TOPS Technology

Module 6 :- Java – Core Java

Theory:

1. Introduction of Java

1) History of Java.

Answer:

James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991. Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called *Oak* after an oak tree that stood outside Gosling's office. Later the project went by the name *Green* and was finally renamed *Java*, from Java coffee, a type of coffee from Indonesia. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.

Sun Microsystems released the first public implementation as Java 1.0 in 1996.It promised write once, run anywhere (WORA) functionality, providing no-cost run-times on popular platforms.

2) Features of Java (Platform Independent, Object-Oriented, etc.).

Answer:

Platform-independent

Java is a platform-independent language, which means it can run on multiple platforms.

Object-oriented

Java is an object-oriented language, which means it allows users to create reusable code and modular programs.

Dynamic

Java is dynamic, which means developers can add new classes and methods to existing packages without changing the original code.

Automatic memory management

Java has automatic memory management, which means it automatically allocates new objects and removes unreferenced objects to make room for new ones.

Architecture neutral

Java is architecture neutral because it doesn't have implementation-dependent features, such as the size of primitive types.

Simplicity

Java is designed to be easy to use, and it's simpler to write, compile, debug, and learn than other languages.

Abstraction

In Java, abstraction is the process of hiding data so that only the essential information is shown to the user.

Inheritance

In Java, inheritance is a feature that allows common state and behavior to be abstracted to a super-class and shared by sub-classes.

Polymorphism

In Java, polymorphism is a feature that allows a single task to be performed in different ways.

3) Understanding JVM, JRE, and JDK.

Answer :-

JVM

The Java Virtual Machine is the core of Java and is responsible for executing Java programs. It's platform-independent and ensures that Java source code is agnostic to the platform. The JVM interprets Java bytecode or compiles it into native code.

JRE

The Java Runtime Environment is used to run Java programs. It includes the JVM, class libraries, and other components needed to run Java applications. The JRE is primarily used for running Java applications, not for development.

JDK

The Java Development Kit is a software development kit used to develop Java applications. It includes the JRE, as well as tools for developing, debugging, and monitoring Java code. The JDK is platform-dependent, meaning different platforms require different versions of the JDK.

4) Setting up the Java environment and IDE (e.g., Eclipse, IntelliJ).

Answer :-

1. Download the Eclipse IDE zip file from eclipse.org.
2. Extract the Eclipse IDE zip file into your Program Files folder.
3. Find the eclipse.exe file in the root folder of the extracted download.
4. Double-click on the eclipse.exe file to start the freshly installed Eclipse IDE.
5. Choose a workspace location and start coding

5) Java Program Structure (Packages, Classes, Methods).

Answer :-

- **Packages**: A group of classes and interfaces that are organized together. Packages provide name-space management and are used to group related types.
- **Classes**: Classes is a collection of Data Member and Member Function (Process, Method) with its behaviours.

  Syntax:-
  ```
      Class Classname
      {
            Data member;
            Member Function;
      }
  ```

- **Methods**: Define the behavior of an object by performing specific actions. Methods accept parameters and return values.

===
===
===
===