

## 1. What is Input and Output Stream in Java?

**Ans:** A stream can be defined as a sequence of data. The `InputStream` is used to read data from a source and the `OutputStream` is used for writing data to a destination.

## 2. What are the methods of `OutputStream`?

- `write ()` - writes the specified byte to the output stream.
- `write (byte [] array)` - writes the bytes from the specified array to the output stream.
- `flush()` - forces to write all data present in the output stream to the destination.
- `close()` - closes the output stream

## 3. What is serialization in Java?

**Ans:** Serialization is the process of converting an object into a stream of bytes to transfer it over a network or to store it in a file or database. In Java, serialization is done by implementing the `Serializable` interface.

## 4. What is the `Serializable` interface in Java?

**Ans:** The `Serializable` interface in Java is a marker interface that has no methods. It is used to mark classes that can be serialized, meaning their object instances can be converted into a stream of bytes.

## 5. What is deserialization in Java?

**Ans:** Deserialization is the process of converting a stream of bytes back into an object instance. This is done after an object has been serialized.

## 6. How is serialization achieved in Java?

**Ans:** Serialization is achieved in Java by implementing the `Serializable` interface. When an object is serialized, its state is converted into a stream of bytes, which can then be transferred over a network or stored in a file or database.

## 7. How is deserialization achieved in Java?

**Ans:** Deserialization is achieved in Java by reading a stream of bytes and using them to recreate the original object instance. This is done by calling the `readObject()` method of an `ObjectInputStream` instance.

## 8. How can you avoid certain member variables of class from getting Serialized?

**Ans:** Mark member variables as static or transient, and those member variables will no more be a part of Serialization.

## 9. What classes are available in the Java IO File Classes API?

**Ans:** The following classes are available in the Java IO API and are important to work with files in Java.

File

RandomAccessFile

FileInputStream

FileReader

FileOutputStream

FileWriter

## 10. What is Difference between Externalizable and Serialization interface?

	<b>Serializable</b>	<b>Externalizable</b>
Methods	It is marker interface and it doesn't have any method.	It's not a marker interface. It has method's called writeExternal() and readExternal().
Default Serialization process	Yes, Serializable provide its own default serialization process, we just need to implement a Serializable interface.	No, we need to override writeExternal() and readExternal() for the serialization process to happen.
Customize serialization process	We can customize default serialization process by defining following methods in our class > readObject() and writeObject().	Serialization process is completely customized. We need to override the Externalizable interface's writeExternal() and readExternal() methods.
Control over Serialization	It provides less control. Over serialization as it's not mandatory to define readObject() and writeObject() methods.	Externalizable provides. You great control over the serialization process as it is important to override writeExternal() and readExternal() methods.
Constructor call during deserialization	Constructor is not called during deSerialization.	Constructor is called during deSerialization.