JVM, Variables in Java

Assignment Solutions

PW SKILLS

1. What is the difference between JVM, JRE and JDK?

Ans: JVM - JVM, which stands for Java Virtual Machine, is a virtual machine that understands and runs java bytecodes.
When you compile a java program, the output is a .class file which contains bytecodes. JVM understands bytecodes and can execute the .class Java files.
There are specific implementations of the JVM for specific platforms - Windows, Linux etc. The same Java .class file can be run for any of the JVM implementations. This is how Java programming language is platform independent and make the Java programs portable i.e write once, run anywhere.
JRE - JRE, which stands for Java Runtime Environment, provides an implementation of the JVM, supporting libraries and other components required to run Java programs. JRE also provides components that enable two kinds of deployment of Java programs. Java plug-in, which enables java programs to run on browsers and Java Web Start, which deploys standalone java applications.
JDK - JDK, which stands for Java Development Kit, contains the JRE plus tools such as compilers, debuggers etc. which are required for developers to develop Java programs.

2. How is Java programming language machine and platform independent?
Ans: When you compile a Java program, the output is a .class file which contains bytecodes that are machine and platform independent. JVM understands the bytecode and runs Java programs. Specific JVMs are implemented for specific platforms. The same .class file can run on any JVM implemented on any platform and machine. So you write a Java program once, compile it once, and run it in any platform.

3. Explain how Java programs are executed by the JVM?
Ans: Java program (.java files) are compiled into bytecodes (.class files) using the Java compiler (javac). A class loader, which is a component of the JVM loads the compiled Java bytecodes into specific areas called runtime data areas. Java execution engine, which is also a component of the JVM executes the Java bytecodes.

4. What is the use of var args ?
Ans: Var args is the feature of Java that allows a method to have variable number of arguments. Var args are used when we are not sure about the number of arguments a method may need. Internally java treats them as array.

Declarations of methods with var args
```
void method(int... x){};
void method(int x, int... y){};
```

5. What are the types of variables in Java?
Ans: There are three types of variables in Java:
- local variable
- instance variable
- static variable

1) Local Variable
A variable declared inside the body of the method is called local variable. You can use this variable only within that method and the other methods in the class aren't even aware that the variable exists.
A local variable cannot be defined with "**static**" keyword.

2) Instance Variable
A variable declared inside the class but outside the body of the method, is called an instance variable. It is not declared as static.

It is called an instance variable because its value is instance-specific and is not shared among instances.

3) Static variable
A variable that is declared as static is called a static variable. It cannot be local. You can create a single copy of the static variable and share it among all the instances of the class. Memory allocation for static variables happens only once when the class is loaded in the memory.

6.How many types of memory areas are allocated by JVM?
Ans: Many types:

Class(Method) Area
Heap
Stack
Program Counter Register
Native Method Stack

7. What is Just-in-time Compiler (JIT)
Ans: JIT is part of the JVM that optimizes the process of converting byte code to machine-specific language. It compiles similar byte codes at the same time and reduces the overall time taken for the compilation of byte code to machine-specific language.