferedReader, InputStreamReader
- FileReader, FileWriter
- FileInputStream, FileOutputStream
- Define serialization
- Implement object serialization with the help of the ObjectInputStream and the ObjectOutputStream