Understanding Data Streams:

- These Streams handle binary I/O operations on primitive data types.
- DataInputStream and DataOutputStream are **filter** streams (A filter stream filters data as it's being read or written to the stream)that let you read or write primitive data types
- DataInputStream and DataOutputStream implement the DataInput and DataOutput interfaces, respectively.
- These interfaces define methods for reading or writing the Java primitive types, including numbers and Boolean values.
- DataOutputStream encodes these values in a machine-independent manner and then writes them to its underlying byte stream.
- DataInputStream is created with a FileInputStream as source for its data.
- DataOutputStream is created with a FileOutputStream as source for its data.

Understanding Serialization

- The process of saving (or) writing state of an object to a file is called serialization.
- In other words it is a process of converting an object from java supported version to network supported version (or) file supported version.

```
ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStream("File Path"));
```

- By using FileOutputStream and ObjectOutputStream classes we can achieve serialization.
- The process of reading state of an object from a file is called DeSerialization.
- By using FileInputStream and ObjectInputStream classes we can achieve DeSerialization.

```
ObjectInputStream ois=new ObjectInputStream(new FileInputStream("File Path"));
```

Important Points to remember

- We can perform Serialization only for Serializable objects.
- An object is said to be Serializable if and only if the corresponding class implements Serializable interface.
- Serializable interface present in java.io package and does not contain any methods. It is marker interface. The required ability will be provided automatically by JVM.
- We can add any no. Of objects to the file and we can read all those objects from the file, but in which order we wrote objects in the same order only the objects will come back ie, reterving order is important.
- If we are trying to serialize a non-serializable object then we will get RuntimeException saying "***NotSerializableException***".

Transient keyword

- ‘transient’ is the modifier applicable only for variables.
- While performing serialization if we don't want to save the value of a particular variable to meet security constraints such type of variable , then we should declare that variable with "transient" keyword.
- At the time of serialization JVM ignores the original value of transient variable and save default value to the file.