Assignment-2

Q1) what is the difference betⁿ JDK, JRE & JVM?

=>

JDK	JRE	JVM
1) JDK stands for 'Java Development Kit'.	1) JRE stands for 'Java Run time Environment.'	1) JVM stands for 'JVM virtual Machine'.
2) It is often called as 'Superset of JRE'	2) It is a set of software tools. responsible of Java for execution of Java program or applicatⁿ.	2) JVM loads, verify and execute Java bytecode
3) It is the foundational component that enables 'Java Applicatⁿ & 'Java applet developmen objects	3) It uses heap space for dynamic memory allocation for Java objects	3) It is known as 'Interpreter'.
4) JDK contains all the tools required to compile, debug & run a program developed using java platform.	4) It uses heap space for dynamic memory allocation for Java objects. 4) JRE is composed. of a varity of other supporting. software tools & features to get the most out of Java application.	4) It is specially responsible for converting bytecode to machine specific code and is. necessary in both JDK & JRE. 5) It is also platform-dependent.

Q2) What is JIT Compiler?

→ 1) JIT is an 'integral part of JVM' (Java-Inline).
2) It is a long-running, computer-intensive program that provides the best environment performance environment
3) It optimizes the performance of the java application at compile or run time.

Advantages:-
1) It requires less memory usages.
2) The code optimization is done at run-time
3) It uses different level of optimization.
4) It reduces the page faults.

Disadvantages:-
1) It increase the complexity of program.
2) The program with less line of code doesn't take the benefit of the JIT compiler.
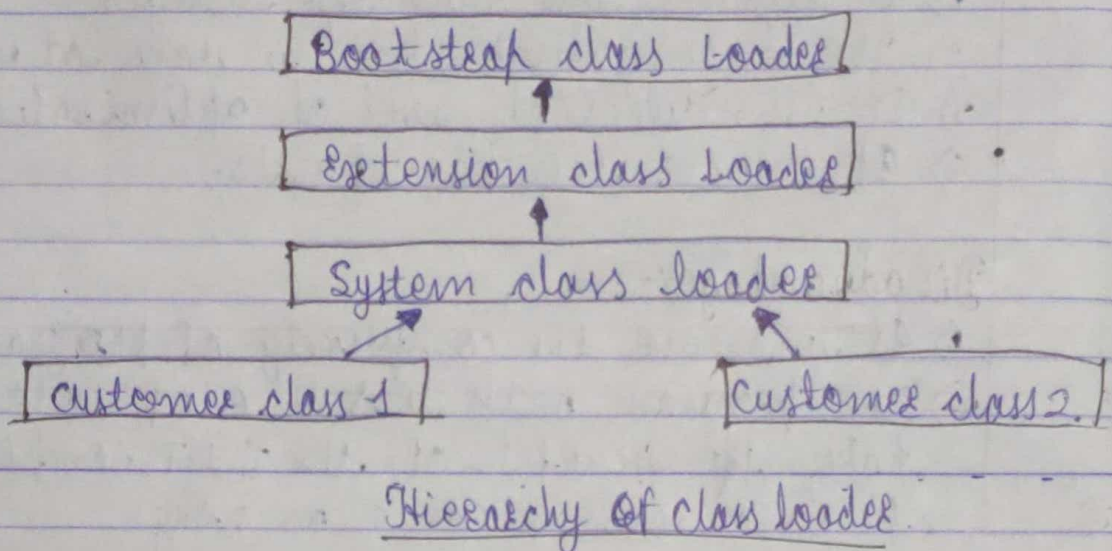3) It uses lots of cache memory.

Q3) What is class loader?

→ 1) Java classloader is an abstract class.
2) It belongs to a java.lang package.
3) It is used to load the classes at run-time.
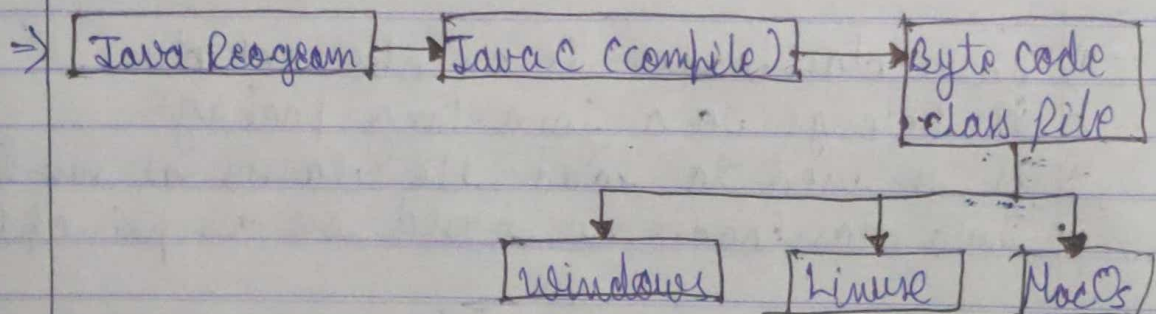4) Java classloader is based on 3 principle.

a) **Delegation:-** It forwards the request for class loading to parent class loader.

b) **Visibility:-** It allows child classloader to see all the classes loaded by parent classloader, but parent class loader cannot see classes loaded by child loader.

c) **Uniqueness:-** It allows to load a class once. It is achieved by delegation principle. It ensures that child classloader doesn't reload the class, which is already loaded by the parent.

```
        ┌──────────────────────────┐
        │  Bootstrap class Loader  │
        └──────────────────────────┘
                     ↑
        ┌──────────────────────────┐
        │ Extension class Loader   │
        └──────────────────────────┘
                     ↑
        ┌──────────────────────────┐
        │  System class loader     │
        └──────────────────────────┘
            ↗                  ↖
┌──────────────────┐    ┌──────────────────┐
│ customer class 1 │    │ customer class 2 │
└──────────────────┘    └──────────────────┘
```

<u>Hierarchy of class loader.</u>

Q 4) What gives java its "write once and Run Anywhere" nature?

```
→ ┌──────────────┐   ┌──────────────────┐   ┌────────────┐
  │ Java Program ├──→│ Java C (compile) ├──→│ Byte code  │
  └──────────────┘   └──────────────────┘   │ class file │
                                             └────────────┘
                          ┌─────────────────────┼──────────────┐
                          ↓                      ↓              ↓
                    ┌──────────┐          ┌──────────┐   ┌────────┐
                    │ windows  │          │  Linux   │   │ MacOS  │
                    └──────────┘          └──────────┘   └────────┘
```
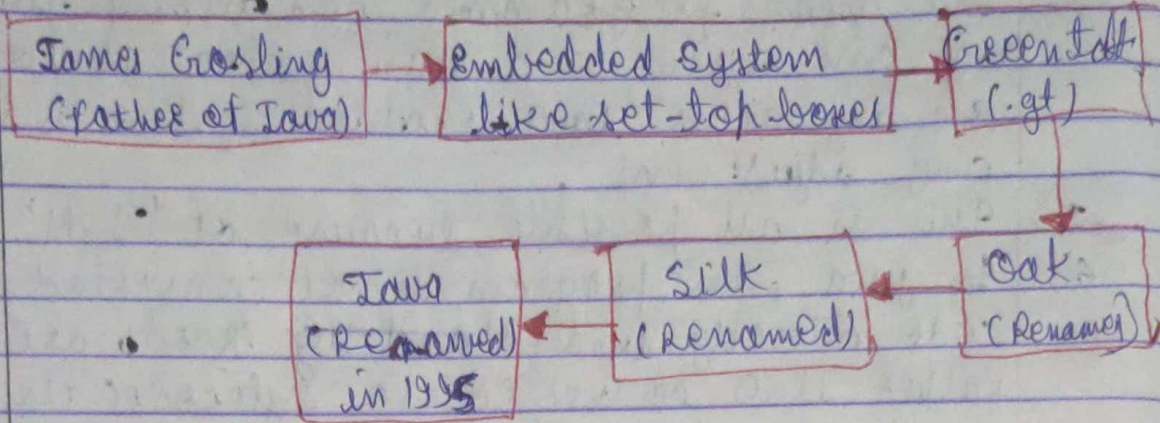
1) Java application are called WORA i.e (write once (and) Run Anywhere).

2) This means programmer can develop java code on one system and can expect it to run on any other Java-enabled system without any adjustment.

3) This is all possible because of "JVM".

4) In java, the program is not converted to code directly understand by Hardware, rather it is converted to 'bytecode(.class file), which is interpreted by JVM. so once compiled it generates bytecode file, which can be run anywhere (any machine) which can be run anywhere which has JVM & hence it gets the nature of 'write once and Run Anywhere'.

Q5) Explain History of Java & who invented Java?

=> 1) History of Java starts with 'Green Team'.

2) The principles for creating Java programming were : "Simple, Robust, Portable, platform-independent, secured. High-performan

3) Java is used in internet programming, mobile devices, games, e-bussiness solution

4) James Grosling, Mike sheridan, Patrick Naughton initiated java language project in June 1991.

These small team of sun engineers called "Green Team."

5) Java was developed by "James Gosling" who is known as father of Java in 1995.

James Gosling (father of Java)	→	Embedded System like set-top boxes	→	Green Jeff (.gt)

Java (Renamed) in 1995	←	Silk (Renamed)	←	Oak (Renamed)

Q6) What was the original name of java? why it was renamed?

⇒ 1) The original name of java was 'Oak' which was developed by a small team of engineers working for 'Micro 'Sun Microsystems'.

2) They called themselves the "Green Team".

3) The 'Oak' name was renamed dueto the fact that Oak was already registered as part of another trademark.

Q9) What is difference bet" System.out.print(); System err print, System.out.println();

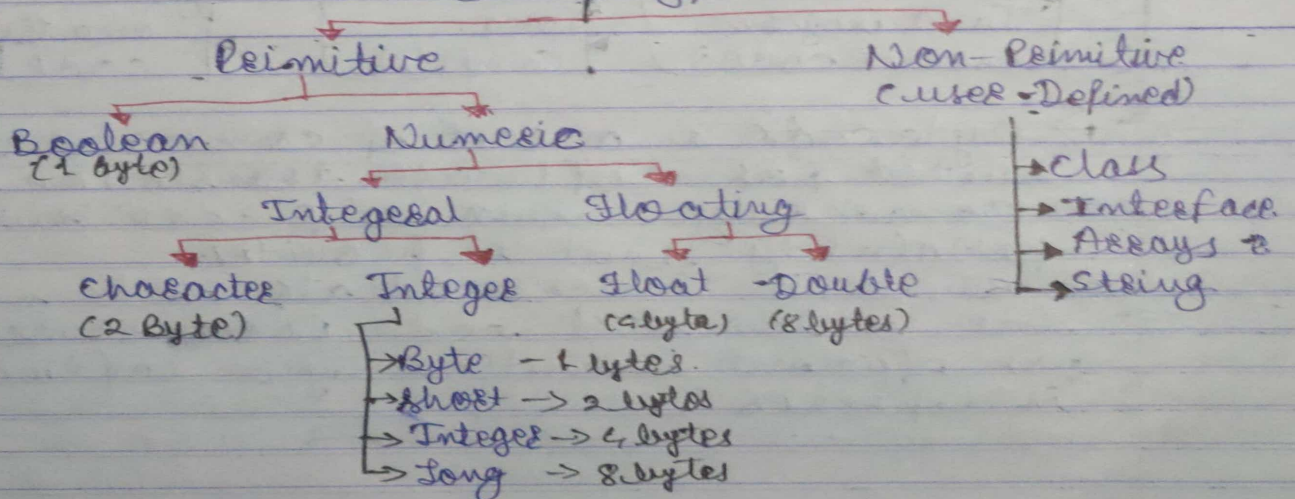| System.Out.print()	System.out.println()	System.err.print
The control or cursor remains on the same line after printing.	1) The control (cursor) moves to the next line after printing	1) System.err.print is used to display error messages
		2) The o/p is displayed in red colour.

Q7) List Features of Java?

=> The following are the features of Java:-
1) Simple
2) Object - Oriented
3) Portable
4) Platform - Independent
5) Secured
6) Robust
7) Architecture neutral
8) Interpreted
9) High performance
10) Multithread
11) Distributed
12) Dynamic

Q8) List various Datatypes in Java

Data types

Primitive   Non-Primitive (user -Defined)

Boolean (1 byte)   Numeric

Integeral   Floating

- Class
- Interface
- Arrays &
- String

character (2 Byte)   Integer   Float (4 byte)   Double (8 bytes)

→ Byte — 1 bytes
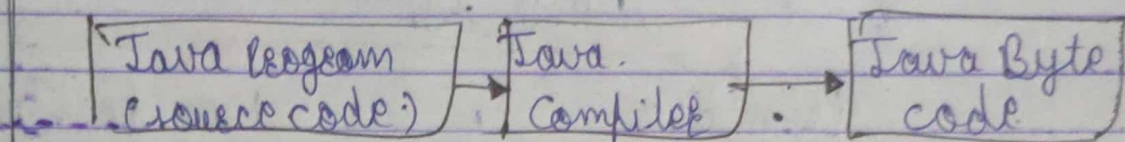→ short → 2 bytes
→ Integer → 4 bytes
→ Long → 8 bytes

Q10) How is java Platform independent?

⇒ 1) When you compile java programs using javac compiler it generates bytecode.
2) We can execute the bytecode in any platform which has JDK installed i.e Java Development Kit.
3) With the help of JVM which is present in JDK the java bytecode is translated into machine understandable code.
4) Hence, Java is Platform independent but it is purely depended on JDK

Q11) What is bytecode? How is it different from machine code?

⇒ Byte Code:

```
[ Java Program      ]    [ Java     ]    [ Java Byte ]
[ (source code) ]  →   [ Compiler ]  → [   code    ]
```

1) Byte code is a sort of command that is suited for software translation operation
2) Commonly known as 'P-code' due to portability that it provides.
3) It is a intermediate code compiled into a low-level code from source code for efficient execution by a software interpreter.

ByteCode	Machine Code
1) It is an intermediate code designed to run on a virtual machine instead of a central processing unit (CPU)	1) It is a computer program made up of the native instructions associated with that particular computer
2) The function of bytecode is to be a format that can be executed efficiently by the virtual machine.	2) Machine code is the language which all programs must be converted into before they can be run.
3) It is platform independent because it can be executed on any platform using the virtual machine.	3) It is not platform independent meaning it can't be run on just any platform with the same OS.

Q 12) Explain various memory logical partition?

=> 1) A logical partition (LPAR) is the division of a computer's memory, and storage into multiple sets of resources so that each set of resources so that each set of resources can be operated independently with its own operating system instance & applications.

2) The number of logical partitions are used for different purposes such as database operation or client/server operation or the separate test & production environment.

3) Each partitions can communicate with the other partitions as if other partition is in a separate machine.

Q(3) what is difference bet" jar file & runnable jar file

Jar File	Runnable jar file
1) Jar file is a java appl" which requires a command line to run, a runnable JAR file can be directly executed by double clicking	1) Runnable jar file allows a use to run java classes, without having to know class name & type them in command prompt, rather the user can just double click on the jar file and the program will fire up.
2) A (Java Archive) is a package file format typically used to aggregate many java class files associated metadata and resources into one file to distribute applica" software or libraries on the java platform.	2) A runnable jar allows java classes to be loaded just like when a user clicks on exe file.

Q(4) what is difference bet" Runnable jar file & exe file

Runnable jar file	exe file
1. Jar file are like dead	1. exe file are like living men
2. Jar file is the combination of compiled java classes	2. Executable jar file is also combination of compiled java classes with main class.

Q15) How is c platform dependent language?

⟹1. C is a portable programming language because it is not tied to any hardware or system.
2. We can say, it is a hardware independent language or platform independent language.
3. That is why c is called "Portable language".
4. C programs does not depend on actually but the executable file that is generated at the end for running the c-program many depend on a platform.
5. When you use os you get other extension for executable files

Q16) What is difference bet path & class path.

Path	class path
1) Path variable is used to set the path for all java software tools like javac.exe, java.exe, javadoc.exe, & so on.	1. class path variable is used to set the path for java classes.
2) variable natbre:- PATH variable value: c: Program Files\ Java\jdk 1.702-21\bin;	2. variable name:-classpath variable value: c:\program files\ Java\jre 2.6.0\ jre \lib\ \rt.jar