What is Java?

Java is a programming language which possesses object oriented features

**What will I get when I download Java software?**

The Java Runtime Environment (JRE) is what you get when you download Java software. The JRE consists of the Java Virtual Machine (JVM), Java platform core classes, and supporting Java platform libraries. The JRE is the runtime portion of Java software, which is all you need to run it in your Web browser.

**What is Java Plug-in software?**

The Java Plug-in software is a component of the Java Runtime Environment (JRE). The JRE allows applets written in the Java programming language to run inside various browsers. The Java Plug-in software is not a standalone program and cannot be installed separately.

**I have heard the terms Java Virtual Machine and JVM. Is this Java software?**

The Java Virtual Machine is only one aspect of Java software that is involved in web interaction. The Java Virtual Machine is built right into your Java software download, and helps run Java applications.

**Default, Private, Public, Protected**

Visible to the package. the default. No modifiers are needed.

Visible to the class only (private).

Visible to the world (public).

Visible to the package and all subclasses (protected).

**Features of Java**

Object Oriented

Robust

Platform Independent

Multithreading

What is Class?

Class is a blueprint of state and behavior that its objects are going to have.

Object: Object is an instance of class which can exhibit all behaviours defined by its class

Method: is a behavior

**What is Interface?**

An interface is a collection of abstract methods. It sets a contract for the classes which are implementing this interface that it will have those all methods which are defined in interface.

If a class does not perform all the behaviors of the interface, the class must declare itself as abstract

Abstract Class?

Use the **abstract** keyword to declare a class abstract. The keyword appears in the class declaration somewhere before the class keyword.

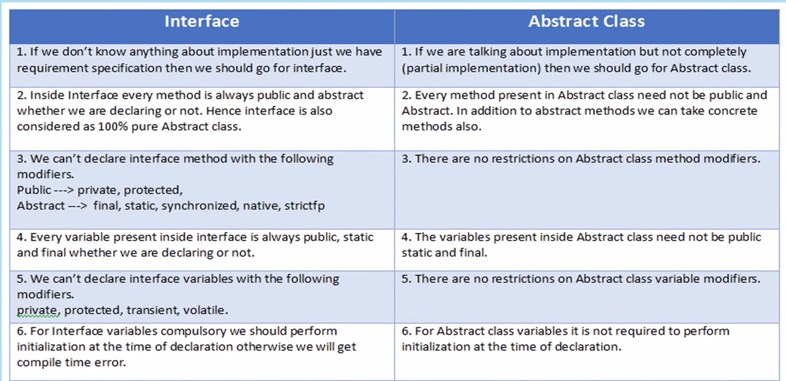
An abstract class is one that cannot be instantiated. All other functionality of the class still exists, and its fields, methods, and constructors are all accessed in the same manner. You just cannot create an instance of the abstract class

If a method of a class is abstract then it is necessary to make the class abstract.

A class inheriting an abstract class must need to implement abstract methods of abstract class.

**Difference between Interface and Abstract Class?**

* Main difference is methods of a Java interface are implicitly abstract and cannot have implementations. A Java abstract class can have instance methods that implements a default behavior.
* Variables declared in a Java interface is by default final. An  abstract class may contain non-final variables.
* Members of a Java interface are public by default. A Java abstract class can have the usual flavors of class members like private, protected, etc..
* Java interface should be implemented using keyword “implements”; A Java abstract class should be extended using keyword “extends”.
* An interface can extend another Java interface only, an abstract class can extend another Java class and implement multiple Java interfaces.
* A Java class can implement multiple interfaces but it can extend only one abstract class.
* Interface is absolutely abstract and cannot be instantiated; A Java abstract class also cannot be instantiated, but can be invoked if a main() exists.
* In comparison with java abstract classes, java interfaces are slow as it requires extra indirection.



**Exception Handling?**

* An exception is a problem that arises during the execution of a program. An exception can occur for many different reasons, including the following:
* A user has entered invalid data.
* A file that needs to be opened cannot be found.
* A network connection has been lost in the middle of communications or the JVM has run out of memory.

Checked Exception: Exception which are checked at compile are called checked exception. All exception which extends exception class (excluding runtime exception are checked exception)

Unchecked exception: which are not checked compile time..these exception extends RuntimeException class.

**Static**

Can be called w/o creating object

One memory is shared by all objects

Static variable is a class variable can’t be declared in a method

Static method can’t invoke instance method or instance variable

Inner nested class can be static class….outer class can’t be static

**Outer class can’t be private or protected or static.**

**Final**

To prohibit object oriented features like inheritance or overriding FINAL is used

If a class is final then no class can inherit it i.e a final class can’t have subclass. However a final class can extend another class i.e final class can be a subclass of another class…concept is no class can inherit a final class.

If a method is final it can’t be overridden.

Final variable behaves like constant variable.

**Garbage Collection**

To release unused memory

**Primitive Datatype**

8 primitive datatypes are there

Byte,short,int,long,float,double,Boolean,char

**Wrapper Class**

As the name says, a wrapper class wraps (encloses) around a data type and gives it an object appearance. Wherever, the data type is required as an object, this object can be used.

int k = 100;  
Integer it1 = new Integer(k);

**This keyword**

This keyword is used to resolve amguity between parameter and instance variable

Int a;

Void add(int a){

This.a = a;

A=a; //will not work

}

This keyword is used call constructor(parameterized or non parameterised)

OOPs Concept

1. Inheritence
2. Encapsulation
3. Polymorphism
   1. Compile Time
   2. Run Time
4. Abstraction
   1. Abstract Class
   2. Interface

**Final Variable**

Final variables need to be initialized only once.

If it is global variable (instance variable) and if constructor is not written then it should be initialized while declaring itself.

If it is global variable (instance variable) and if constructor is written then it could/couldn’t be initialized while declaring itself. But initialization would be once only. Best is to initialize in constructor.

If it is local variable then it could/couldn’t be initialized while declaring itself. But initialization would be once only