FAQ's on IO Streams

1. What value does read() return when it reaches the end of a file?

The read() method returns -1 when it reaches the end of a file.

2. What is serialization?

Serialization is a mechanism by which you can save the state of an object by converting it to a byte stream.

3. How do I serialize an object to a file?

The class, whose instances are to be serialized, should implement an interface Serializable. Then you pass the instance to the ObjectOutputStream, which is connected to a fileoutputstream. This will save the object to a file.

4. Which methods of Serializable interface should I implement?

The serializable interface is an empty interface; it does not contain any methods. So we do not implement any methods.

5. Can you customize the seralization process? If yes, then how can one have a control over the serialization process?

Yes it is possible to have control over serialization process. For this, the class should implement Externalizable interface. This interface contains two methods namely readExternal and writeExternal.

You should implement these methods and write the logic for customizing the serialization process.

6. What is the common usage of serialization?

Whenever an object is to be sent over the network, objects need to be serialized. Moreover if the state of an object is to be saved, objects need to be serilazed.

7. What is Externalizable interface?

Externalizable is an interface which contains two methods readExternal and writeExternal. These methods give you a control over the serialization mechanism.

Thus, if your class implements this interface, you can customize the serialization process by implementing these methods.

8. When can you serialize an object? What happens to the object references included in the object once you serialize them?

The serialization mechanism generates an object graph for serialization. Thus it determines whether the included object references are serializable or not. This is a recursive process.

When an object is serialized, all the included objects are also serialized along with the original object.

9. While serializing the object, shat should be taken care of?

One should make sure that all the included objects are also serializable. If any of the objects is not serializable then it throws a NotSerializableException.

10. What happens to the static fields of a class during serialization?

There are three exceptions in which serialization does not necessarily read and write to the stream. These are:

- Serialization ignores static fields, because they are not part of any particular state.
- Base class fields are only handled, if the base class itself is serializable.
- Transient fields.

11. 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 leads to significant errors.

12. How does Java handle integer overflows and underflows?

It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

13. Does garbage collection guarantee that a program will not run out of memory?

Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection.

14. What is the difference between preemptive scheduling and time slicing?

Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence.

Under time slicing, a task executes for a predefined slice of time and then re-enters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

15. What is the initial state of a thread, when it is created and started?

A thread is in the ready state after it has been created and started.

16. What is a transient variable?

A transient variable is a variable that may not be serialized.

17. Why do threads block on I/O?

Threads block on I/O (that is enters the waiting state) so that other threads may execute while the I/O Operation is performed.

18. What value does readLine() return when it has reached the end of a file?

The readLine() method returns null when it has reached the end of a file.

19. When a thread blocks on I/O, what state does it enter?

A thread enters the waiting state when it blocks on I/O.

20. What value does read() return when it has reached the end of a file?

The read() method returns -1 when it has reached the end of a file.

21. 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.

22. What is the purpose of the File class?

The File class is used to create objects that provide access to the files and directories of a local file system.

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

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

24. What an I/O filter?

An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.

25. What is the difference between the File and RandomAccessFile classes?

The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

26. 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.

27. What are Transient and Volatile Modifiers?

Transient Modifiers: The transient modifier applies to variables only and it is not stored as part of its object's Persistent state. Transient variables are not serialized.

Volatile Modifiers: Volatile modifier applies to variables only and it tells the compiler that the variable modified by volatile can be changed unexpectedly by other parts of the program.

28. What is a stream, types of Streams and classes of Streams?

A Stream is an abstraction that either produces or consumes information. There are two types of Streams and they are:

- **Byte Streams:** Provide a convenient means for handling input and output of bytes.
- **Character Streams:** Provide a convenient means for handling input & output of characters.
- **Byte Streams Classes:** Are defined by using two abstract classes, namely InputStream and OutputStream.
- **Character Streams Classes:** Are defined by using two abstract classes, namely Reader and Writer.

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

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

30. 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.