Assignment-3

_/_/__

1. Components of JDK :

JDK is a s/w development environment provided by Oracle Corporation enabling to build Java applications.

(i) Java compiler (javac) : It transforms Java source code into bytecode (.Java → .class)

(ii) JVM : It is runtime that executes Java bytecode

(iii) JRE : Subset of Jdk that includes JVM and class libraries

(iv) Java API : collection of pre built classes and methods

(v) jdb : Tool for debugging Java applications

2. difference (JVM, Jdk, JRE)

JVM → abstract machine
→ primarily assists in executing codes

JVM perform tasks such as : Loads code
verifies code
Execute code and
Provides runtime environment

JRE → It is used to provide runtime environment
set of libraries + other files that JVM uses

Jdk → JRE + development tools
* standard edition, Enterprise edition, Micro Edition.

3. Role of JVM in Java :

It enables the execution of Java bytecode. It acts as interpreter b/w Java programming language and the underlying h/w.

JVM executes code by

loading : class loader loads .class file into memory.

verifying : It checks the bytecode for security.

executing : It either interprets the bytecode directly

Managing Memory : It handles memory allocation and garbage collection

_/_/_

4. Memory Management in Java
- Memory Mgt is the process of allocation and de-allocation of objects.

Method Area: It is part of Heap memory which is shared among all the threads. It is used to store class structure, Super class name, interface name and constructors.
Heap Area: It stores the actual objects. It can be of fixed or dynamic size. There exists only one heap for each running JVM process.
Stack Area: It generates when a thread creates. The stack memory is allocated per thread. It is used to store data and partial results. It contains references to heap objects
Native Method Stack: It is a stack for native code written in language other than Java. Java Native Interface calls the native stack
PC registers: Each thread has a p.c register associated with it. It stores the return address of a native pointer

5. JIT compiler and its role in JVM
→ The JIT Compiler in JVM dynamically compiles sections of byte code into native machine code at runtime.
→ Java bytecode is the instruction set of Java virtual machine, the language to which source code is compiled

6. Architecture of the JVM
- consists of several components, including class loader, runtime data area, execution engine and native method interface.
These work together to execute Java byte code and Manage runtime Operations.

__/__/__

7. How does Java achieve platform independence.
→ platform independent bcoz "Write once run Anywhere" approach. Source code is compiled into byte code which is Platform neutral

8. Significance of classloader in Java, Process of garbage collection
→ class loaders are responsible for loading Java classes dynamically to JVM during runtime
→ Garbage collection in Java is the process of automatically reclaiming memory by deleting objects

9. four access modifiers in Java
1. public → Access from anywhere
2. protected → Accessible within same package and subclasses
3. default : Accessible only within the same Package
4. Private → Accessible only within defining class

10. can you Override a method with differen access modifie
→ NO, you cannot override a method with a more restrictive access modifier in a subclass. for example a protected method in the superclass cannot be overridden as private in subclass

12. diff b/w protected and default

Protected → access allows a member to be accessible within the same package and subclases, even if they are in different packages
Default → It allows a member to be accessible only within the same package.

_/_/_

13. Is it possible to make a class private in Java? if yes
→ NO, you cannot make a top level class private in Java
Limitations:
→ A private inner class is only accessible within Outerclass
→ It cannot be accessed or instantiated from outside the Outer class

14 Can a top level class in Java be declared as protected or private
→ NO, they cannot be declared bcoz The access modifiers are in samepackage or subclass

15. what happens if you declare a variable or method as private in class and try to access it from same package
→ Causes Compilation error

16. Concept of Package private or default access.
→ If you dont specify an access modifier, the member or class has package-private access by default
visibility by → within the same package
outside the package