LB Assignment 02

Java Basics

PAGE
DATE

Q.1- What is diff. bet^n JDK, JRE & JVM?

→ Java Development kit (JDK) is a software development environment which is used to develop java applications & applets. It physically exits. It contains JRE + development tools.
JDK contains private Java Virtual Machine (JVM) & few other resources such as interpreter / loader, compiler, an archiver, documentation generator to compile the development of a java application.

Java Runtime Environment (JRE) is a set of software tools which are used for developing java applications. It is used to provide runtime environment. It is implimentation of JVM. It physically exits. It contains set of libraries & other files that JVM uses at runtime.

Java Virtual Machine (JVM) is an abstract machine. It doesn't physically exits. It is a specification that provides a runtime environment in which Java byte code can be executed. It can also run those programs which are written in other languages & compiled to Java byte code.

JVM are available for many hardware & software platforms. JVM, JRE & JDK are platform dependent because the configuration of each OS is different from each other.
However, Java is platform independent.

Q.2 What is JIT compiler?

→ Just-In-Time (JIT) compiler is a component of runtime environment that improves the

performance of Java applications by compiling byte codes to native machine code at run time. JIT has following Advantages

- It requires less memory usages.
- The code optimization is done at run time.
- It reduces the page fault.

Q-3 What is class loader?

→ Java class loader is part of JRE that dynamically loads Java classes into JVM. Java runtime system does not need to know about files & file systems because of classloaders. Java classes aren't loaded into memory all at once, but when required by an application. Here Java Classloader is called JRE and these classloaders load classes into memory dynamically

Types of Java classloaders –

1. Bootstrap classloader

A machine code which kickstarts the operation when JVM calls it. It is not Java class. Its job is to load first pure java classloader. It loads classes from location rt.jar. It is also called as Primordial classloader.

2. Extension Classloader

It is child of Bootstrap classloader and loads extensions of core java classes from the respective JDK extension library. It loads files from jre/lib/ext directory or any other directory pointed by system property java.ext.dirs.

3. System classloader

It loads application type classes found

in environment variable CLASSPATH, -classpath or -cp command line option. It is child class of Extension classloader.

Q-4 Explain various memory logical partitions.

→ A logical partition (LPAR) is a division of computer's processor's memory and storage into multiple sets of resources so that each set of resources can be operated independently with its own operating system instance and applications. The number of logical partitions that can be created depends upon system's processor model & resources available.

Typically partitions are used for different purposes such as database operation or client/server operation or to ~~test~~ seperate test and production environment.

PAGE
DATE

Q-5 What gives Java its "Write once & run anywhere nature".

→ One of the initial "killer feature" of Java was supposed to be the write once, run anywhere nature of it. Earlier it is not practically possible to have different versions of an application for different devices because the devices having variety of CPU's, operating system & browsers. The same code must run on all the computers therefore we need a portable code.

Portability refers to the ability to run a program on different machines. "Java is portable", means that you can run Java bytecode on any hardware that has a compliant JVM.

The Java compiler compiles a Java program (.java file) and converts it into class file (.class) that contains bytecodes, which is intermediate language betn source code & machine code. These bytecodes are not platform specific, so with the help of JVM, Java program can run on wide variety of platforms. JVM is platform dependent i.e. its implimentation differs from platform to platform but these all JVM can execute same Java bytecode.

Q-6 Explain History of Java. Who invented Java?

→ Java was originally designed for interactive television, but it was too advanced technology for digital cable television industry at that time. Java team members (Green Team), initiated this

project to develop language for digital devices such as set top boxes, T.V, etc. However it was best suited for internet programming. Later, Java technology was incorporated by Netscape.

The principles for creating Java programming were "Simple, Robust, Portable, Platform-independent, secured, High performance, Multithreaded, Architecture Neutral, Object oriented, Interpreted & Dynamic". Java was developed by James Gosling, who is known as father of Java, in 1995. James G. & his teams started project in 1990's.
Currently Java is used in internet programming, mobile devices, games, e-business solutions, etc.

Firstly, it was called "Greentalk" then "Oak" was developed as part of Green project.
In 1995 Oak was renamed as "Java".
Java is an island in Indonesia where first coffee was produced (Java coffee). Java is just a name not acronym.
Initially developed by James G. at Sun Microsystems (which is now subsidiary of Oracle Corp.).

Q-7 What was original name of Java? Why it was renamed?

→ Firstly it was called "Greentalk" by James Gosling file extension was .gt.
After that it was called "Oak" as a symbol of strength. In 1995 "Oak" was renamed as "Java" because it was already taken by Oak Tech.
The suggested word were "dynamic", "revolutionary"

silk, jolt, DNA, etc. They wanted something that reflects the essence of technology: revolutionary, dynamic, lively, cool, unique, easy to spell & fun to say.

Q-8 List features of Java.

→ The primary objective of Java Programming language creation was to make it portable, simple & secure programming language. The features of Java are also known as Java buzzwords.

- Simple
- Object-oriented
- Portable
- Platform independent.
- Secured
- robust
- Architecture neutral
- Interpreted
- High performance
- Multithreaded
- Distributed
- Dynamic

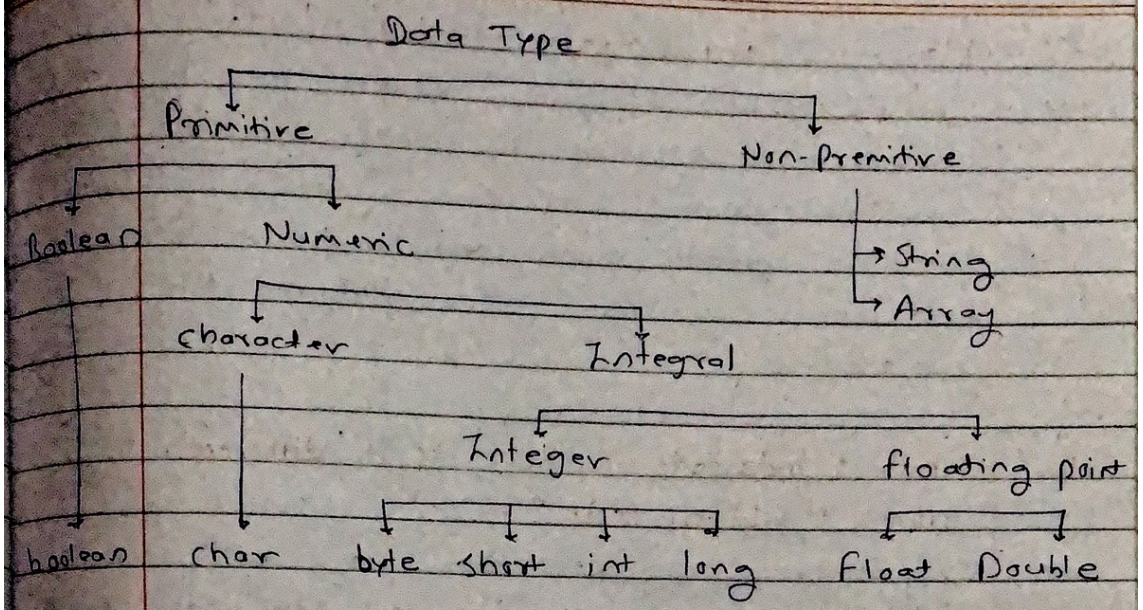
Q-9 List various data types in Java.

1. Java Primitive data types

They are building blocks of data manipulation

- boolean (1 bit)
- byte (1byte)
- char (2byte)
- short (2byte)
- int (4byte)
- long (8byte)
- float (4byte)
- double (8byte)

PAGE
DATE



Q-10 What is difference bet^n System.out.print, System.out.println, System.~~out~~.err.print.

→ System.out is a print stream to which we can write characters. It gives outputs the data we write to it on command Line interface console/terminal. mostly used for console applications / programs to display result to user. System.err is also a print stream. It work same as system.out. It is mostly used to output error texts.

System.out.println()	System.~~out~~.err.println()
- will print Standard output of system	- will print to the standard error.
- mostly used to display result on console	- used to display output error texts.
- default black colour	- most IDE's give it a red colour to differentiate.

PAGE
DATE

Q-11 How is Java Platform independent?

→ Java provide platform independence by making use of Java byte code or .class file is generated during compilation of code. This byte code is platform independent and can run on any system regardless of the platform it is built upon.

Q-12 What is byte code? How is it different from machine code?

→ Byte code is an intermediate code bet^n source code & machine code. It is low level code obtained after compilation of source code which is written in high-level language. It is processed by a virtual machine like (JVM) Java Virtual Machine.

Byte code is a non-runnable code after it is translated by an interpreter into Machine code then it is understandable by machine. It is compiled to run on ~~Jav~~ JVM, any system having JVM can run it irrespective of their operating system. That's why Java is platform-independent. Byte code is reffered to as a portable code.

Machine Code - It is a set of instructions that is directly machine understandable & it is processed by Central Processing Unit (CPU). Machine code is in binary (0's & 1's) format which is completely different than byte code and source code. It is most lowest-level representation of source code. It is obtained after compilation or interpretation.

PAGE
DATE

Q-13 What is difference bet^ Jar file & Runnable jar file ?

→ In simple terms JAR file is a Java application which requires a command line to run & a runnable jar file can be directly executed by double clicking it.

A JAR (Java Archive) is a package file format typically used to aggregate many Java class files & associated metadata & resources (text, images) into one file to distribute application software or libraries on Java platform.

In simple words Jar file is a file that contains a compressed version of .class files, audio files, image files or directories. We can imagine a .jar file as a zipped file (.zip) that is created by using WinZip software. Even, WinZip can be used to extract contains of a .jar file. So you can use them for tasks such as lossless data compression, archiving, decompression & archive unpacking.

A runnable jar file allows a use to run Java classes without having to know class names & type them in a command prompt, rather the user can just double click on jar file & the program will fire up.

A runnable jar allows Java classes to be loaded just like when a user clicks an exe file.

PAGE
DATE

Q-14 What is difference bet^n Runnable jar file & exe file?

→ Jar file are like dead body, .exe file are like living being.
Jar file is the combination of compiled java classes.
Executable file (.jar) is also a combination of java classes with Main class.

Q-15 How is "C" platform dependent language?

→ "C" is portable programming language, Because it is not tied to any hardware or system. It is a hardware independent language.
C programs does not depend on platforms. But the executable file that is generated at the end may depend on a platform.

C programs have ".c" as their extension. When you compile that program Compiler forms a object file ".o". When you run, it generates executable file. The extension & type of this executable file depends on compiler you are using, and on which OS you are using particular compiler

e.g. you are writing code on Windows PC & using code Blocks or an IDE with GCC GNU compiler. Now when you compile & run that code, the file that actually running will be of type ".exe". The exe file on Windows is executable file, which gets executed.

Now similarly when you other OS you get

other extension for executable files. for Mac in C after compiling you will get file with extension ".out". This file is generated when you compile it using the terminal of macOS or any other Unix terminal. But in case of Linux, there is no any exact standard type of extension that you can say its an extension of all executable file

When you say 'C' programs are platform dependent then its not correct. The C program alone with their extension as '.c' formed can be copied and used on any platforms, whether windows, Mac or Linux. But the executable file that particular OS formed at end is dependent on platform. The compiler for that particular OS is responsible for that