Java Revision

**Bytecode** is an instruction set designed for efficient execution by a software interpreter. Java uses bytecode in its Java virtaul Machine JVM. A typical instruction set is a computer achitecture and bytecode is an assembly set. Java code is converted to bytecode when compiled and is computer readable. It contains load and store values, arithmetic and logic and other abilities like stack management, type conversion and return invocations.

**Java Virtual machine JVM** is an abstract computing machine that enables a computer to run Java. Three notions of JVM are specification, implementation and instance. The implementation is a program that must meet the requirements of the specification of JVM. The instance is an implementation running a process that executes a program compile into bytecode.

**Typing Discipline** of a language defines the nature of type constraints and how they are evaluated. Characteristics are :

> Type safety ensures that mathematical operations do not recieve strings or arrays

> Optimization of types to handle associated values which enable their storage and usage by algorithms.

> Documentation illustrates the ability of the user to declare their intent of the data type.

> Abstraction is the use of high-level data types which allow the user to think about values as objects and not a collection of bits.

Examples : Static(Java), Dynamic (Prolog), Explicit (SQL) ,

Nominal(C++), Structural(Haskell), Strong (Ruby), weak (PHP)

**Java Runtime Environment** is the list of software tools required for developing Java applications on some computer.

**Javadoc** is the documentation for the Java programming language which contains the API for all of the native Java libraries.

**Jar** is a file format extension which aggregates Java files. The JAR file allows java runtimes to deploy an entire application in a single request. They contain a manifest which describes how to use the associated JAR file.

**Open JDK** or java development kit is open source implementation of the standard edition of Java. Its main components are the JVM, class libraries and javac compiler.

**Java Garbage collection** is memory management built-in to the system which allows developers to create objects without having to worry about memory allocation and de-allocation. The collector automatically reclaims memory for reuse. Memory is marked and wiped through the process of mark-and-sweep. Normal Deletion, another technique moves unused objects and pointers to free space. Garbage collection is generational with older objects more likely to be marked and swept.

**Threading** is the execution of the smallest sequence of instructions managed by a scheduler. Threads exist within processes sharing resources and running simultaneously. In java all programs have at least one thread created by the JVM. The main method is one example of a thread. Creating an interface and extending classes from that is how to establish multi-threading.

**Operator Overloading** is specific case of polymorphism in which different operators have different implementations depending on their arguments. Objects can be performed with arithmetic but not necessarily meant to be arithmetic functions instead having some other mearing. Java doesn't necessarily support user defined operator overloading but supports the handling of strings via concatenation.

**Interfaces** are a reference type in Java. It is similar to a class but it contains abstract functions. Classes implement the interface which then inherit its abstract methods. Interfaces cannot be instantiated as they are abstract and only contain behaviours which a class describes for an object. Interfaces do not contain constructors. Static and final are the only declarative fields allowed in an interface. Interfaces can extend interfaces but not extend from a class. The benefit of using interfaces is associated with refactoring, polymorphism and object encapsulation.

**Classes** are the building blocks of object-oriented design in Java. It can be seen as a blueprint for which objects are created. Classes contain local variables, instance variables and class variables. Locals are contained within methods, constructors and are destroyed once a method has finished. Instance variables are within a class but not inside a method. They are initialised when a class is instantiated and can be accessed from any method or block inside the class. Class variables are held with a static keyword and outside of a method. Constructors create new instances of an object and classes may contain many constructors. Objects follow three rules: declaration; instantiation and initialization inside a class.

**Java Keywords**

* Java Files:
  + Class
  + Interface
  + Enum
* Data Types
  + Byte
  + Short
  + Int
  + Long
  + Float
  + Double
  + Char
  + Boolean
* Control Statements
  + If
  + Else
  + Switch
  + Case
  + While
  + Do
  + For
  + Break
  + Continue
  + Return
  + Goto
* Access Modifiers
  + Private
  + Public
  + Protected
* Modifiers
  + Static
  + Void
  + Final
  + Default
  + Const
  + Abstract
  + Native
  + Transient
  + Volatile
  + Synchronised
  + Strictfp
* Object Representation
  + This
  + Super
  + InstanceOf
* Inheritance Relationship
  + Extends
  + Implements
* Package
  + Package
  + Import
* Exception Handling
  + Try
  + Catch
  + Finally
  + Throw
  + Throws
  + Assert
* Literals
  + Null
  + True
  + False

**Java Applets** are small applications that are written in Java and compile to java bytecode. They are utilised to provide interactive features mainly to web pages which cannot provide functionality by HTML alone. They can provide I/O and graphic display

**Java Servlets** are programs which extend the capability of a server. The implement applications which are hosted on web servers and are Javas comparison to PHP and Asp.NET. Servlets act as a middle layer between requests and the web browser or other HTTP clients and databases or applications on the HTTP server. Inputs can be collected through forms, records from databases. This allows web pages to be created dynamically. Servlets offer increased performance over CGI programs and are platform-independent. All java class libraries are available to the servlets and executed within the address space of the web server. HTTP request data either implicit or explicit is processed by servlets coming to and from a database.

**Java Graphics** is an abstract base class in the Java API which allows an application to draw component objects which are graphical visual objects. Java provides 2D functionality but can be further complicated using Swing which is a GUI widget toolkit for Java and provides more functionality

**Java Swing** is a GUI toolkit that is embedded into the Java libraries. It contains graphics packages for creating Java GUI’s. It is designed to create window-based applications. It is platform-independent and provides a broad range of graphical options for its frames.

**Java Generics** are a type system which extend the usability of the Java type system.They allow a function to operate on multiple objects of various types while ensuring compile time type safety. Classes, interfaces and methods are generic if they declare one or more type variables.

**Java Database Connectivity** JDBC is an API for Java which describes how an application can access a database. The JDBC drivers implement the defined interfaces of the JDBC API for interacting with a relational database. The JDBC drivers translate SQP queries into Java which can be used to manipulate a database.

**Java Web Start is software that** allows a user to run and download Java applications on the web. It eliminates the use of lengthy download and installation procedures and guarantees a one-click latest version of the application.

**@Override** is a command in Java which allows a subclass to implement a parent class method based on its requirements. Overriding in OO programming overrides the functionality of an existing method. It tells the compiler that the method in a subclass is overriding its parent class method. It is beneficial in overriding interfaces methods as to ignore warnings that are displayed at compile-time.