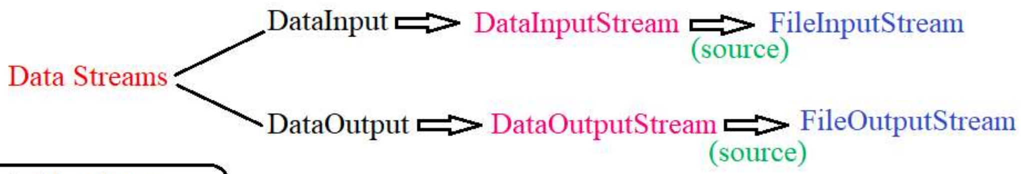
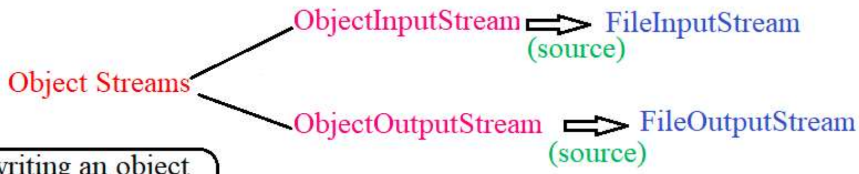
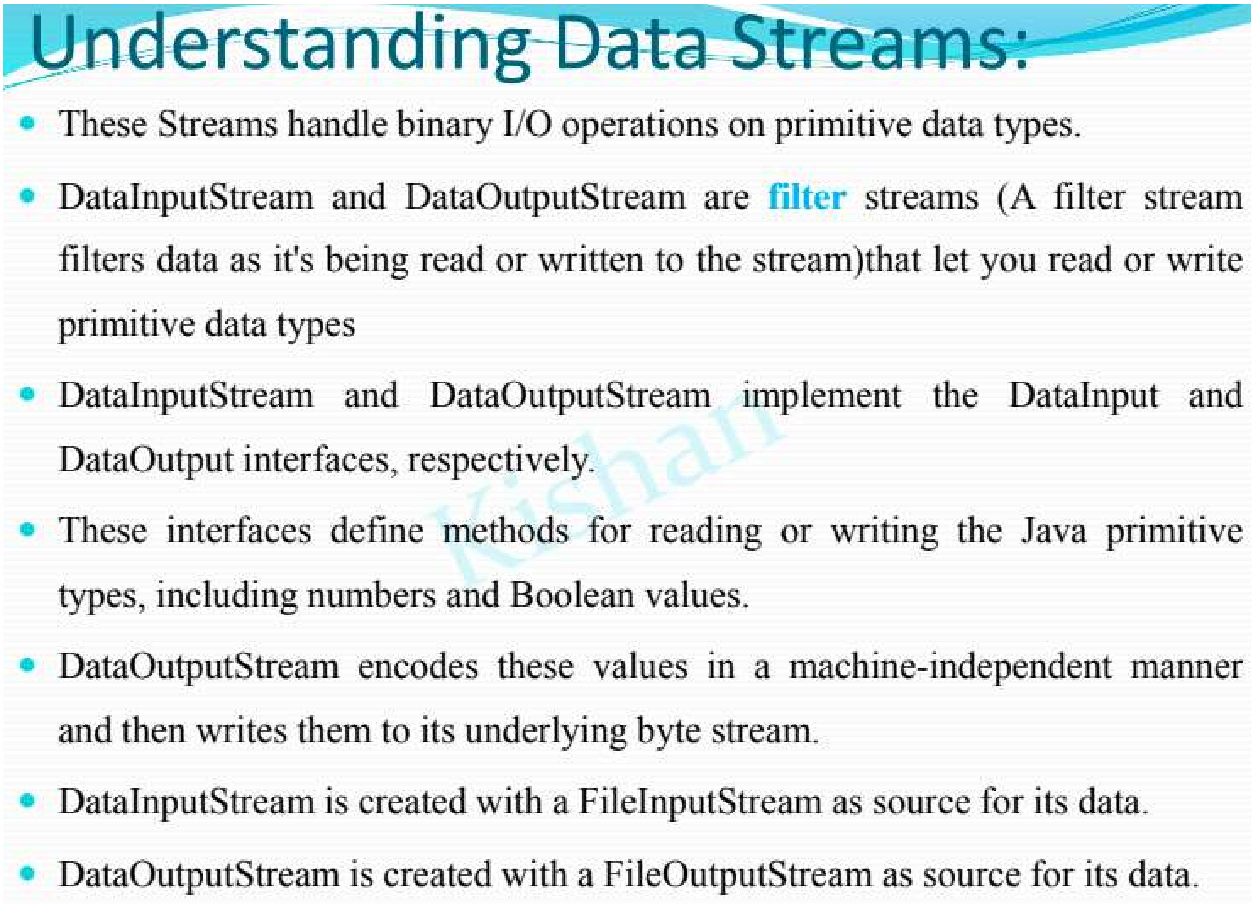


These are used to transfer primitive datatypes in a secure manner. Data Streams are also known as Filter Streams.

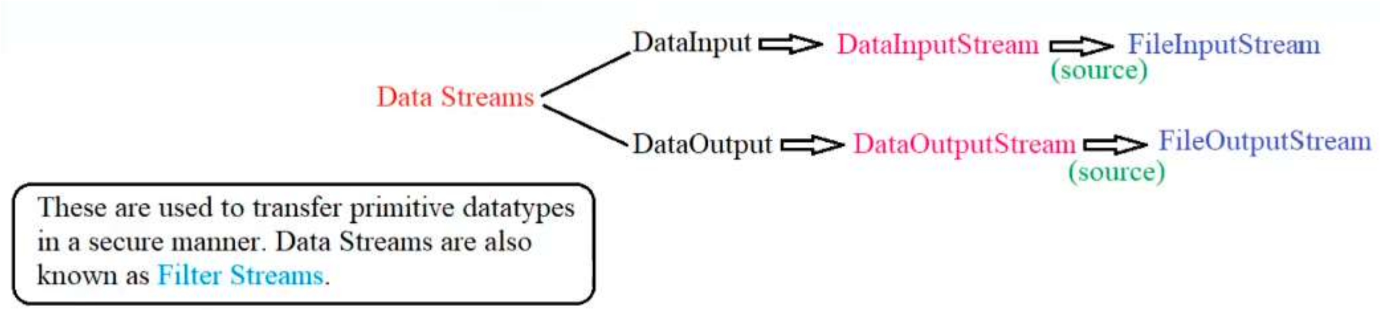


Serialization is a process of writing an object into a file, inorder to perform this serialization we need to use Object streams



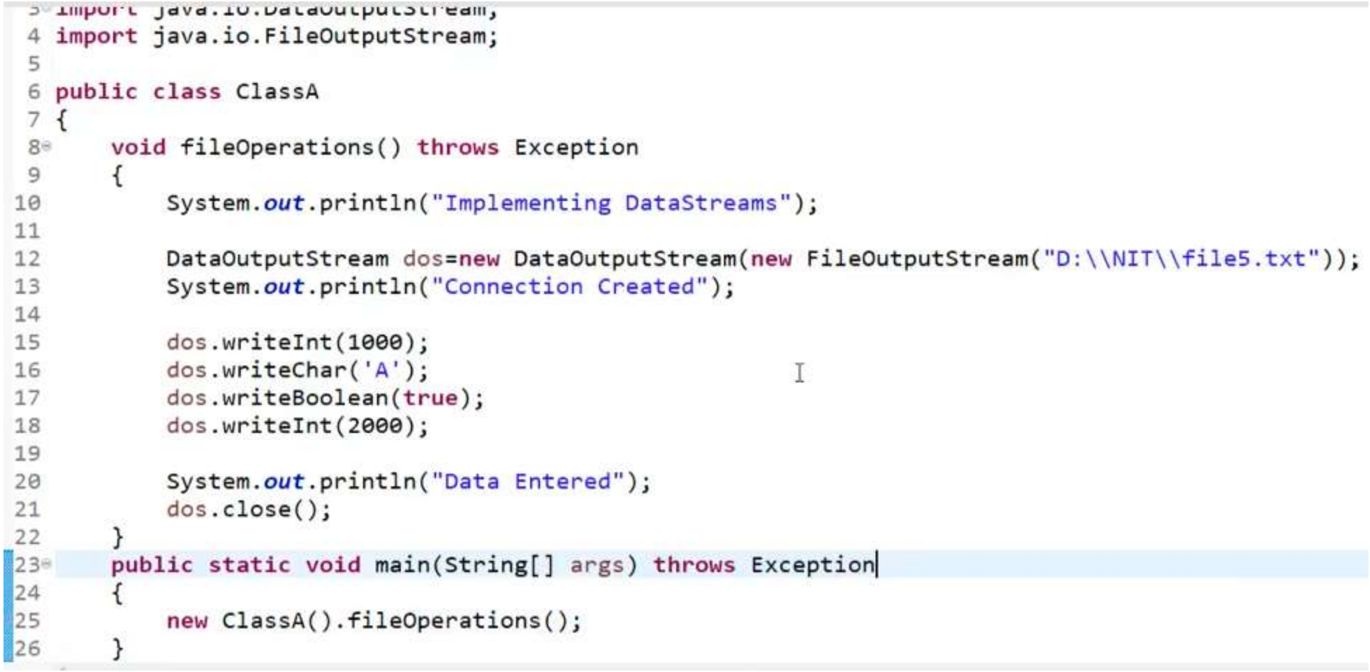
Understanding Data Streams:

- 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.

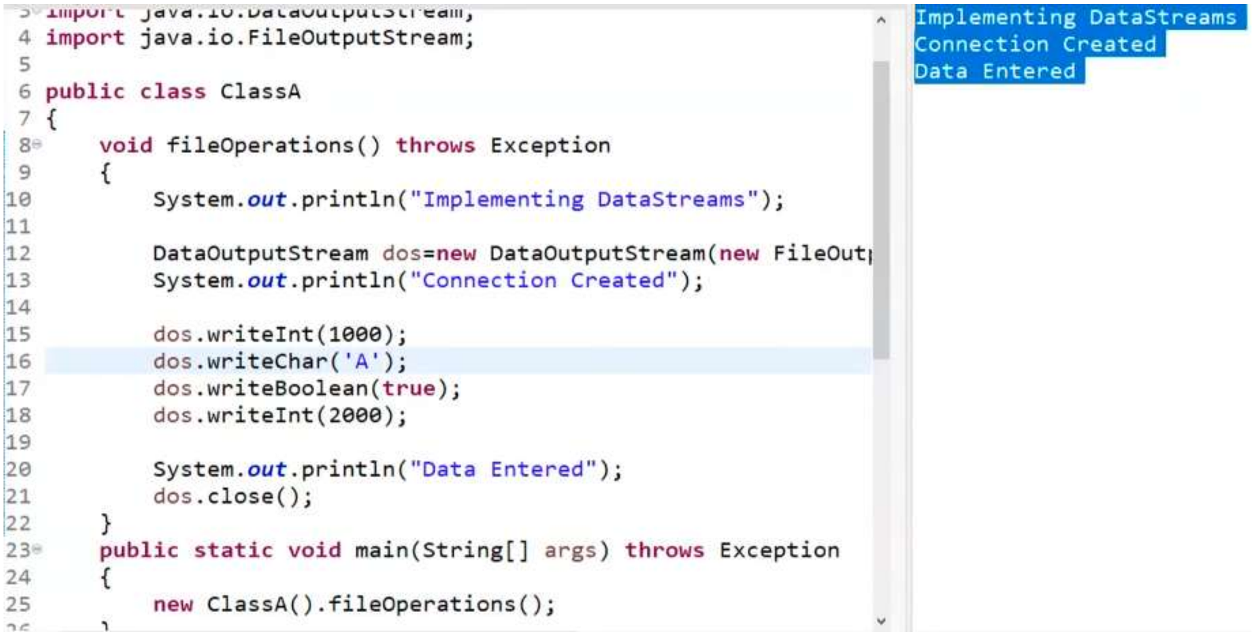


These are used to transfer primitive datatypes in a secure manner. Data Streams are also known as Filter Streams.

DataInpt and DataOutput are interfaces

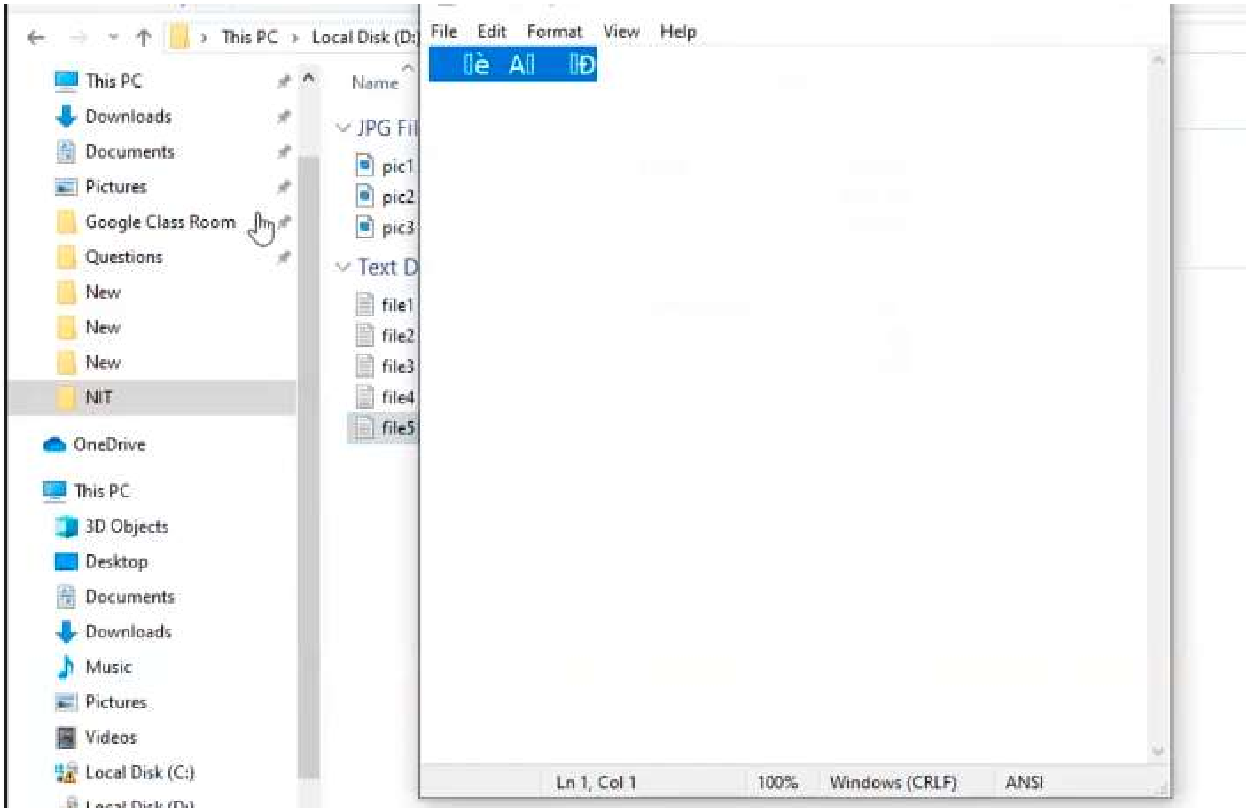


```java
import java.io.DataOutputStream;
import java.io.FileOutputStream;

public class ClassA
{
    void fileOperations() throws Exception
    {
        System.out.println("Implementing DataStreams");

        DataOutputStream dos=new DataOutputStream(new FileOutputStream("D:\\NIT\\file5.txt"));
        System.out.println("Connection Created");

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();
    }
    public static void main(String[] args) throws Exception
    {
        new ClassA().fileOperations();
    }
```

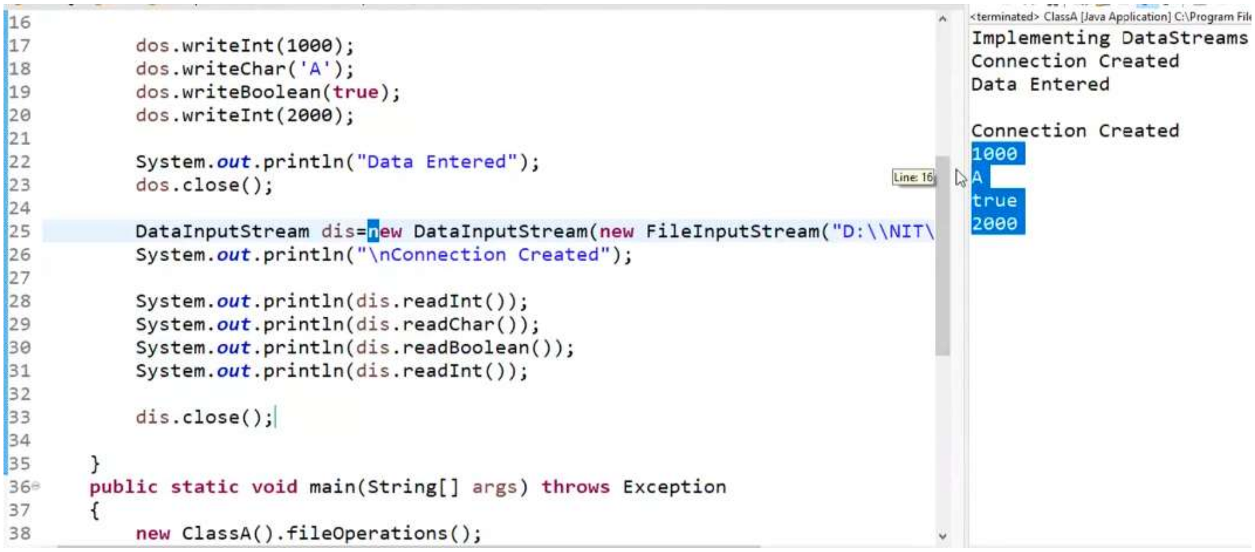


```java
import java.io.DataOutputStream;
import java.io.FileOutputStream;

public class ClassA
{
    void fileOperations() throws Exception
    {
        System.out.println("Implementing DataStreams");

        DataOutputStream dos=new DataOutputStream(new FileOut
        System.out.println("Connection Created");

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();
    }
    public static void main(String[] args) throws Exception
    {
        new ClassA().fileOperations();
    }
```

```
Implementing DataStreams
Connection Created
Data Entered
```

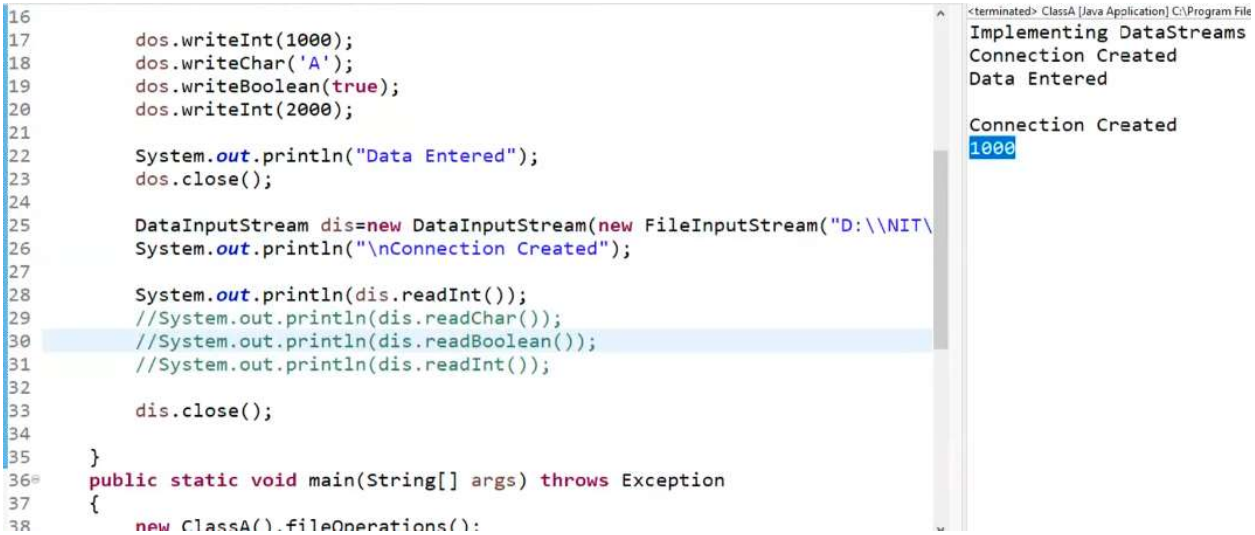


```
▯è A▯  ▯Ð
```



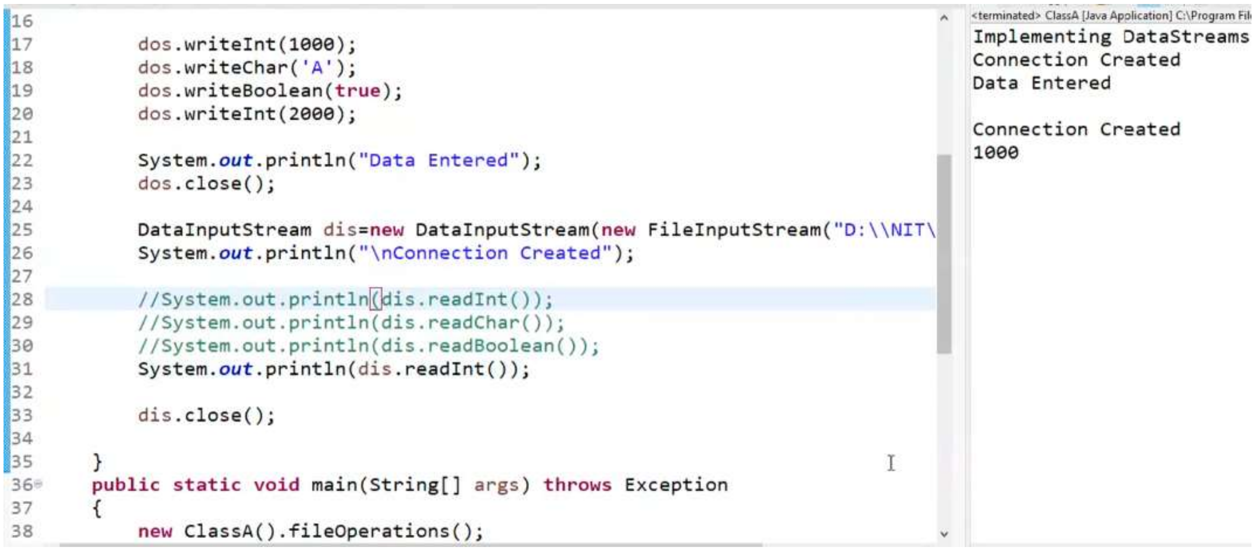
```java

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();

        DataInputStream dis=new DataInputStream(new FileInputStream("D:\\NIT\
        System.out.println("\nConnection Created");

        System.out.println(dis.readInt());
        System.out.println(dis.readChar());
        System.out.println(dis.readBoolean());
        System.out.println(dis.readInt());

        dis.close();

    }
    public static void main(String[] args) throws Exception
    {
        new ClassA().fileOperations();
```

```
<terminated> ClassA [Java Application] C:\Program Fil
Implementing DataStreams
Connection Created
Data Entered

Connection Created
1000
A
true
2000
```



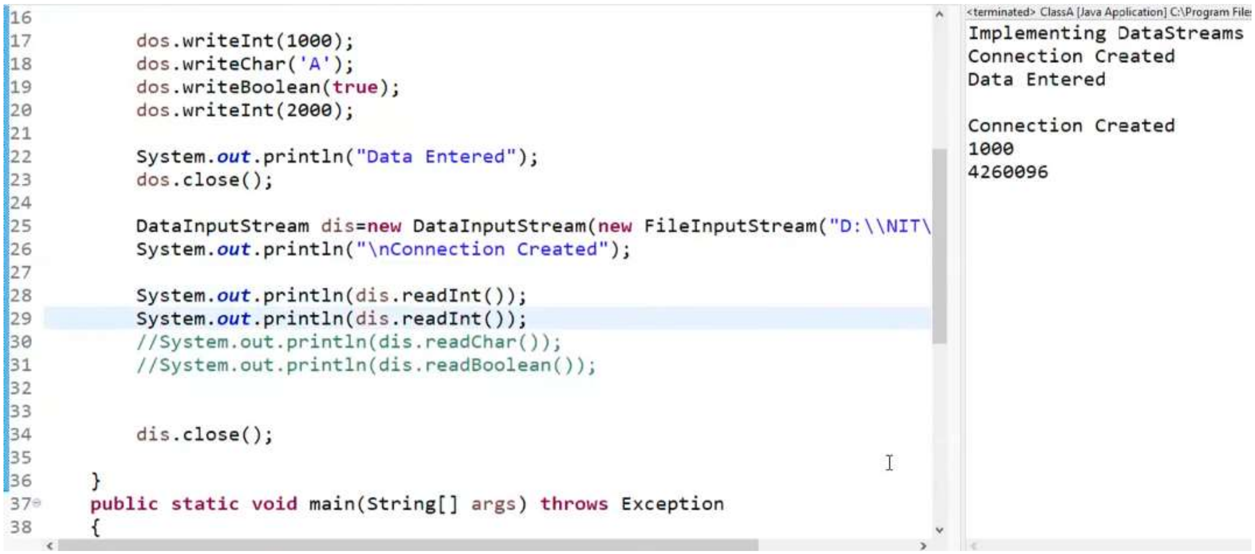
```java

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();

        DataInputStream dis=new DataInputStream(new FileInputStream("D:\\NIT\
        System.out.println("\nConnection Created");

        System.out.println(dis.readInt());
        //System.out.println(dis.readChar());
        //System.out.println(dis.readBoolean());
        //System.out.println(dis.readInt());

        dis.close();

    }
    public static void main(String[] args) throws Exception
    {
        new ClassA().fileOperations();
```

```
<terminated> ClassA [Java Application] C:\Program File
Implementing DataStreams
Connection Created
Data Entered

Connection Created
1000
```



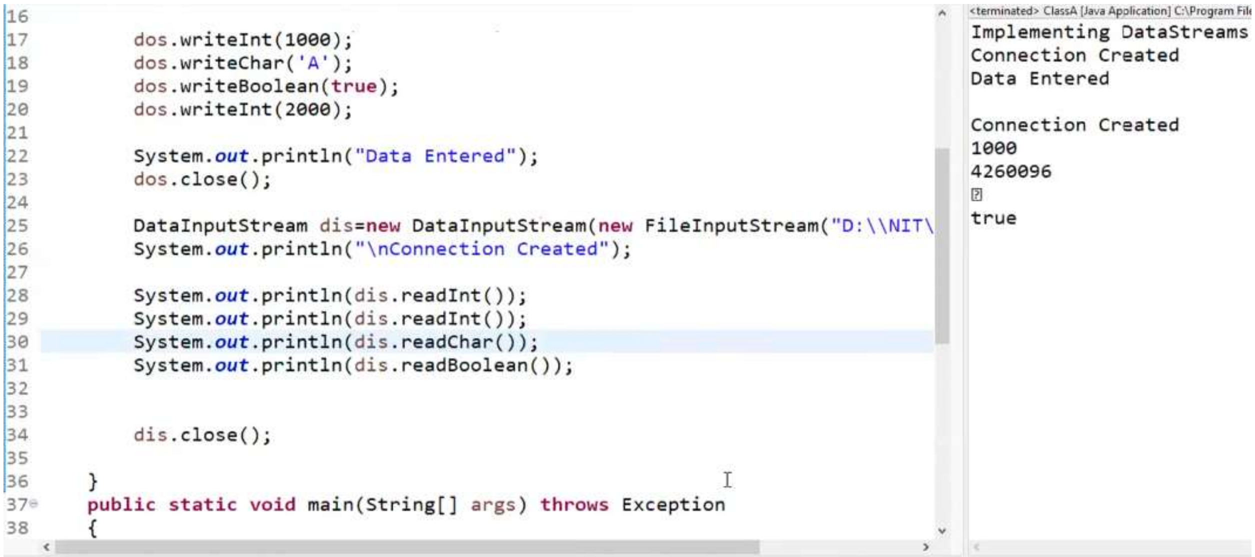
```java

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();

        DataInputStream dis=new DataInputStream(new FileInputStream("D:\\NIT\
        System.out.println("\nConnection Created");

        //System.out.println(dis.readInt());
        //System.out.println(dis.readChar());
        //System.out.println(dis.readBoolean());
        System.out.println(dis.readInt());

        dis.close();

    }
    public static void main(String[] args) throws Exception
    {
        new ClassA().fileOperations();
```

```
<terminated> ClassA [Java Application] C:\Program Fil
Implementing DataStreams
Connection Created
Data Entered

Connection Created
1000
```



```java

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();

        DataInputStream dis=new DataInputStream(new FileInputStream("D:\\NIT\
        System.out.println("\nConnection Created");

        System.out.println(dis.readInt());
        System.out.println(dis.readInt());
        //System.out.println(dis.readChar());
        //System.out.println(dis.readBoolean());


        dis.close();

    }
    public static void main(String[] args) throws Exception
    {
```

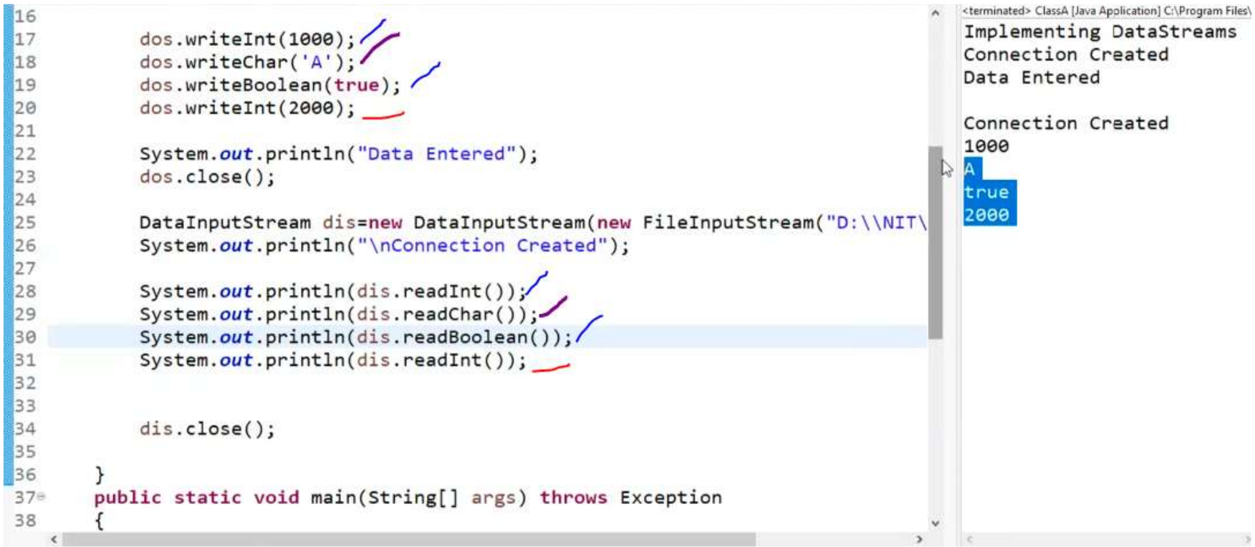
```
Implementing DataStreams
Connection Created
Data Entered

Connection Created
1000
4260096
```



```java

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();

        DataInputStream dis=new DataInputStream(new FileInputStream("D:\\NIT\
        System.out.println("\nConnection Created");

        System.out.println(dis.readInt());
        System.out.println(dis.readInt());
        System.out.println(dis.readChar());
        System.out.println(dis.readBoolean());


        dis.close();

    }
    public static void main(String[] args) throws Exception
    {
```

```
Implementing DataStreams
Connection Created
Data Entered

Connection Created
1000
4260096
⍰
true
```

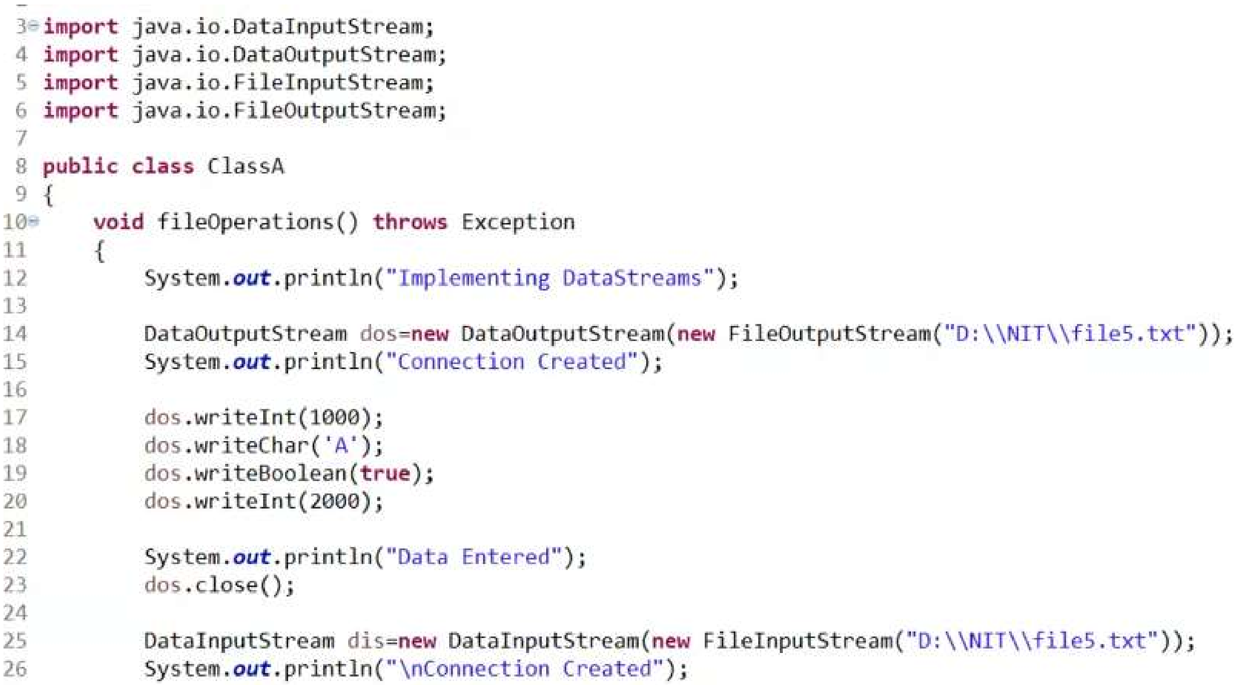
Whenever we are dealing with the data streams definitely, we should maintain order otherwise we will get inconsistent results.



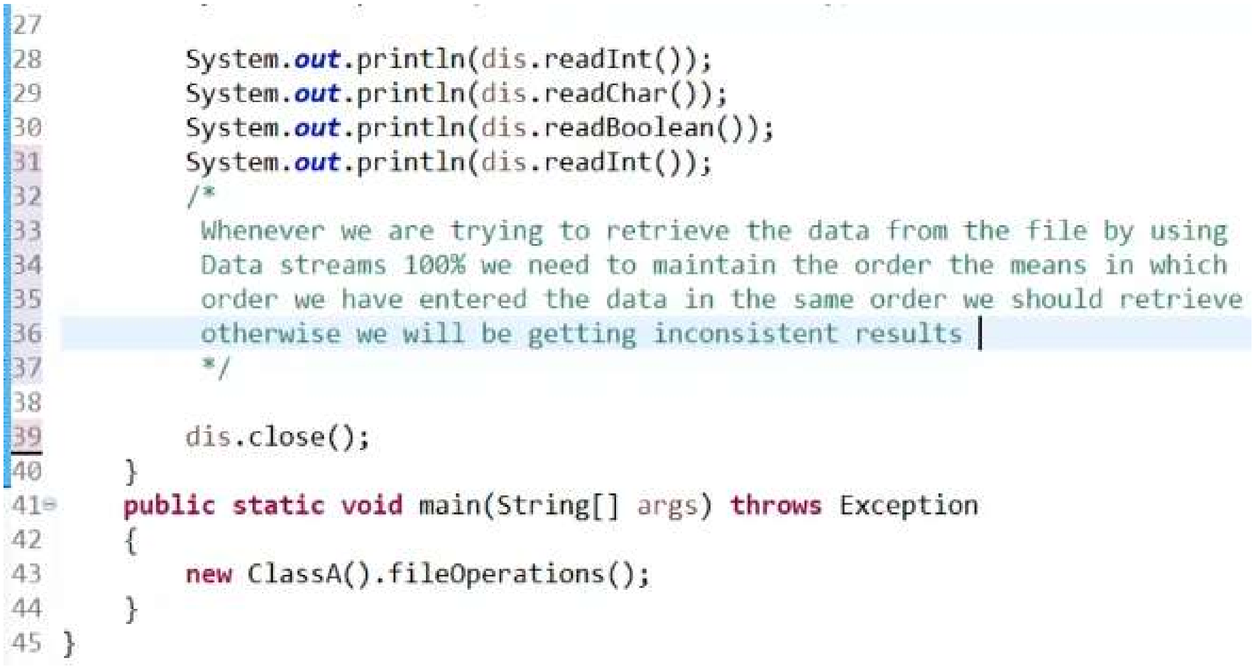
```
        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();

        DataInputStream dis=new DataInputStream(new FileInputStream("D:\\NIT\
        System.out.println("\nConnection Created");

        System.out.println(dis.readInt());
        System.out.println(dis.readChar());
        System.out.println(dis.readBoolean());
        System.out.println(dis.readInt());


        dis.close();

    }
    public static void main(String[] args) throws Exception
    {
```

```
<terminated> ClassA [Java Application] C:\Program Files\
Implementing DataStreams
Connection Created
Data Entered

Connection Created
1000
A
true
2000
```

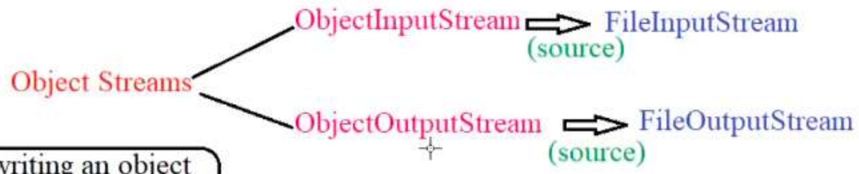
If we are talking about 100 variables then how can we remember the order, whenever we are dealing with datatype streams, we should go with one data type.



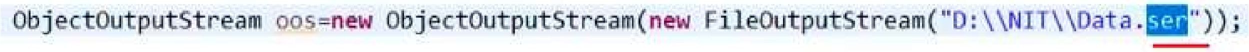
```
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.FileInputStream;
import java.io.FileOutputStream;

public class ClassA
{
    void fileOperations() throws Exception
    {
        System.out.println("Implementing DataStreams");

        DataOutputStream dos=new DataOutputStream(new FileOutputStream("D:\\NIT\\file5.txt"));
        System.out.println("Connection Created");

        dos.writeInt(1000);
        dos.writeChar('A');
        dos.writeBoolean(true);
        dos.writeInt(2000);

        System.out.println("Data Entered");
        dos.close();

        DataInputStream dis=new DataInputStream(new FileInputStream("D:\\NIT\\file5.txt"));
        System.out.println("\nConnection Created");
```



```
        System.out.println(dis.readInt());
        System.out.println(dis.readChar());
        System.out.println(dis.readBoolean());
        System.out.println(dis.readInt());
        /*
         Whenever we are trying to retrieve the data from the file by using
         Data streams 100% we need to maintain the order the means in which
         order we have entered the data in the same order we should retrieve
         otherwise we will be getting inconsistent results 
         */

        dis.close();
    }
    public static void main(String[] args) throws Exception
    {
        new ClassA().fileOperations();
    }
}
```

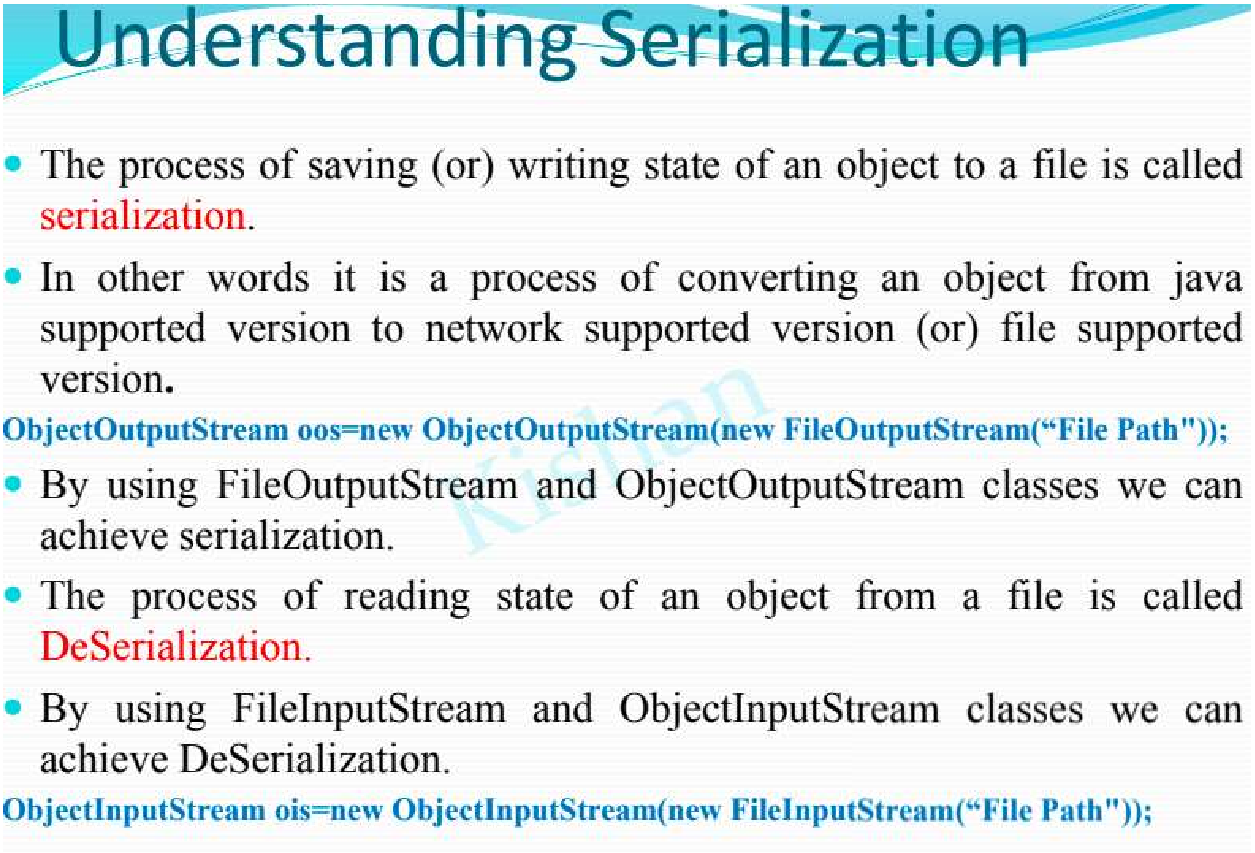


Serialization is a process of writing an object into a file, inorder to perform this serialization we need to use Object streams



```
ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStream("D:\\NIT\\Data.ser"));
```

When we are writing an object into a file the file extension must be in serializable



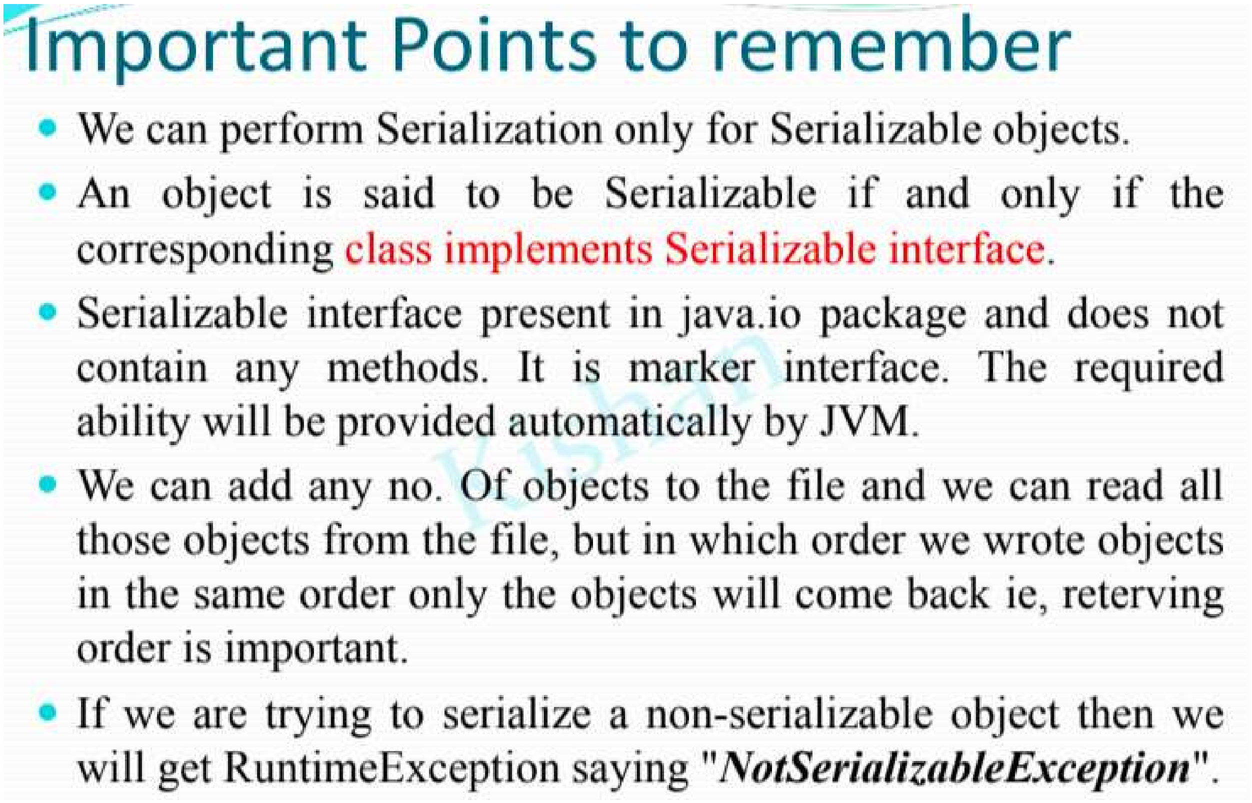
Understanding Serialization

- 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"));
```

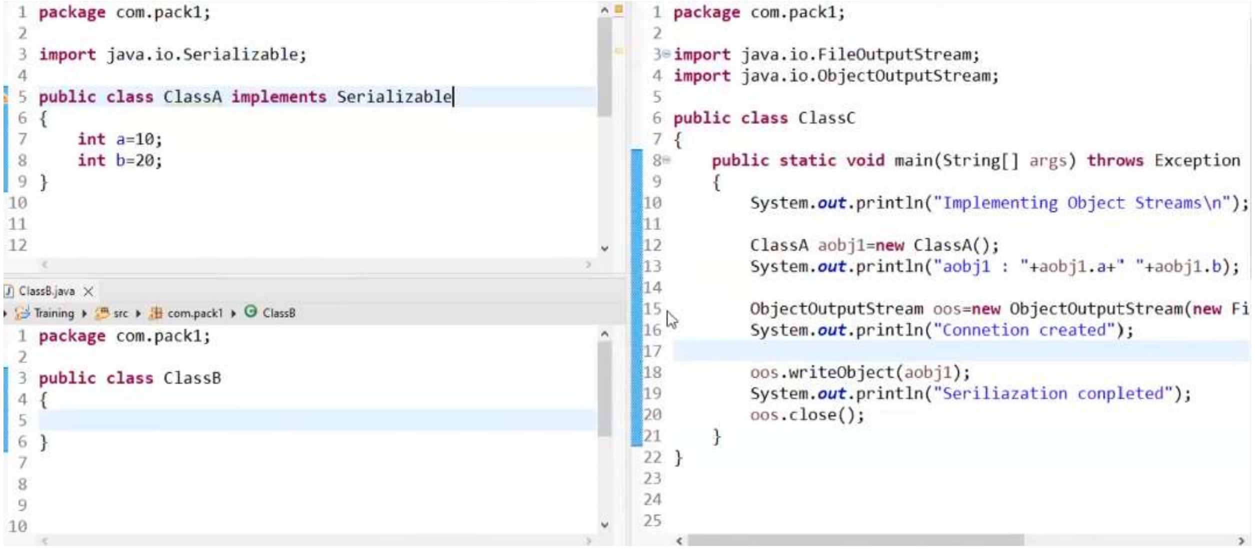
- By using FileOutputStream and ObjectOutputStream classes we can achieve serialization.
- The process of reading state of an object from a file is called DeSerialization.
- By using FileInputStream and ObjectInputStream classes we can achieve DeSerialization.

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



Important Points to remember

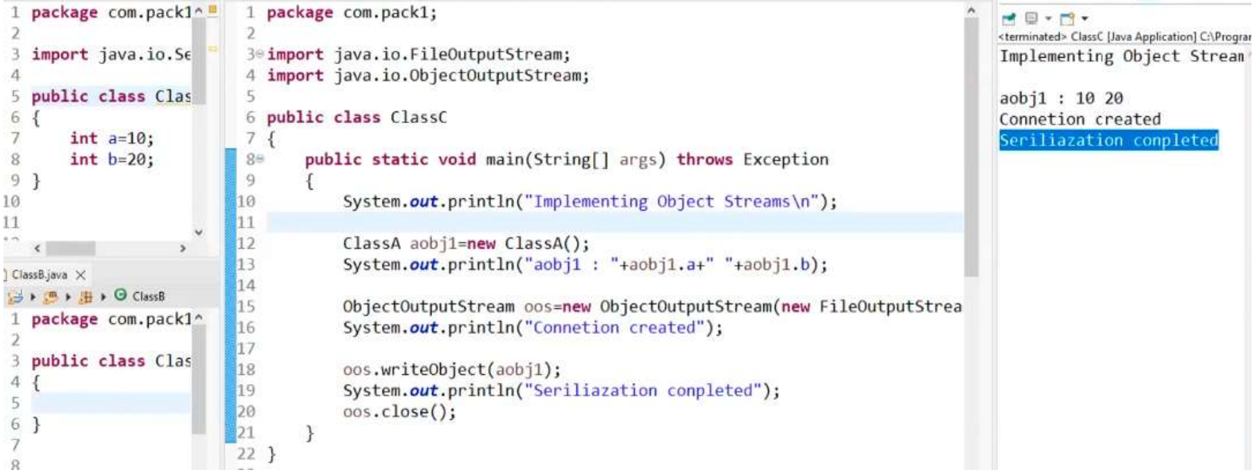
- We can perform Serialization only for Serializable objects.
- An object is said to be Serializable if and only if the corresponding class implements Serializable interface.
- Serializable interface present in java.io package and does not contain any methods. It is marker interface. The required ability 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***".



```java
package com.pack1;

import java.io.Serializable;

public class ClassA implements Serializable
{
    int a=10;
    int b=20;
}
```

```java
package com.pack1;

public class ClassB
{

}
```

```java
package com.pack1;

import java.io.FileOutputStream;
import java.io.ObjectOutputStream;

public class ClassC
{
    public static void main(String[] args) throws Exception
    {
        System.out.println("Implementing Object Streams\n");

        ClassA aobj1=new ClassA();
        System.out.println("aobj1 : "+aobj1.a+" "+aobj1.b);

        ObjectOutputStream oos=new ObjectOutputStream(new Fi
        System.out.println("Connetion created");

        oos.writeObject(aobj1);
        System.out.println("Seriliazation conpleted");
        oos.close();
    }
}
```

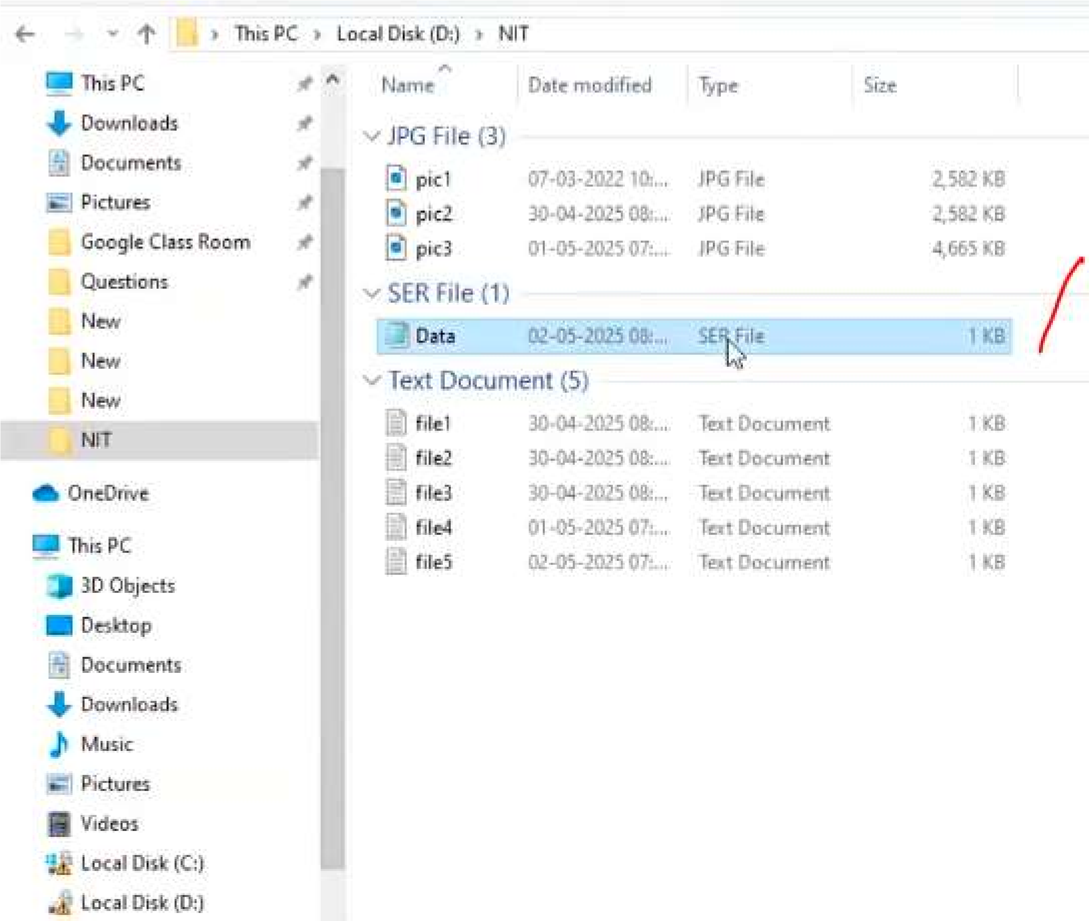


```java
package com.pack1

import java.io.Se

public class Clas
{
    int a=10;
    int b=20;
}
```

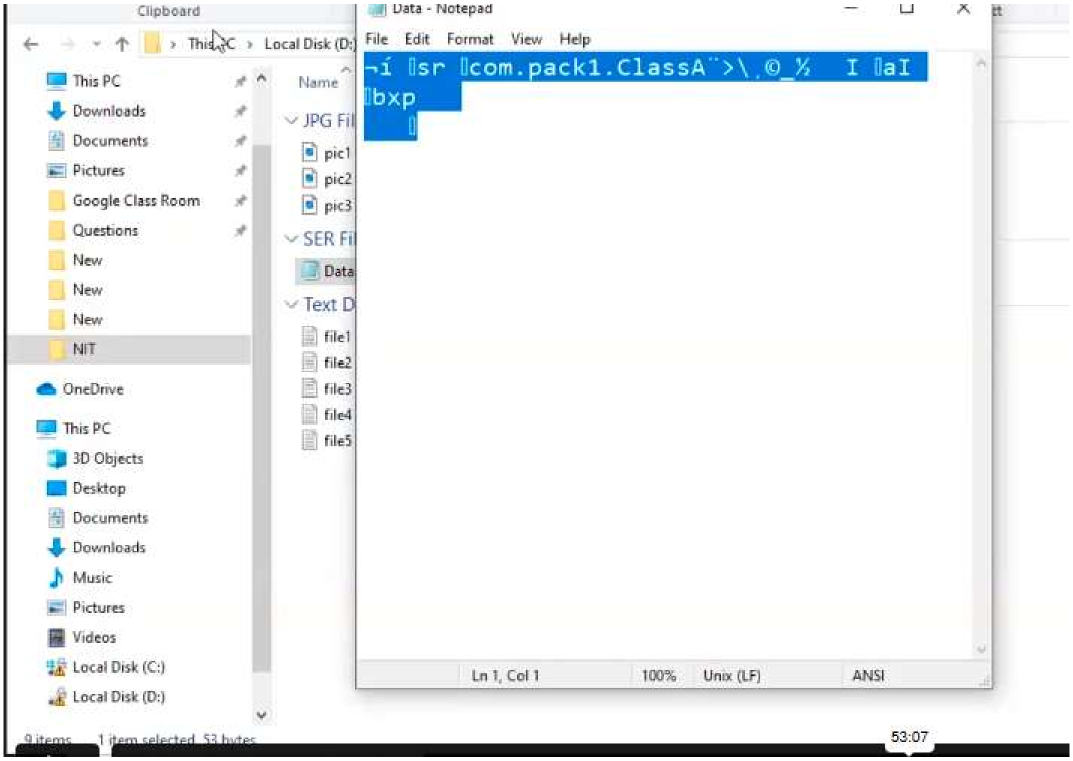
```java
package com.pack1

public class Clas
{

}
```

```java
package com.pack1;

import java.io.FileOutputStream;
import java.io.ObjectOutputStream;

public class ClassC
{
    public static void main(String[] args) throws Exception
    {
        System.out.println("Implementing Object Streams\n");

        ClassA aobj1=new ClassA();
        System.out.println("aobj1 : "+aobj1.a+" "+aobj1.b);

        ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStrea
        System.out.println("Connetion created");

        oos.writeObject(aobj1);
        System.out.println("Seriliazation conpleted");
        oos.close();
    }
}
```

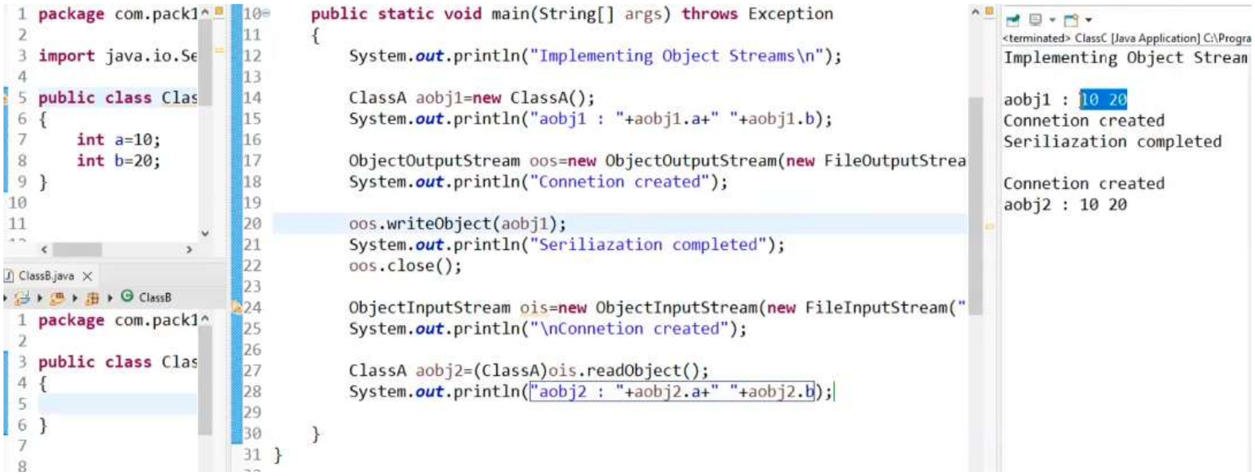
```
Implementing Object Strean

aobj1 : 10 20
Connetion created
Seriliazation conpleted
```



Name	Date modified	Type	Size
JPG File (3)			
pic1	07-03-2022 10:...	JPG File	2,582 KB
pic2	30-04-2025 08:...	JPG File	2,582 KB
pic3	01-05-2025 07:...	JPG File	4,665 KB
SER File (1)			
Data	02-05-2025 08:...	SER File	1 KB
Text Document (5)			
file1	30-04-2025 08:...	Text Document	1 KB
file2	30-04-2025 08:...	Text Document	1 KB
file3	30-04-2025 08:...	Text Document	1 KB
file4	01-05-2025 07:...	Text Document	1 KB
file5	02-05-2025 07:...	Text Document	1 KB



```
¬í sr com.pack1.ClassA¨>\¸©_½  I aI 
bxp
    
```



```
package com.pack1

import java.io.Se

public class Clas
{
    int a=10;
    int b=20;
}
```

ClassB.java

```
package com.pack1

public class Clas
{

}
```

```
    public static void main(String[] args) throws Exception
    {
        System.out.println("Implementing Object Streams\n");

        ClassA aobj1=new ClassA();
        System.out.println("aobj1 : "+aobj1.a+" "+aobj1.b);

        ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStrea
        System.out.println("Connetion created");

        oos.writeObject(aobj1);
        System.out.println("Seriliazation completed");
        oos.close();

        ObjectInputStream ois=new ObjectInputStream(new FileInputStream("
        System.out.println("\nConnetion created");

        ClassA aobj2=(ClassA)ois.readObject();
        System.out.println("aobj2 : "+aobj2.a+" "+aobj2.b);

    }
}
```

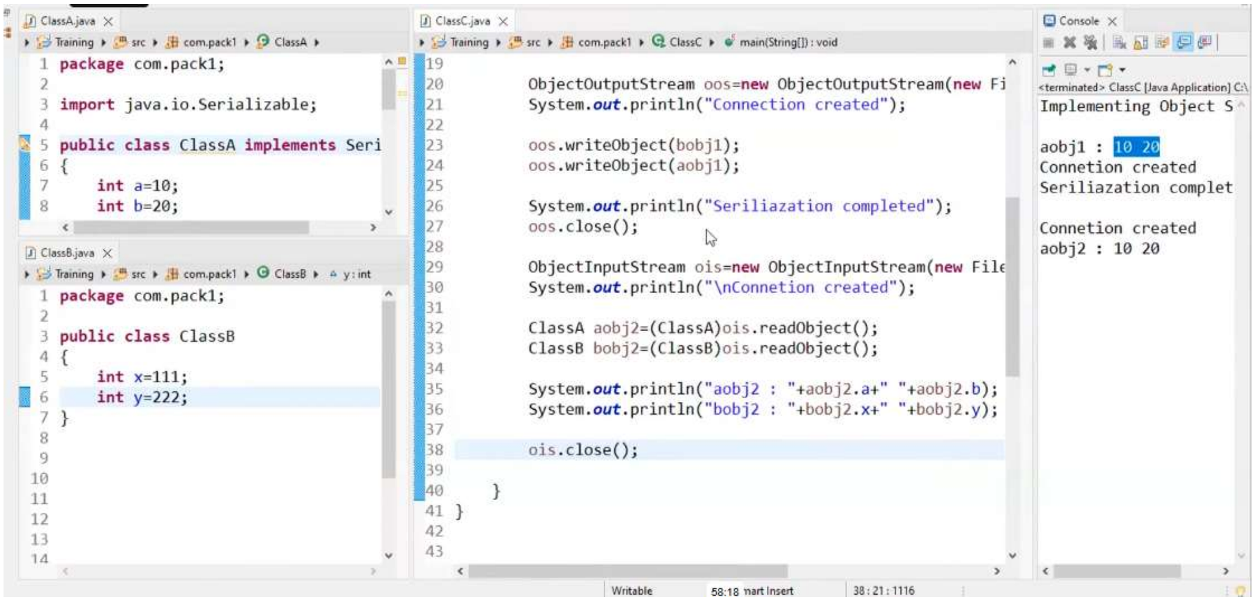
<terminated> ClassC [Java Application] C:\Progra

```
Implementing Object Stream

aobj1 : 10 20
Connetion created
Seriliazation completed

Connetion created
aobj2 : 10 20
```

De-serialization

Retrieving the object from a file

Can we pass multiple objects into file?



ClassA.java

Training › src › com.pack1 › ClassA ›

```
package com.pack1;

import java.io.Serializable;

public class ClassA implements Seri
{
    int a=10;
    int b=20;
```

ClassB.java

Training › src › com.pack1 › ClassB › y : int

```
package com.pack1;

public class ClassB
{
    int x=111;
    int y=222;
}
```

ClassC.java

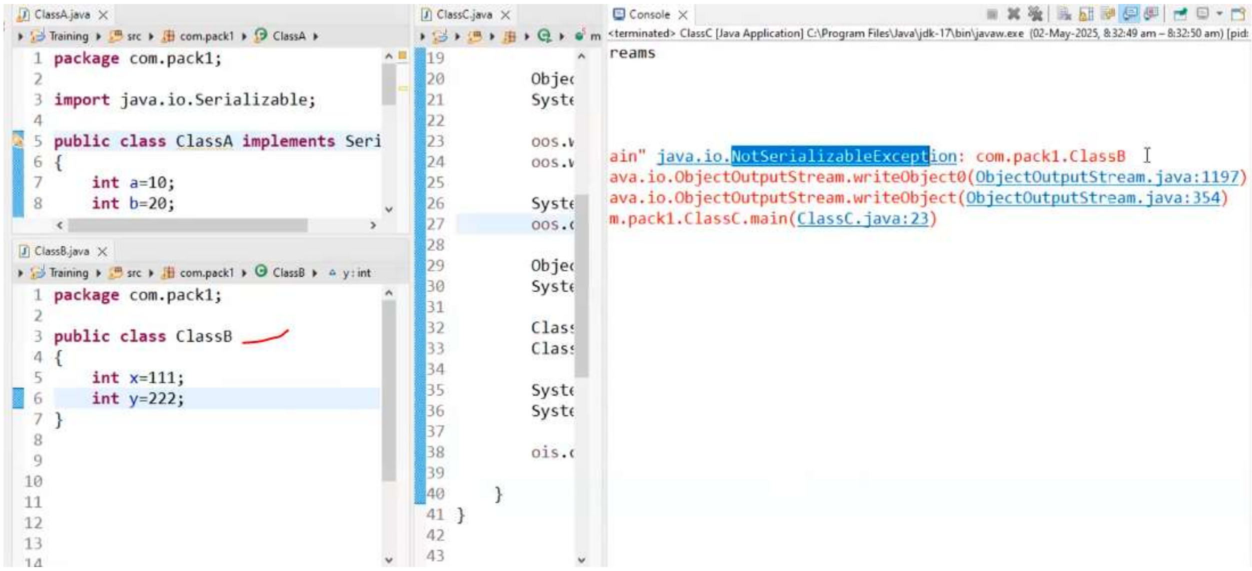
Training › src › com.pack1 › ClassC › main(String[]) : void

```

        ObjectOutputStream oos=new ObjectOutputStream(new Fi
        System.out.println("Connection created");

        oos.writeObject(bobj1);
        oos.writeObject(aobj1);

        System.out.println("Seriliazation completed");
        oos.close();

        ObjectInputStream ois=new ObjectInputStream(new File
        System.out.println("\nConnetion created");

        ClassA aobj2=(ClassA)ois.readObject();
        ClassB bobj2=(ClassB)ois.readObject();

        System.out.println("aobj2 : "+aobj2.a+" "+aobj2.b);
        System.out.println("bobj2 : "+bobj2.x+" "+bobj2.y);

        ois.close();

    }
}
```

Console

<terminated> ClassC [Java Application] C:\

```
Implementing Object S

aobj1 : 10 20
Connetion created
Seriliazation complet

Connetion created
aobj2 : 10 20
```



ClassA.java
Training › src › com.pack1 › ClassA

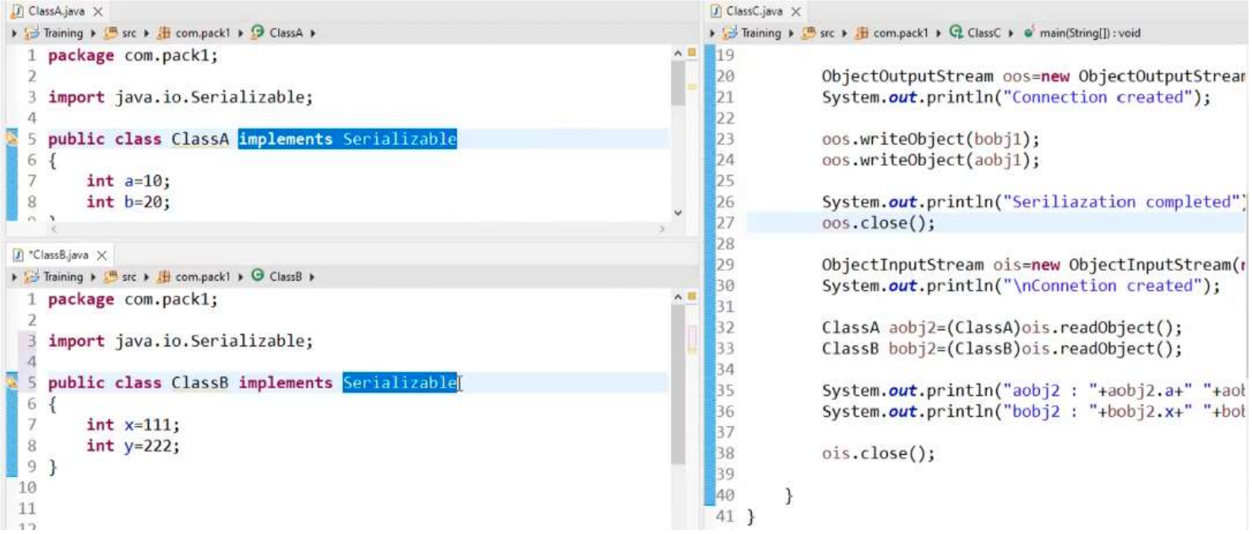
```java
package com.pack1;

import java.io.Serializable;

public class ClassA implements Seri
{
    int a=10;
    int b=20;
```

ClassB.java
Training › src › com.pack1 › ClassB › y : int

```java
package com.pack1;

public class ClassB
{
    int x=111;
    int y=222;
}
```

```
        Objec
        Syste

        oos.w
        oos.w

        Syste
        oos.c

        Objec
        Syste

        Class
        Class

        Syste
        Syste

        ois.c

    }
}
```

```
<terminated> ClassC [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe  (02-May-2025, 8:32:49 am – 8:32:50 am) [pid:
reams



ain" java.io.NotSerializableException: com.pack1.ClassB
ava.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1197)
ava.io.ObjectOutputStream.writeObject(ObjectOutputStream.java:354)
m.pack1.ClassC.main(ClassC.java:23)
```



ClassA.java
Training › src › com.pack1 › ClassA

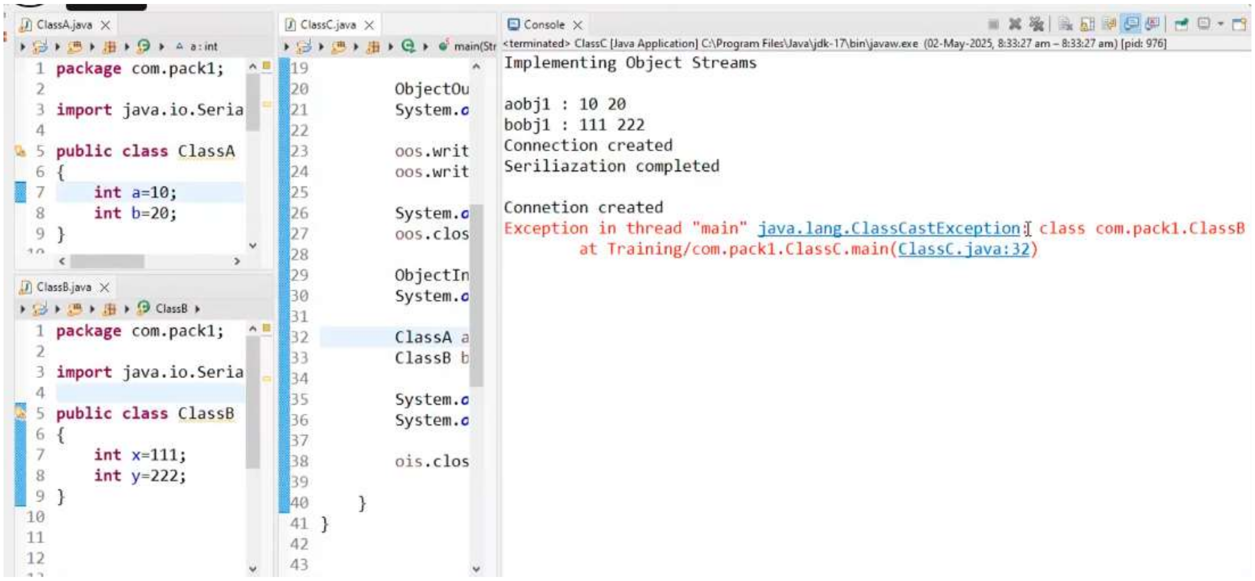
```java
package com.pack1;

import java.io.Serializable;

public class ClassA implements Serializable
{
    int a=10;
    int b=20;
```

*ClassB.java
Training › src › com.pack1 › ClassB

```java
package com.pack1;

import java.io.Serializable;

public class ClassB implements Serializable
{
    int x=111;
    int y=222;
}
```

ClassC.java
Training › src › com.pack1 › ClassC › main(String[]) : void

```java

        ObjectOutputStream oos=new ObjectOutputStrear
        System.out.println("Connection created");

        oos.writeObject(bobj1);
        oos.writeObject(aobj1);

        System.out.println("Seriliazation completed")
        oos.close();

        ObjectInputStream ois=new ObjectInputStream(r
        System.out.println("\nConnetion created");

        ClassA aobj2=(ClassA)ois.readObject();
        ClassB bobj2=(ClassB)ois.readObject();

        System.out.println("aobj2 : "+aobj2.a+" "+aot
        System.out.println("bobj2 : "+bobj2.x+" "+bot

        ois.close();

    }
}
```



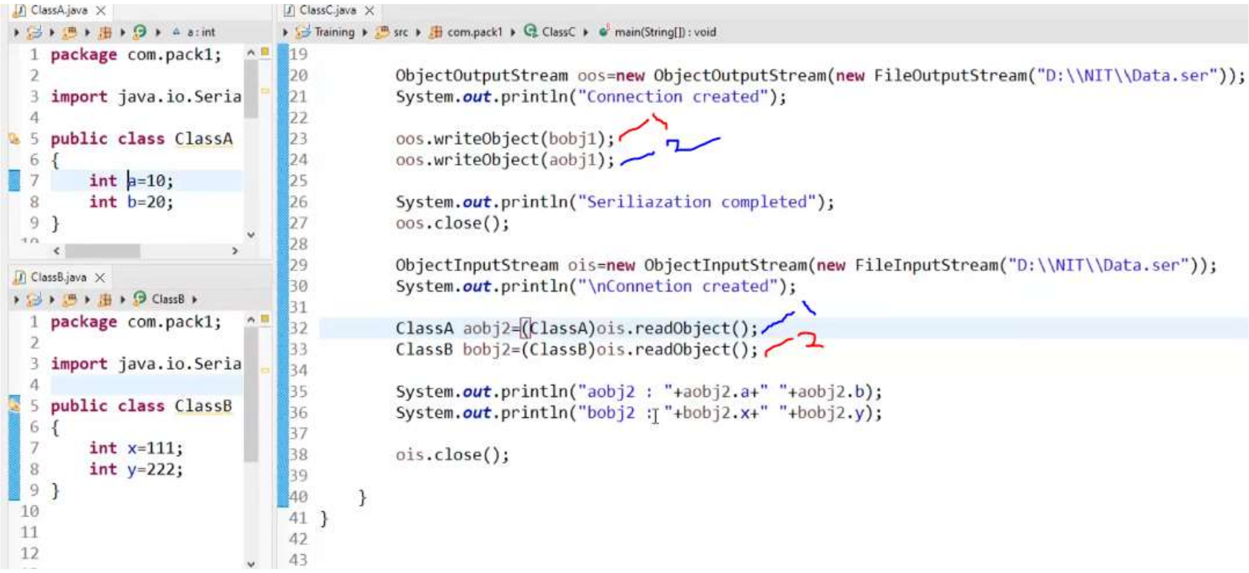
```
package com.pack1;

import java.io.Seria

public class ClassA
{
    int a=10;
    int b=20;
}
```

```
package com.pack1;

import java.io.Seria

public class ClassB
{
    int x=111;
    int y=222;
}
```

```
        ObjectOu
        System.o

        oos.writ
        oos.writ

        System.o
        oos.clos

        ObjectIn
        System.o

        ClassA a
        ClassB b

        System.o
        System.o

        ois.clos

    }
}
```

```
<terminated> ClassC [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe  (02-May-2025, 8:33:27 am – 8:33:27 am) [pid: 976]
Implementing Object Streams

aobj1 : 10 20
bobj1 : 111 222
Connection created
Seriliazation completed

Connetion created
Exception in thread "main" java.lang.ClassCastException: class com.pack1.ClassB
        at Training/com.pack1.ClassC.main(ClassC.java:32)
```

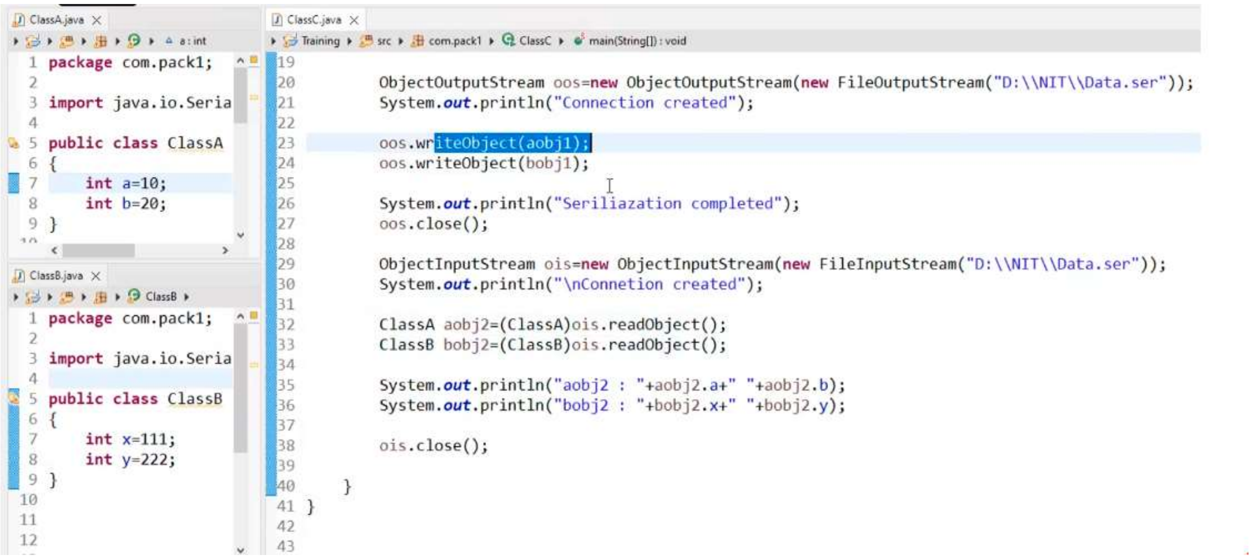
Because of the order



```
package com.pack1;

import java.io.Seria

public class ClassA
{
    int a=10;
    int b=20;
}
```

```
package com.pack1;

import java.io.Seria

public class ClassB
{
    int x=111;
    int y=222;
}
```

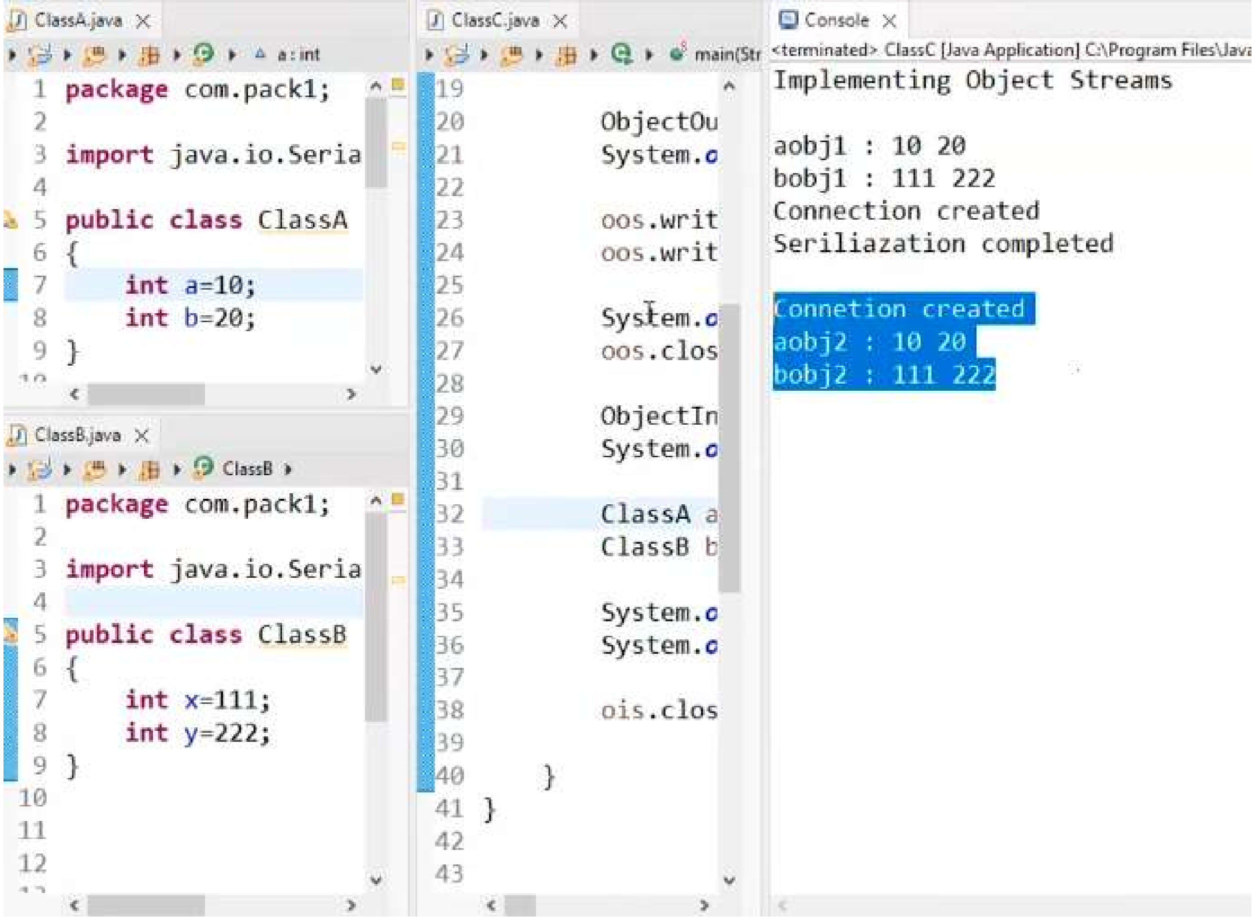
```
        ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStream("D:\\NIT\\Data.ser"));
        System.out.println("Connection created");

        oos.writeObject(bobj1);
        oos.writeObject(aobj1);

        System.out.println("Seriliazation completed");
        oos.close();

        ObjectInputStream ois=new ObjectInputStream(new FileInputStream("D:\\NIT\\Data.ser"));
        System.out.println("\nConnetion created");

        ClassA aobj2=(ClassA)ois.readObject();
        ClassB bobj2=(ClassB)ois.readObject();

        System.out.println("aobj2 : "+aobj2.a+" "+aobj2.b);
        System.out.println("bobj2 : "+bobj2.x+" "+bobj2.y);

        ois.close();

    }
}
```



```
package com.pack1;

import java.io.Seria

public class ClassA
{
	int a=10;
	int b=20;
}
```

```
		ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStream("D:\\NIT\\Data.ser"));
		System.out.println("Connection created");

		oos.writeObject(aobj1);
		oos.writeObject(bobj1);

		System.out.println("Seriliazation completed");
		oos.close();

		ObjectInputStream ois=new ObjectInputStream(new FileInputStream("D:\\NIT\\Data.ser"));
		System.out.println("\nConnetion created");

		ClassA aobj2=(ClassA)ois.readObject();
		ClassB bobj2=(ClassB)ois.readObject();

		System.out.println("aobj2 : "+aobj2.a+" "+aobj2.b);
		System.out.println("bobj2 : "+bobj2.x+" "+bobj2.y);

		ois.close();

	}
}
```

```
package com.pack1;

import java.io.Seria

public class ClassB
{
	int x=111;
	int y=222;
}
```

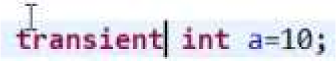


```
package com.pack1;

import java.io.Seria

public class ClassA
{
	int a=10;
	int b=20;
}
```

```
package com.pack1;

import java.io.Seria

public class ClassB
{
	int x=111;
	int y=222;
}
```

```
		ObjectOu
		System.o

		oos.writ
		oos.writ

		System.o
		oos.clos

		ObjectIn
		System.o

		ClassA a
		ClassB b

		System.o
		System.o

		ois.clos

	}
}
```

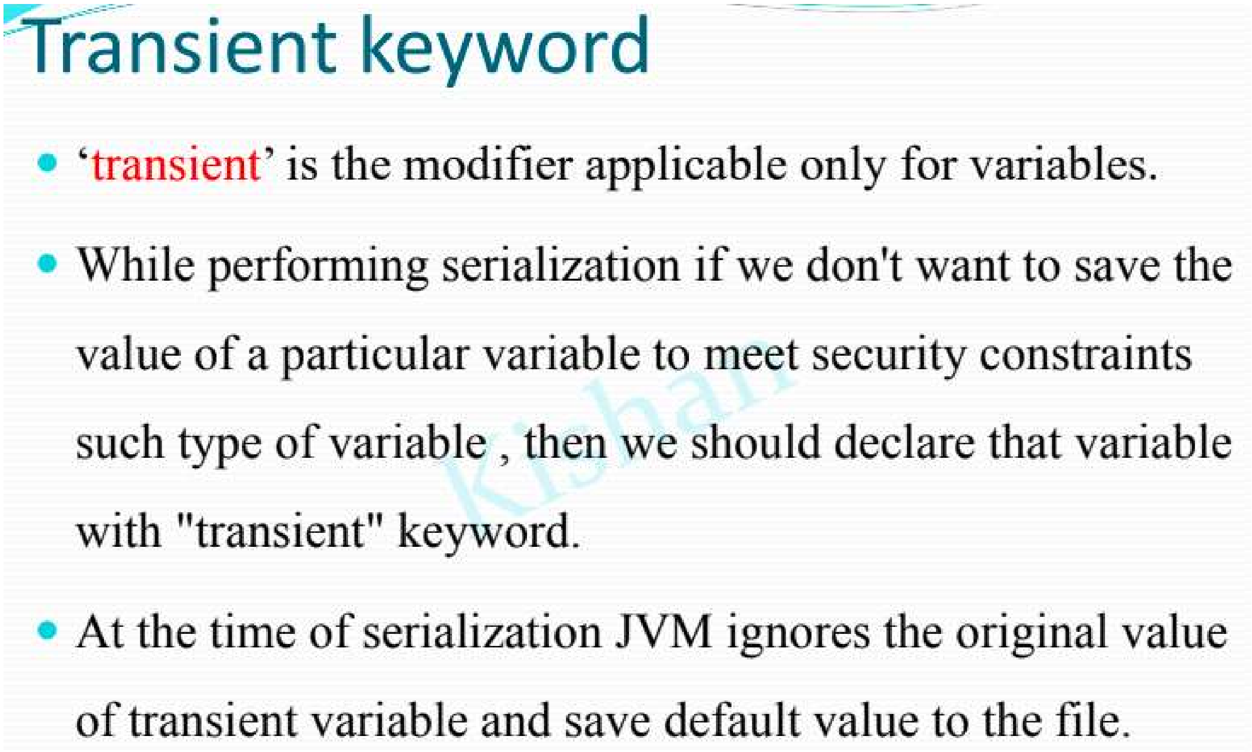
```
Implementing Object Streams

aobj1 : 10 20
bobj1 : 111 222
Connection created
Seriliazation completed

Connetion created
aobj2 : 10 20
bobj2 : 111 222
```



```
transient int a=10;
```

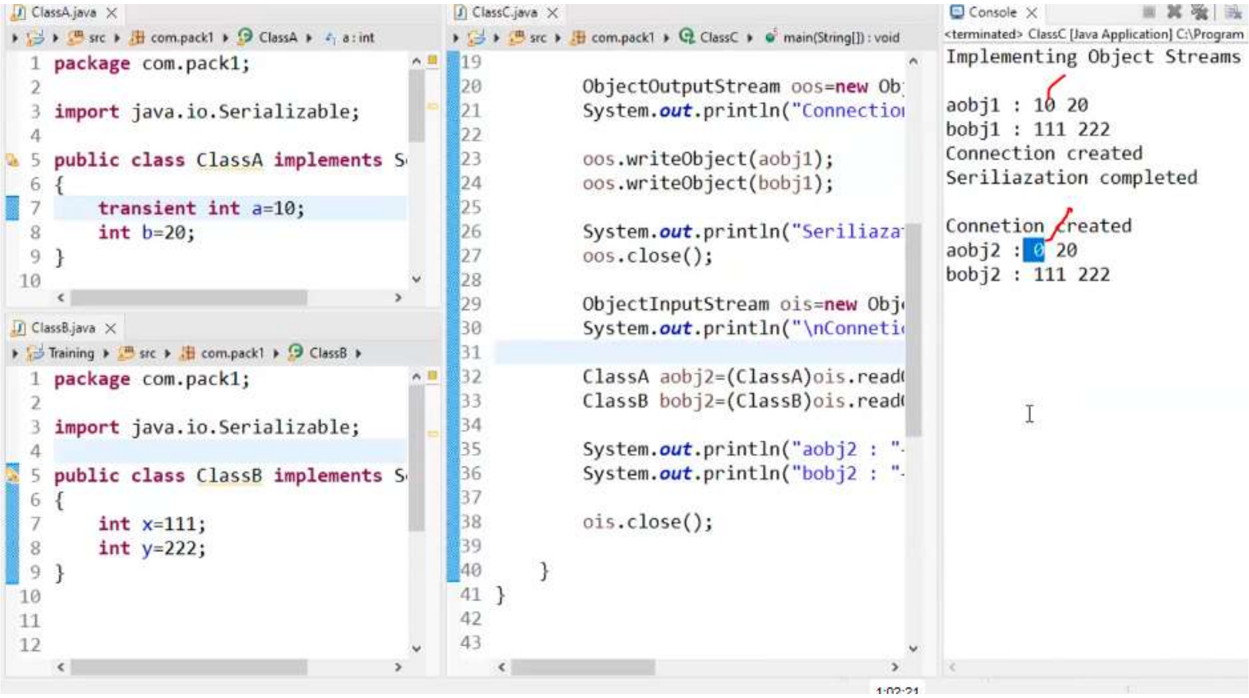
if we are declaring any variable as transient, if we are performing serialization, JVM will ignore the original value of that variable and only the default value of that variable is stored.

Transient key word is used for only variables not for methods



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.

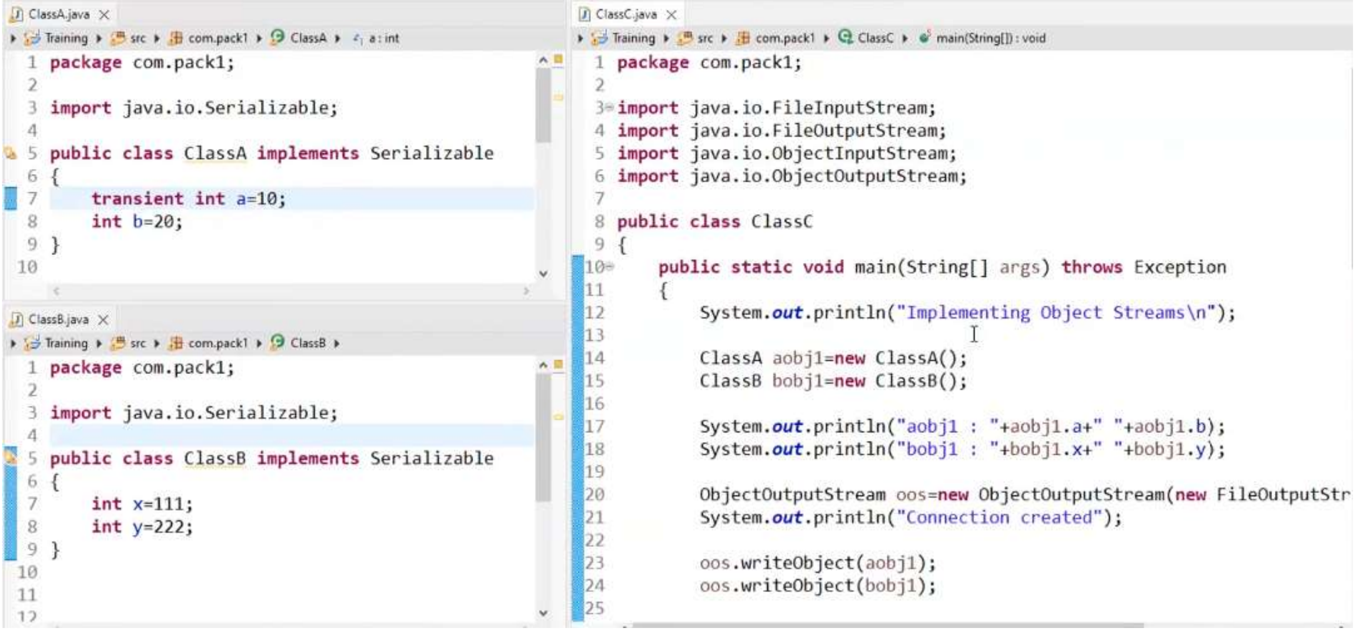


```java
package com.pack1;

import java.io.Serializable;

public class ClassA implements S
{
    transient int a=10;
    int b=20;
}
```

```java
package com.pack1;

import java.io.Serializable;

public class ClassB implements S
{
    int x=111;
    int y=222;
}
```

```java

        ObjectOutputStream oos=new Ob
        System.out.println("Connectio

        oos.writeObject(aobj1);
        oos.writeObject(bobj1);

        System.out.println("Seriliaza
        oos.close();

        ObjectInputStream ois=new Obj
        System.out.println("\nConneti

        ClassA aobj2=(ClassA)ois.read(
        ClassB bobj2=(ClassB)ois.read(

        System.out.println("aobj2 : "-
        System.out.println("bobj2 : "-

        ois.close();

    }
}
```

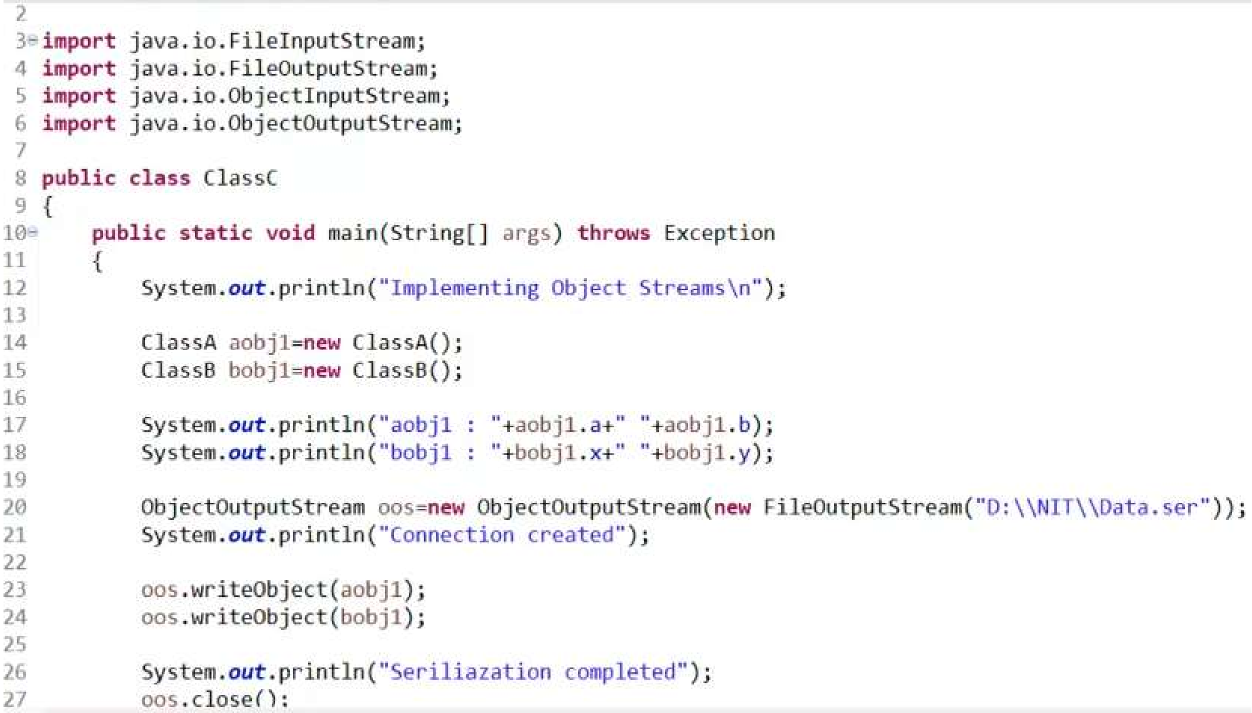
```
Implementing Object Streams

aobj1 : 10 20
bobj1 : 111 222
Connection created
Seriliazation completed

Connetion created
aobj2 : 0 20
bobj2 : 111 222
```



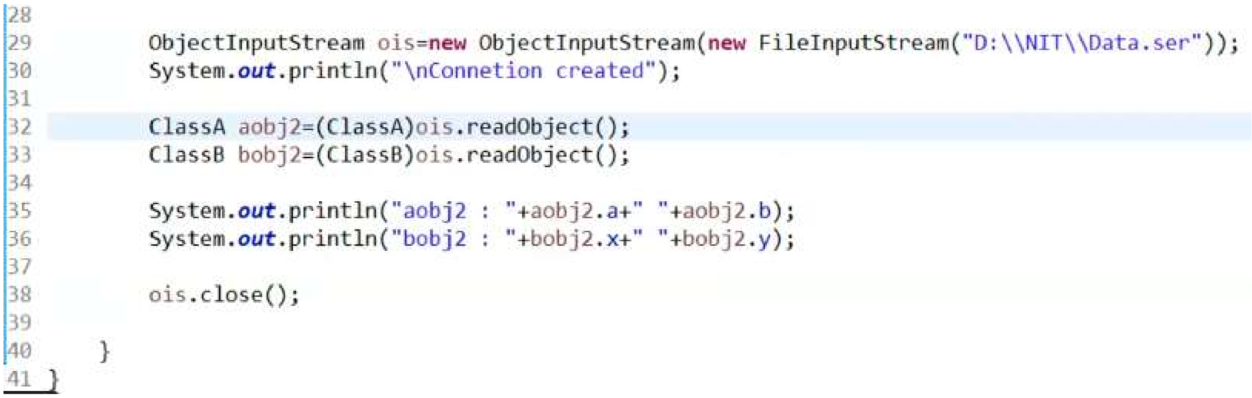
```java
package com.pack1;

import java.io.Serializable;

public class ClassA implements Serializable
{
    transient int a=10;
    int b=20;
}
```

```java
package com.pack1;

import java.io.Serializable;

public class ClassB implements Serializable
{
    int x=111;
    int y=222;
}
```

```java
package com.pack1;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;

public class ClassC
{
    public static void main(String[] args) throws Exception
    {
        System.out.println("Implementing Object Streams\n");

        ClassA aobj1=new ClassA();
        ClassB bobj1=new ClassB();

        System.out.println("aobj1 : "+aobj1.a+" "+aobj1.b);
        System.out.println("bobj1 : "+bobj1.x+" "+bobj1.y);

        ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStr
        System.out.println("Connection created");

        oos.writeObject(aobj1);
        oos.writeObject(bobj1);

```



```java

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;

public class ClassC
{
    public static void main(String[] args) throws Exception
    {
        System.out.println("Implementing Object Streams\n");

        ClassA aobj1=new ClassA();
        ClassB bobj1=new ClassB();

        System.out.println("aobj1 : "+aobj1.a+" "+aobj1.b);
        System.out.println("bobj1 : "+bobj1.x+" "+bobj1.y);

        ObjectOutputStream oos=new ObjectOutputStream(new FileOutputStream("D:\\NIT\\Data.ser"));
        System.out.println("Connection created");

        oos.writeObject(aobj1);
        oos.writeObject(bobj1);

        System.out.println("Seriliazation completed");
        oos.close();
```



```java

        ObjectInputStream ois=new ObjectInputStream(new FileInputStream("D:\\NIT\\Data.ser"));
        System.out.println("\nConnetion created");

        ClassA aobj2=(ClassA)ois.readObject();
        ClassB bobj2=(ClassB)ois.readObject();

        System.out.println("aobj2 : "+aobj2.a+" "+aobj2.b);
        System.out.println("bobj2 : "+bobj2.x+" "+bobj2.y);

        ois.close();

    }
}
```