Assignment-01

Q1. What is the difference between Compiler and Interpreter

Answer:-

Compiler

- It is a software which takes sourcecode(HLL) as the input and generates MLL code as the output to convert the HLL code to MLL code compiler will scan the HLL code only once.
- Compiler will speed up the process.
- Compiler in one Scan will identify all the problems in the code(if found).

Interpreter

- It is a software which takes sourcecode(HLL) as the input and generates MLL code as the output to convert the HLL code to MLL code interpreter will scan the HLL code multiple times(depends on the instructions).
- Interpreter will slow down the process.
- Interpreter will do scanning line by line so it takes more time for identifying the problem.

Q2. What is the difference between JDK, JRE, and JVM?

Answer:

JDK: JDK stands for Java Development kit, it provides libraries and the required files to run our java programs

JDK:: JRE + JVM

JRE: Java Runtime Environment, It provides suitable environment to run our java program.

JVM: Java Virtual Machine, It is responsible to run our java programs on the basis of Multithreading.

Q3. How many types of memory areas are allocated by JVM?

Answer:

Java Virtual Machine is a program/software which takes Java bytecode (.class files) and converts the byte code (line by line) into machine understandable code.

JVM has five memory locations namely −

- Heap − Runtime storage allocation for objects (reference types).
- Stack − Storage for local variables and partial results. A stack contains frames and allocates one for each thread. Once a thread gets completed, this frame also gets destroyed. It also plays roles in method invocation and returns.
- PC Registers − Program Counter Registers contains the address of an instruction that JVM is currently executing.
- Execution Engine − It has a virtual processor, interpreter to interpret bytecode instructions one by one and a JIT, just in time compiler.
- Native method stacks − It contains all the native methods used by the application.

Q4. What is JIT compiler?

Answer:

The Just-In-Time (JIT) compiler is a component of the runtime environment that improves the performance of Java applications by compiling bytecodes to native machine code at run time.

Q5. What are the various access specifiers in Java?

Answer:

There are four types of Java access specifiers:

- **Private:** The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
- **Default:** The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.
- **Protected:** The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
- **Public:** The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

Q6. What is a compiler in Java?

Answer:

Compiler is a software which takes sourcecode(HLL) as the input and generates MLL code as the output to convert the HLL code to MLL code compiler will scan the HLL code only once.

Q7. Explain the types of variables in Java?

Answer:

There are three types of variables in java

instance variables

- Variables whose value changes from Object to object is called "instance variables".

- For every object separate copy of instance variables will be created.
- Memory for these variables would be given on the heap area, so jvm would perform initialisation with default values.

static variables

- If the value of the variable changes from object to object then it is not recommended to keep those variables as static, then those type of variables should be kept as static.
- In case of instance variables for every object separate copy will be created but in case of static variables a single copy will be created at class level and shared by every object of that class.
- static variables will get a memory on the method area, it is not required to perform initialisation jvm will perform initialisation.

local variables

- Variables which are created by the programmer to meet the temporary requirements is called "local variables".
- Variables which are a part of method signature, method body those variables are called as "Local variables".
- Memory for those variables will be given on the stack memory, jvm will not initialise any value for those variables, compulsorily the user should initialise the value.

Q8. What are the Datatypes in Java?

Answer:

Data types specify the different sizes and values that can be stored in the variable. There are two types of data types in Java:

Primitive data types: The primitive data types include boolean, char, byte, short, int, long, float and double.

Non-primitive data types: The non-primitive data types include Classes, Interfaces, and Arrays.

Q9. What are the identifiers in java?

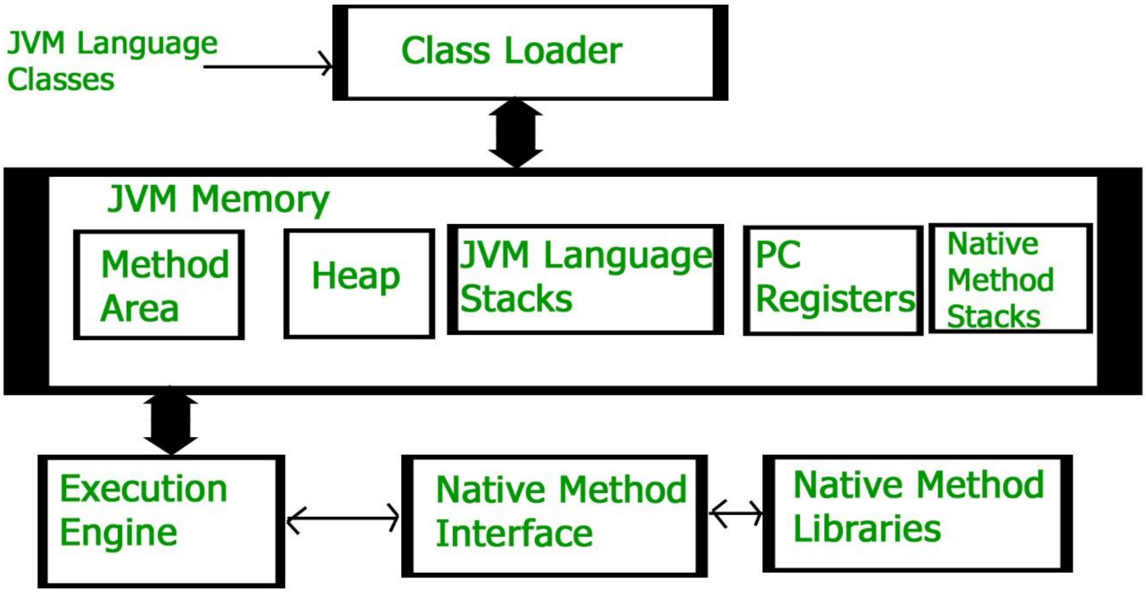
Answer:

- identifier is a name in java program.
- identifier can be className, method name, variable name, label name.
- The only character allowed for java identifiers is

  a to z, A to Z, 0 to 9, $, _

Q10. Explain the architecture of JVM

Answer:



- Classloader − Loads the class file into the JVM.
- Class Area − Storage areas for a class elements structure like fields, method data, code of method etc.
- Heap − Runtime storage allocation for objects.
- Stack − Storage for local variables and partial results. A stack contains frames and allocates one for each thread. Once a thread gets completed, this frame also gets destroyed. It also plays roles in method invocation and returns.
- PC Registers − Program Counter Registers contains the address of an instruction that JVM is currently executing.
- Execution Engine − It has a virtual processor, interpreter to interpret bytecode instructions one by one and a JIT, just in time compiler.
- Native method stack − It contains all the native methods used by the application.