What is JDK, JRE, JVM

The Java Development Kit (**JDK**) is a software development environment used for developing Java applications and applets. The **JRE**, or Java RTE, is developed by Sun Microsystems (the creator of Java) and includes the Java Virtual Machine (JVM), code libraries, and components, which are necessary to run programs written in Java. The Java Virtual Machine (**JVM**) is the runtime engine of the Java Platform, which allows any program written in Java or other language compiled into Java bytecode to run on any computer that has a native **JVM**.

Is JVM, a compiler or interpreter?

Interpreter.

Why Java don't use pointers?

1. **Memory access via pointer arithmetic:** this is fundamentally unsafe. Java has a robust security model and disallows pointer arithmetic for the same reason. It would be impossible for the **Virtual Machine** to ensure that code containing pointer arithmetic is safe without expensive runtime checks.

2. **Security:** By not allowing pointers, Java effectively provides another level of **abstraction** to the developer. No pointer support makes Java more secure because they point to memory location or used for **memory management** that loses the security as we use them directly.

3. **Passing argument by reference:** Passing a reference which allows you to change the value of a variable in the **caller's** scope. Java doesn't have this, but it's a pretty rare use case and can easily be done in other ways. This is in general equivalent to changing a field in an object scope that both the caller and **callee** can see.

4. **Manual memory management:** you can use pointers to manually control and allocate **blocks of memory**. This is useful for some bigger applications like games, device drivers etc. but for general-purpose **Object-Oriented programming** it is simply not worth the effort. Java instead provides very good automatic **Garbage Collection** (GC) which takes care of memory management.

What are various types of Class loaders used by JVM?

Bootstrap, Extension, System  
 How are classes loaded by JVM?

**An application or system class loader loads our own files in the classpath.** **Extension class loaders load classes that are an extension of the standard core Java classes.**  **A bootstrap or primordial class loader is the parent of all the others.**  
Which memory areas does instance and static variables use?

The PermGen or Metaspace space of Heap Memory.  
 What is PermGen or Permanent Generation?

**PermGen** is memory allocation before Java 7 in JVM. the **PermGen** will store the bycode of class and method and loaded ad beginning of JVM started. all the jar in the classpath will be loaded into memory in JVM that allocated and name as **PermGen**.  
What is metaspace ?

Metaspace is a new memory space – starting from the Java 8 version; **it has replaced the older PermGen memory space**. The most significant difference is how it handles the memory allocation.

Describe what happens when an object is created in Java.

* Memory is allocated from heap to hold all instance variables and implementation-specific data of the object and its super classes. Implementation-specific data includes pointers to class and method data.
* The instance variables of the objects are initialized to their default values.
* The constructor for the most derived class is invoked. The first thing a constructor does is call the constructor for its super classes. This process continues until the constructor for java.lang.Object is called, as java.lang.Object is the base class for all objects in java. Before the body of the constructor is executed, all instance variable initializers and initialization blocks are executed.
* Then the body of the constructor is executed. Thus, the constructor for the base class completes first and constructor for the most derived class completes last.

Different types of memory used by JVM?

Heap **Memory**, which is the storage for Java objects.

Non-Heap **Memory**, which is **used** by Java to store loaded classes and other meta-data.

**JVM** code itself, **JVM** internal structures, loaded profiler agent code and data, etc.

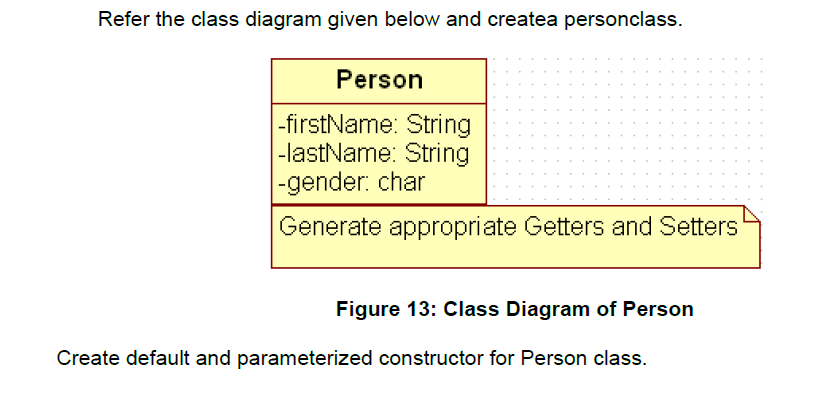
Does Java Pass by Value or Pass by Reference

**Java does** manipulate objects by **reference**, and all object variables are references. However, **Java** doesn't **pass** method arguments by **reference**; it **passes** them by **value**.

What are native methods?

A **native method** is a **Java method** (either an instance **method** or a class **method**) whose implementation is written in another programming language such as C.

Lab work:



Also Create “PersonMain.java” program and write code for following operations:

a) Create an object of Person class and specify person details through constructor.

b): Modify to accept phone number of a person. Create a new method to implement the same and also define method for displaying persondetails.

c): Modify the above program, to accept only ‘M’ or ‘F’ as gender field values. Use Enumeration for implementing the same

--After spring--

d) Add a method called calculateAge which should accept person’s date of birth and calculate age of a person.

e) Add a method called getFullName(String firstName, String lastName) which should return full name of a person

f) Display person details with age and fullname.

.

|  |
| --- |
| Write a program to accept a number from user as a command line argument and check whether the given number is positive or negative number. |
|  |
|  |