Java Interview Questions

Q: What is JRE and why is it required?
Answer: JRE stands for "Java Runtime Environment" which you usually download as Java software. The JRE comprises of the Java Virtual Machine, Java platform classes, and supporting libraries. The JRE is the runtime component of Java software and is all you need to run any Java application.

Q: What is JDK and why is it required?
Answer: The JDK is a superset of the JRE and includes everything that the JRE contains. Additionally, it comes with the compilers and debuggers tools required for developing Java applications.

Q: What is JVM and why is it required?
Answer: JVM stands for The Java Virtual machine. It translates and executes the Java bytecode. It's the entity which transforms Java to become a "portable language" (i.e. write once, run anywhere). Though, each platform has its implementation of JVM like the Windows, Linux, MacOS, etc. have a distinct version of JVM to run bytecode.

Q: Distinguish between the Path and Classpath?
Answer: The <Path> and <Classpath> are OS level environment variables. Path defines the location where the system can look up for the executables (.exe) files, and Classpath specifies the location of the Java class files.

Q: Distinguish between a constructor and method?
Answer: A constructor gets automatically invoked to create an object whereas the method gets called explicitly.

Q: Is it permissible for a constructor to have a different name than its class name in Java?
Answer: No, constructors in Java should have the same name as their classes. If the name is different, then it would behave like a standard method.

Q: Is there any difference between an argument and a parameter?
Answer: While defining methods, you pass variables which you refer as parameters. And when you call these methods and pass values for the variables then they are phrased as arguments.

Q: How your program would behave if you declare the main method as private?
Answer: It would get compiled correctly but will throw the error "Main method not public." at runtime.

Q: What if an application get multiple classes having main() methods?
Answer: It's certainly possible to have multiple main methods in different classes. When you start the application, you've to provide the startup class name for execution. The JVM then looks up for

the main method only in the class whose name you've supplied. Hence, you won't observe any conflict with the multiple classes having the <main()> definition.

Q: What difference you see between pass by reference and pass by value in Java?
Answer: Pass by reference indicates, passing the address itself rather than passing the value. Pass by value means is giving a copy of the value.

Q: What do you understand by Byte Code?
Java compiler generates bytecode for all the Java code and converts into class files. The bytecode is platform independent and needs the platform-specific JVM for the execution.

Q: What do you make of each keyword in public static void main (String args[])?

- Public- <main(..)> is the entry point method which the JVM calls when a program is run. So it's mandatory for it to get accessible from the Java environment. Hence, the access specifier has to be public.
- Static- JVM must be capable of calling this method w/o creating an instance of the class. So the method has to be declared as static.
- Void- <main()> doesn't return anything so it's return type must be void.
- The argument string represents the argument type passed from the console, and the <args> is an array of strings specified at the command line.

Q: How do compare the final, finally and finalize keywords?

- final– It's used to declare a constant.
  - Variables defined in an interface are implicitly final.
  - You can't extend a final class.
- finally– It makes you handle exceptions.
  - It's a keyword used for exception handling. The code under the <finally> block gets executed apparently.
- finalize– It helps in garbage collection.
  - The <finalize()> method is used just before an object is destroyed and garbage collected.

Q: Can you compile a Java class successfully without having the main method?
Yes, we can compile, but it won't run. The <main> method works as the startup function for a Java class, and the JVM calls it for the program execution.

Q: What do you make of System, out and <println> in the function System.out.println()?

- System -> A predefined final class,
- out -> PrintStream object and,
- The <println> -> built-in overloaded method of the out object.

Q: What do you understand by the explicit casting?
It's a process which instructs the complier about transforming the object into a different type.
e.g. long no = 99999;
int new_no = (int) no; // Explicit casting

Q: Would a Java program compile/run if we use <static public void> instead of <public static void>?
Yes, the program will compile and run as usual.

Q: How would you prove that an array is not null but is empty?
Call the <Print array.length>. It will print 0. That suggests that the array is empty. If it would've been null then, it would've thrown a <NullPointerException> on calling the <Print array.length>.

Q: What do you understand of Garbage Collection and how to call it explicitly?
If the object is no longer belong to any variable, Java automatically reclaims the memory. This process is known as garbage collection. You can use the <System.gc()> method to call it explicitly.

Q: How comes an unreachable object become reachable again, is it at all possible?
Yes, an unreachable object may get to reachable state. It can happen if the object <finalize> method gets called during the garbage collection, and there you have set an object making a reference to it. This situation would cause the garbage collection to skip and make the object reachable again.