Java Interview Questions

Brought to you by:
World's Largest Resource for FREE Interview / Viva Questions
http://www.fundoosite.com/interview-questions/
A Vyom Enterprise – www.vyomworld.com

1. How can you achieve Multiple Inheritance in Java?

Java's interface mechanism can be used to implement multiple inheritance, with one important difference from c++ way of doing MI: the inherited interfaces must be abstract. This obviates the need to choose between different implementations, as with interfaces there are no implementations.

2. Replacing Characters in a String

```
// Replace all occurrences of 'a' with 'o'
String newString = string.replace('a', 'o');
Replacing Substrings in a String
static String replace(String str,
String pattern, String replace) {
int s = 0;
int e = 0;
StringBuffer result = new StringBuffer();
while ((e = str.indexOf(pattern, s)) >= 0) {
result.append(str.substring(s, e));
result.append(replace);
s = e+pattern.length();
}
result.append(str.substring(s));
return result.toString();
}
Converting a String to Upper or Lower Case
// Convert to upper case
String upper = string.toUpperCase();
// Convert to lower case
String lower = string.toLowerCase();
Converting a String to a Number
int i = Integer.parseInt("123");
long l = Long.parseLong("123");
float f = Float.parseFloat("123.4");
double d = Double.parseDouble("123.4e10");
Breaking a String into Words
String aString = "word1 word2 word3";
StringTokenizer parser =
new StringTokenizer(aString);
while (parser.hasMoreTokens()) {
processWord(parser.nextToken());
```

3. Searching a String

```
String string = "aString";
// First occurrence.
int index = string.indexOf('S'); // 1
// Last occurrence.
index = string.lastIndexOf('i'); // 4
// Not found.
index = string.lastIndexOf('z'); // -1
```

4. Connecting to a Database and Strings Handling

Constructing a String

If you are constructing a string with several appends, it may be more efficient to construct it using a StringBuffer and then convert it to an immutable String object.

```
StringBuffer buf = new StringBuffer("Initial Text");
// Modify
int index = 1;
buf.insert(index, "abc");
buf.append("def");
// Convert to string
String s = buf.toString();
```

Getting a Substring from a String

```
int start = 1;
int end = 4;
String substr = "aString".substring(start, end); // Str
```

5. What is a transient variable?

A transient variable is a variable that may not be serialized. If you don't want some field not to be serialized, you can mark that field transient or static.

6. What is the difference between Serializalble and Externalizable interface?

When you use Serializable interface, your class is serialized automatically by default. But you can override writeObject() and readObject()two methods to control more complex object serailization process. When you use Externalizable interface, you have a complete control over your class's serialization process.

7. How many methods in the Externalizable interface?

There are two methods in the Externalizable interface. You have to implement these two methods in order to make your class externalizable. These two methods are readExternal() and writeExternal().

8. How many methods in the Serializable interface?

There is no method in the Serializable interface. The Serializable interface acts as a marker, telling the object serialization tools that your class is serializable.

9. How to make a class or a bean serializable?

By implementing either the java.io.Serializable interface, or the java.io.Externalizable interface. As long as one class in a class's inheritance hierarchy implements Serializable or Externalizable, that class is serializable.

10. What is the serialization?

The serialization is a kind of mechanism that makes a class or a bean persistence by having its properties or fields and state information saved and restored to and from storage.

11. What are synchronized methods and synchronized statements?

Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

12. What is synchronization and why is it important?

With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often causes dirty data and leads to significant errors.

13. What is the purpose of finalization?

The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.

14. What classes of exceptions may be caught by a catch clause?

A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.

15. What is the difference between the Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy?

The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

16. What happens when a thread cannot acquire a lock on an object?

If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available.

17. What restrictions are placed on method overriding?

Overridden methods must have the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides. The overriding method may not throw any exceptions that may not be thrown by the overridden method.

18. What restrictions are placed on method overloading?

Two methods may not have the same name and argument list but different return types.

19. How does multithreading take place on a computer with a single CPU?

The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.

20. How is it possible for two String objects with identical values not to be equal under the == operator?

The == operator compares two objects to determine if they are the same object in memory. It is possible for two String objects to have the same value, but located indifferent areas of memory.

21. How are this() and super() used with constructors?

this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.

22. What class allows you to read objects directly from a stream?

The ObjectInputStream class supports the reading of objects from input streams.

23. What is the ResourceBundle class?

The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.

24. What interface must an object implement before it can be written to a stream as an object?

An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.

25. What is Serialization and deserialization?

Serialization is the process of writing the state of an object to a byte stream. Deserialization is the process of restoring these objects.

26. What are the Object and Class classes used for?

The Object class is the highest-level class in the Java class hierarchy. The Class class is used to represent the classes and interfaces that are loaded by a Java program.

27. Can you write Java code for declaration of multiple inheritance in Java ?

```
Class C extends A implements B
{
}
```

28. What do you mean by multiple inheritance in C++ ?

Multiple inheritance is a feature in C++ by which one class can be of different types. Say class teachingAssistant is inherited from two classes say teacher and Student.

29. Write the Java code to declare any constant (say gravitational constant) and to get its value.

```
Class ABC
{
        static final float GRAVITATIONAL_CONSTANT = 9.8;
        public void getConstant()
        {
                system.out.println("Gravitational_Constant: " + GRAVITATIONAL_CONSTANT);
        }
}
```

30. What are the disadvantages of using threads?

DeadLock.

31. Given two tables Student(SID, Name, Course) and Level(SID, level) write the SQL statement to get the name and SID of the student who are taking course = 3 and at freshman level.

```
SELECT Student.name, Student.SID
FROM Student, Level
WHERE Student.SID = Level.SID
AND Level.Level = "freshman"
AND Student.Course = 3;
```

32. What do you mean by virtual methods?

virtual methods are used to use the polymorhism feature in C++. Say class A is inherited from class B. If we declare say fuction f() as virtual in class B and override the same function in class A then at runtime appropriate method of the class will be called depending upon the type of the object.

33. What do you mean by static methods?

By using the static method there is no need creating an object of that class to use that method. We can directly call that method on that class. For example, say class A has static function f(), then we can call f() function as A.f(). There is no need of creating an object of class A.

34. What do mean by polymorphism, inheritance, encapsulation?

Polymorhism: is a feature of OOPl that at run time depending upon the type of object the appropriate method is called.
Inheritance: is a feature of OOPL that represents the "is a" relationship between different objects(classes). Say in real life a manager is a employee. So in OOPL manger class is inherited from the employee class.
Encapsulation: is a feature of OOPL that is used to hide the information.

35. What are the advantages of OOPL?

Object oriented programming languages directly represent the real life objects. The features of OOPL as inhreitance, polymorphism, encapsulation makes it powerful.

36. How many methods do u implement if implement the Serializable Interface?

The Serializable interface is just a "marker" interface, with no methods of its own to implement.

37. Are there any other 'marker' interfaces?

java.rmi.Remote
java.util.EventListener

38. What is the difference between instanceof and isInstance?

instanceof is used to check to see if an object can be cast into a specified type without throwing a cast class exception. isInstance() determines if the specified object is assignment-compatible with the object represented by this Class. This method is the dynamic equivalent of the Java language instanceof operator. The method returns true if the specified Object argument is nonnull and can be cast to the reference type represented by this Class object without raising a ClassCastException. It returns false otherwise.

39. why do you create interfaces, and when MUST you use one?

You would create interfaces when you have two or more functionalities talking to each other. Doing it this way help you in creating a protocol between the parties involved.

40. What's the difference between the == operator and the equals() method? What test does Object.equals() use, and why?

The == operator would be used, in an object sense, to see if the two objects were actually the same object. This operator looks at the actually memory address to see if it actually the same object. The equals() method is used to compare the values of the object respectively. This is used in a higher level to see if the object values are equal.

Of course the the equals() method would be overloaded in a meaningful way for whatever object that you were working with.

41. Discuss the differences between creating a new class, extending a class and implementing an interface; and when each would be appropriate.

* Creating a new class is simply creating a class with no extensions and no implementations. The signature is as follows
public class MyClass()
{
}

* Extending a class is when you want to use the functionality of another class or classes. The extended class inherits all of the functionality of the previous class. An example
of this when you create your own applet class and extend from java.applet.Applet. This gives you all of the functionality of the java.applet.Applet class. The signature would
look like this

```
public class MyClass extends MyBaseClass
{
}
```

* Implementing an interface simply forces you to use the methods of the interface implemented. This gives you two advantages. This forces you to follow a standard(forces
you to use certain methods) and in doing so gives you a channel for polymorphism. This isn't the only way you can do polymorphism but this is one of the ways.

```
public class Fish implements Animal
{
}
```

42. Given a text file, input.txt, provide the statement required to open this file with the appropriate I/O stream to be able to read and process this file.

43. Name four methods every Java class will have.

```
public String toString();
public Object clone();
public boolean equals();
public int hashCode();
```

44. What does the "abstract" keyword mean in front of a method? A class?

Abstract keyword declares either a method or a class. If a method has a abstract keyword in front of it, it is called abstract method.Abstract method has no body. It has only arguments and return type. Abstract methods act as placeholder methods that are implemented in the subclasses. Abstract classes can't be instantiated.If a class is declared as abstract,no objects of that class can be created.If a class contains any abstract method it must be declared as abstract.

45. Does Java have destructors?

No garbage collector does the job working in the background

46. Are constructors inherited? Can a subclass call the parent's class constructor? When?

You cannot inherit a constructor. That is, you cannot create a instance of a subclass using a constructor of one of it's superclasses. One of the main reasons is because you probably don't want to overide the superclasses constructor, which would be possible if they were inherited. By giving the developer the ability to override a superclasses constructor you would erode the encapsulation abilities of the language.

47. What synchronization constructs does Java provide? How do they work?

48. Why "bytecode"? Can you reverse-engineer the code from bytecode?

49. Does Java have "goto"?

No

50. What does the "final" keyword mean in front of a variable? A method? A class?

FINAL for a variable : value is constant
FINAL for a method : cannot be overridden
FINAL for a class : cannot be derived

IMPORTANT RESOURCES FOR STUDENTS

- FREE career resources for students – www.vyomworld.com.
- World's Largest Free eBooks directory – www.bestebooksworld.com
- Take Free Online Exams/Mock exams for GRE, GATE, TOEFL, MCSD, MCSE, CCNA, CAT, C, C++, JAVA, IIT etc at www.testsworld.com
- Get Complete set of Placement papers of all companies & Software jobs FREE at www.jobsassist.com
- FREE Download 20,000++ softwares & ebooks at www.vyomlinks.com
- Get Thousands of FREE Source codes in most of Computer platforms at www.sourcecodesworld.com
- Get your FREE dose of Fresher/Experienced Jobs direct to your mail box at www.jobsassist.com/careermag
- Discuss Freely all your enquiries and doubts to the greatest possible depth at www.discussionsworld.com
- World's largest FREE interview / viva questions resource – www.fundoosite.com/interview-questions/