Category – Java Basics

1. What makes Java unique among other technologies?

Hints – platform independence, security, auto mem management, support for multi-threading,remoting,finding run-time info(reflection)

1. Is Java 100 % object oriented?

No---prim data types still supported,multiple inheritance is not supported.

1. What is the significance of .class file?

Binary intermediate code—meant to be interpreted by JRE(i.e actually by JVM) -- same for all platforms,compact

1. Why compile time & run time environments are separate in Java?

--Platform indepencdence---ensures identical , uniform o/p for all platforms.(exc – AWT,Thrds –if not using wait/notification)

1. Is Java executable or interpreted language?

Intepreted – at runtime

1. If Java is interpreted language , then is it extremely slower than any other executable languages like CPP?

---performance comparable to any other executable lang --- due to the optimizations offered by JIT compiler.

1. What do you understand by JRE or Java run-time system?

Hint -The combination of Java virtual machine and Java API form a "platform" for which all Java programs are compiled. Its called *Java runtime system(or JRE) or Java platform.* Java programs can run on many different kinds of computers because the Java Platform can itself be implemented in software. (ref – jvm\_readme)

1. What is exact significance of JVM ? Explain different types.

Hint – jvm\_readme

1. How does Java interact with native OS?

Hint- When running on a Java virtual machine that is implemented in software on top of a host operating system, a Java program interacts with the host by invoking *native methods*. In Java, there are two kinds of methods: Java and native. A Java method is written in the Java language, compiled to bytecodes, and stored in class files. A native method is written in some other language, such as C, C++, or assembly, and compiled to the native machine code of a particular processor. Native methods are stored in a dynamically linked library whose exact form is platform specific. While Java methods are platform independent, native methods are not. When a running Java program calls a native method, the virtual machine loads the dynamic library that contains the native method and invokes it. Native methods are the connection between a Java program and an underlying host operating system.

1. What is main role of class loader?

Hint – Applying security and network- mobility is the main job of class loader architecture. Its part of JVM. It allows Java application to load Java classes(.class) in JVM’s memory.

1. What are types of class-loaders?

Hint - A Java application can use two types of class loaders: a "bootstrap" class loader and user-defined class loaders. **The bootstrap class loader** (there is only one of them) is a part of the Java virtual machine implementation. The bootstrap class loader loads classes, including the classes of the Java API, in some default way, usually from the local disk.**User defined class loader** - at run-time, a Java application can install user-defined class loaders that load classes in custom ways, such as by downloading class files across a network.

Eg –RMI class loader, Applet class loader.

For details – classloaders\_readme

1. What is role of Byte code verifier?

Hint – To perform mainly these checks—

Branches are always to valid locations

Data is always initialized and references are always type-safe

Access to "private" or "package private" data and methods is rigidly controlled.

For details – byte\_code\_verifier\_readme

1. What is role of security manager?

Hint – It is integral part of JVM. Earlier versions used security manager and later versions use access controller for following.

It consults security policy (text file for granting permissions) to determine whether the operation should be allowed. By enforcing the security policy established by the security manager and access controller, the Java API helps to establish a safe environment in which you can run potentially unsafe code. Eg : Downloaded applets running on client machines.

1. Why does Java maintain primitive data types? State different data types & their precision.

Hint – for the purpose of efficiency – as compiler already knows primitive data type’s width & behaviour.

1. What are Automatic conversions & narrowing conversions?

---byte-🡪short-🡪int-🡪long🡪float-🡪double – auto conversion. (eg --- byte b1=10;byte b2=b1+10;byte b3=5;b2=b1+b3; b1 += b2;)

1. How does Garbage collector work?

Hint –Daemon thrd,periodic interval,job—to check for heaped objects ---having no active reference--- re-cycle resources(mem – java resources)

Category – OOP implementation in Java & Basic Java API

1. Explain 2 diferent types of polymorphism. How its implemented in Java?

2. What are different restrictions on overriding form of methods?

--same meth name, ret type—same or sub-type(sub-class),method signature-same,scope-same/wider,can’t add any new chked exc.

3. What is co-variance ?

--added from JDK 1.5 onwards. It means---overriding form of the method can return the sub-type of the original form.

Eg (in Emp class --- Emp getInstance(){…..}, in Mgr class --- Emp/Mgr getInstance(){….}

4. What is dynamic method dispatch? What are the restrictions?

---- The decision of which form of the method to be sent for execution – is taken dynamically(in run-time)

How to use it ? --- Super-class ref referring to sub-class inst or i/f ref pointing to imple. class inst.

Restriction--- Features(members) accessible --- only those mentioned in super-class or i/f.

5. Explain Upcasting & downcasting of references. What is role of ‘instanceof’ keyword?

---upcasting --- super-type ref referring to sub-type.(auto.)

--- downcasting ---- requires specific type casts.--- sub-type ref --- referring to super type.

instanceof -- run time operator – chks actual type of the ref.---avoid ClassCastException

6. What is the difference between abstract class & interface?

--- abstract classes may contain – constructrs,methods& data members. I/f—100% specification, D.M --- public,static,final & methods – public & abstract.

7. What is difference between --- final,finally,finalize

--final ref--- can’t be re-assigned.

--finally ---exec always (except--- System.exit())

--finalize --- a method called by GC – before releasing Object’s resources.

8. What are different types of nested classes supported in Java?

---inner(non-static nested),statically nested,method local inner class & anonymous inner class.

Hint – For details – nested\_class\_rules

9. Can you create array of heterogeneous types in Java ? If yes how ?

--Yes , by creating array of type Object

10. What do you understand by mutable arrays? What about its references?

11. What are the things to remember while proper casting?

* Check for proper casting by using the following guidelines
  + Casts up the hierarchy are done implicitly
  + Downward casts must be to a sub-class and checked by the compiler
  + If the compiler allows the casting, the object type is checked at runtime when runtime errors can occur(to avoid run-time err—ClassCastException, use instanceof)

12. Can abstract class contain a constructor? If yes explain its purpose.

--yes ---must when super-class has private D.M – for its init.

13. Can any Java class have private constructor? If yes explain its purpose.

---eg scenario –for singleton pattern –to control class instantiation.

14. What are wrapper classes ?

---class equivalent of primitive data types.

15 JDK 1.5 onwards, why significance of wrappers is reduced?

--auto-boxing & auto-unboxing

16. Give scenario where finally block does not execute.

17. What is difference between equals and ==?

---equals –typically used for comparing the values(objects) referred by the references.

== -- comparing the references.

18 What is constant or literal string pool?

-- created by JVM – when it comes across literal string in Java appln. Only 1 copy of the reference will exist in the pool.

19. In following code sample – how many objects become candidates for garbage collection at --- // last line of block # 1

|  |
| --- |
| public class ImmutableStrings  {  public static void main(String[] args)  {  String one = "someString";  String two = new String("someString");    one = two = null;  // last line of block # 1  String thre=”someString”;  //some more code follows…..  }  } |

20. Immutable string object and mutable string reference.

--- string object , once created can’t be modified. But its ref(unless final) can be re-assigned.

21. Difference between String,StringBuffer,StringBuilder

---StringBuffer- thrdsafe & StringBuilder ---thrd unsafe.

22. Revise String class API –format(java.util.Formatter) ,indexOf,contains,split,compareTo,equals.

23. Revise API of – Date,GregorianCalendar,StringTokenizer,Scanner

24. How to convert GC to Date?

---getTime()

25. Explain Date Time formatting.

-- ref java.util.Formatter --- %tT,%tC,%tD

26. Explain overview of Collection framework.

--- readymade implementation of standard algorithms,which also allows you extensibility(via abstract classes)

Collection i/f --- generic i/f w/o any particular functionality.--- Can attach Iterator

List/Set/Map

List --- allows dups,index based growable collection,order is significant.(Iterator/LI)

Set—doesn’t allow dups, un-ordered, unsorted collection. No additional API from Collection i/f.(supports Iterator, but doesn’t support LI)---

Map—DOESN’T extend Collection i/f --- can’t attach iterator.(For iteration purposes--- 1st convert it to Set/Collection & then iterate). ----set of entries(key& value)---keys distinct.

27. State the scenario – for choosing LinkedList over ArrayList.

---Choose LinkedList – in case of insert/remove operations anywhere in the list.

28. When do you override both hashCode & equals methods in your class?

--- It becomes mandatory to override both of the methods –when u are adding class objects to the Set.

(HashSet,TreeSet,LinkedHashSet)

Internals of Hashing algorithm---

Set will invoke hashCode()--- get bucket ID---if bucket is empty , adds data directly. BUT if bucket is partially filled ---invokes equals() . If data element is not present in bucket(ie. Equals() rets false for all bucket members) , then data element is added to the set.(Rule --- if equals() method rets true – for 2 references, then theit hashCode() values must be same)

29. What is the advantage of set over list functionality?

---detects & removes dups, Set offers constant time factor for add,search,remove operations

30 Difference between ArrayList & Vector OR and HashMap & HashTable

Vector is thrd safe(synchronized)—slower

ArrayList – is thrd –unsafe---faster.

--

31 If your domain data (Business Objects) is stored in a List/Set implementation, how will you sort it using natural ordering or custom ordering?

List --- Collections.sort(List l1) / Collections.sort(List l1,Comparator inst)

Set --- convert HashSet – to TreeSet

Map—convert HashMap to TreeMap

32. Above question can be rephrased as – difference between Comparable & Comparator

--Comparable specifies –natural ordering criteria, to be added in the Business class itself by overriding compareTo.

--Comparator specifies –custom/external ordering criteria, to be added in the seprate class(or inner class) i by overriding compare.

33.

What is a special feature of Java I/O

---device independence.

33. Explain difference between Scanner and BufferedReader.

---Scanner supports parsing of text data, BR –does not.

---Scanner is not associated with any buffer(except line buffer), BR can be created with specified size buffer.

Typically u can combine both.

Eg ---- To read data from socket –in parsed,buffered manner.

BufferedReader br=new Bufferedreader(new InputStreamReader(s1.getInputStream());

Scanner sc=new Scanner(br.readLine());

34. What are important features of PrintWriter?

Hint – buffered char oriented, o/p stream, can wrap File,Writer,OutputStream & provides autoFlush capacity.

35. What is serialization? Its rules and associated I/O API classes, transient keyword

Details for 35,36,37 – in serialization\_readme

36 Give examples of non-serializable Java classes – Connection,File

37. What is serialversionUID?

38 True or False?

If a superclass is not serializable then the subclass can't be serializable. -- false

When objects are deserialized they are read back in last-in, first out sequence. --- false

When an object is deserialized, its constructor does not run. ---true

39. What are Java annotations? Supported since which version?

--- commands/instructions for Java compiler/JRE/Both.

Eg -- @Override,@SuppressWarnings

JEE annotations ---@Resource,@Stateless,@PreConstruct,@Entity

Supported since JDK 1.5 onwards

40. What are different ways of creating threads in Java?

---extending Thread class,implementing Runnable i/f, or by creating a Runnable Task & using the support from Executor Framework.

41. What are different states of a thread?

---does not exist,new(newly created thrd),ready-to-run,running,blocked,dead.

Running-🡪blocked(sleep,i/o blocking,unavailability of monitor,wait,join)

Running-🡪rdy (yield,time slice over in time-sliced multi-tasking,lower prio. Thread gets pre-empted by a higher prio thread---in pre-emptive multi-tasking.)

Running-🡪dead--- run() method over or run() gets unhandled exception.

42. Explain yield() method of Thread class.

---it’s a static method of thread class.

When running thrd invokes yield() method, it gives up control of CPU & enters ready state.

43. Explain InterruptedException, interrupt() & its use.

--The methods with InterruptedException(checked exc)

sleep,join,wait ---implies that –if a thrd is blocked either by calling sleep/join/wait can be un-blocked –by throwing interrupt() signal . The thrd then get IE.

44. What will happen if you call run() instead of start()?

--no new thrd will be created

45 What will happen when u extend from Thread class & do not override run()?

---thrd will execute blank(do-nothing) run() method inherited from Thread class , so does not exec. any B.L

46. What are different ways of applying synchronization?

--via synchrnozed methods or synch blocks.

Whats the difference between 2 approaches?

--- synchronized methods --- lock is applied for the entire method. It represents –locking code inherently.

---sync blocks – lock applied only for the block,finer control, can apply thrd safety externally.

47. What is the meaning of static synchronized methods?

--- the lock for thrd safety is applied at the class level(meaning --- if 1 thrd has invoked any static synch method of the class, then any other thrd invoking same or any other static , sync method of the same class will get blocked)

48. Explain similarity & difference between sleep() & wait()

--both are blocking calls, include in throws clause IE,ca be specified with tmout.

--Difference ---sleeping thrd sleeps inside the monitor. Waiting thrd waits outside the monitor.

49. Can a thread blocked on I/O , get unblocked by raising interrupt signal?

---No . Options are either use java.nio or daemon thrds.

50. Diffentiate between user thread & daemon thread.

---user thrds don’t necessarily expire when parent thrd expires. Daemon thrds expire along with parent thrds.

51. Differentiate between notify() & notifyAll()

----notify() --- unblocks only 1 waiting thrd on the same monitor.

---notifyAll() --- unblocks all waiting thrd on the same monitor.

52. Explain with example how will you create animated applet or Frame

---

53. Differentiate between AWT & Swing

--AWT components –heavy weight components.(as actual GUI is rendered by the native windowing component)

--Swing comps—light weight—do not rely on underlying OS for GUI rendering ---100 % Java Comps(Java Beans)

AWT ---thrd safe, Swing ---thrd un-safe components

AWT ---does n’t support PL&F, Swing supports --- dynamic changing of Look & Feel of the GUI appln/applet

AWT ---doesn’t support various panes, Swing supports

AWT – doesn’t support MVC, swing supports.

53 Which are thrds in GUI appln/applet?

Main thrd(invoke main method, applet’s --- init,start,stop,destroy)

Event Dispatcher thrd --- paint,event handling methods(actionPerformed..)

GC – Daemon Thrd

+ Optional user created thrds.

54. Discuss concurrency issues in Swing.

---Ensure that ---add swing components to the Frame/Applet using Event dispatcher thrd.

---Don’t delay any of the system’s thrds(main or Event dispatcher) – in such case create your own thread.

For actual example – HelloGUIWorld.java

55 Can you create GUI application having – Jframe,JApplet,JButton,JTF,JLabel,JPanel,JMenu,JList,JComboBox,JRadioButton,JCheckBox & their event handling.

56. Differentiate between applet and application

---application---standalone appln, trusted,exec starting point –main –sequential execution

---applet –Java class developed @ server & dynamically downloaded to clnt machine along with web page.---untrusted(CANT --- r/w system properties,r/w clnt’s hard disk,open any new socket cons--- otherwise gets SecurityException)

Applet code ---consists of call-back methods(init,start,stop,destroy)---typically no main.

57. Can you name any Java API abstract classes – having no abstract functionality?

---Adapter classes.

58. How will you develop TCP server application handling multiple clients concurrently?

59. What is the difference between socket & serversocket class

SS—represents server side control port. Cant attach data streams to SS. Its job--- connection establishment with clnt.

SS has – backlog factor=no of concurrent reqs SS can handle.

Socket class – represents either Server side data socket or client side socket. Used for data transfer between server & clnt. No backlog factor here.

60. Can you upload or download text/binary file using Socket class API?

--yes

61. What is RMI?

---Using server proxy classes(stub), clnt can invoke methods of the server object(remote object) directly(ie. Independent of location of clnt & server). Its blocking method call.---100 % Java –language dependent soln for distributed programming. --- works on JRMP / RMI/IIOP protocols.

62. RMI layers

--- clnt appln layer, stub, Clnt side RRL,TL---TL,server side RRL(distributed Garbage collection),Server appln layer(hosting remote object)

63. What is role of RMIRegistry?

---bind & lookup.

63 How do you distribute Java SE applications?

---bundle Java SE application/applets in JAR form & distribute the same.

64. Can you create executable JAR?

---yes --- by creating manifest file in JAR. ---specify name of the class containing main() in manifest file & can create a Runnable JAR