**1. Why is java Object Oriented Programming language?**

Java is purely an object-oriented language due to the absence of global scope. Everything in java is an object, all the program codes and data resides within classes and objects. It comes with an extensive set of classes, arranged in packages, object model in java in sample and easy to extend.

**2. Why to use Java ?**

Java is platform-independent. The programming language is structured in such a way that developers can write code anywhere and run it anywhere without worrying about the underlying computer architecture. It is also referred to as write once, run anywhere.

**3. Java Classes ?**

A class is a group of objects which have common properties. It is a template or blueprint from which objects are created. It is a logical entity. It can't be physical.

Everything in Java is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is an object. The car has attributes, such as weight and color, and methods, such as drive and brake.

A Class is like an object constructor, or a "blueprint" for creating objects.

**4. Java objects ?**

Physical instance of a class is called as an object. An object is a real world entity which has state and behavior.

**5. Java constructors ?**

In Java, a constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory.

It is a special type of method which is used to initialize the object.

Every time an object is created using the new() keyword, at least one constructor is called.

**6. Java methods ?**

A method is a block of code or collection of statements or a set of code grouped together to perform a certain task or operation. It is used to achieve the reusability of code. We write a method once and use it many times. We do not require to write code again and again. It also provides the easy modification and readability of code, just by adding or removing a chunk of code. The method is executed only when we call or invoke it.

**7. What is Java ?**

Java is a programming language and a platform. Java is a high level, robust, object-oriented and secure programming language.

**8. Features of Java ?**

Simple

Object-Oriented

Portable

Platform independent

Secured

Robust

Architecture neutral

Interpreted

High Performance

Multithreaded

Distributed

Dynamic

**9. What is JVM ?**

JVM (Java Virtual Machine) is an abstract machine. It is called a virtual machine because it doesn't physically exist. It is a specification that provides a runtime environment in which Java bytecode can be executed. It can also run those programs which are written in other languages and compiled to Java bytecode.

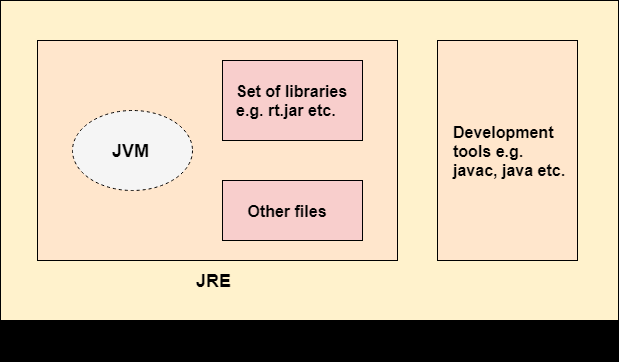
JVMs are available for many hardware and software platforms. JVM, JRE, and JDK are platform dependent because the configuration of each [OS](https://www.javatpoint.com/os-tutorial) is different from each other. However, Java is platform independent. There are three notions of the JVM: *specification*, *implementation*, and *instance*.

The JVM performs the following main tasks:

* Loads code
* Verifies code
* Executes code
* Provides runtime environment

**10. What is JRE ?**

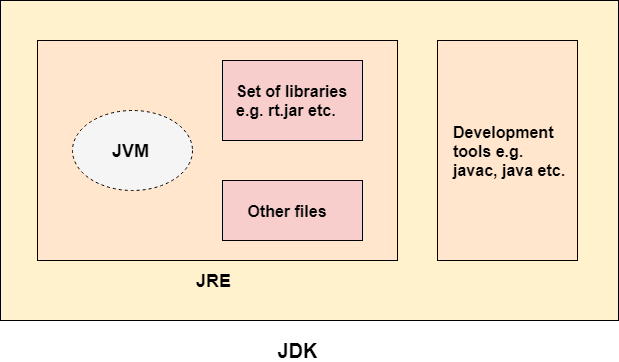
JRE is an acronym for Java Runtime Environment. It is also written as Java RTE. The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.



**11. What is JDK ?**

JDK is an acronym for Java Development Kit. The Java Development Kit (JDK) is a software development environment which is used to develop Java applications and [applets](https://www.javatpoint.com/java-applet). It physically exists. It contains JRE + development tools.

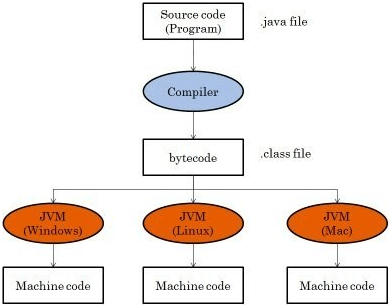
The JDK contains a private Java Virtual Machine (JVM) and a few other resources such as an interpreter/loader (java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc), etc. to complete the development of a Java Application.



**12. Java Byte Code ?**

Java bytecode is the instruction set for the Java Virtual Machine. It acts similar to an assembler. As soon as a java program is compiled, java bytecode is generated. In more apt terms, java bytecode is the machine code in the form of a .class file. With the help of java byte code we achieve platform independence in java.

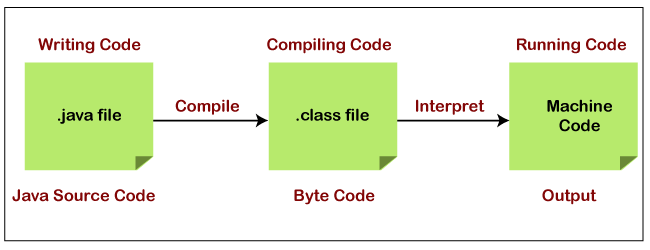
After the first compilation, the bytecode generated is now run by the Java Virtual Machine and not the processor in consideration. This essentially means that we only need to have basic java installation on any platforms that we want to run our code on. Resources required to run the bytecode are made available by the Java Virtual Machine.



**13. Java interpreter ?**

Java is a platform-independent programming language. It means that we can run Java on platforms that have a **Java interpreter**. It is the reason that makes the Java platform-independent. The Java interpreter converts the Java bytecode (.class file) into the code understood by the operating system.

**Java interpreter** is a computer program (system software) that implements the JVM. It is responsible for reading and executing the program. It is designed in such a way that it can read the source program and translate the source code instruction by instruction. **It converts the high-level program into assembly language** (machine language).



**14. Why there is no global scope in java ?**

**Global variables are not technically allowed in Java**. A global variable is one declared at the start of the code and is accessible to all parts of the program. Since Java is object-oriented, everything is part of a class. A static variable can be declared, which can be available to all instances of a class.