High Level
Input and Output

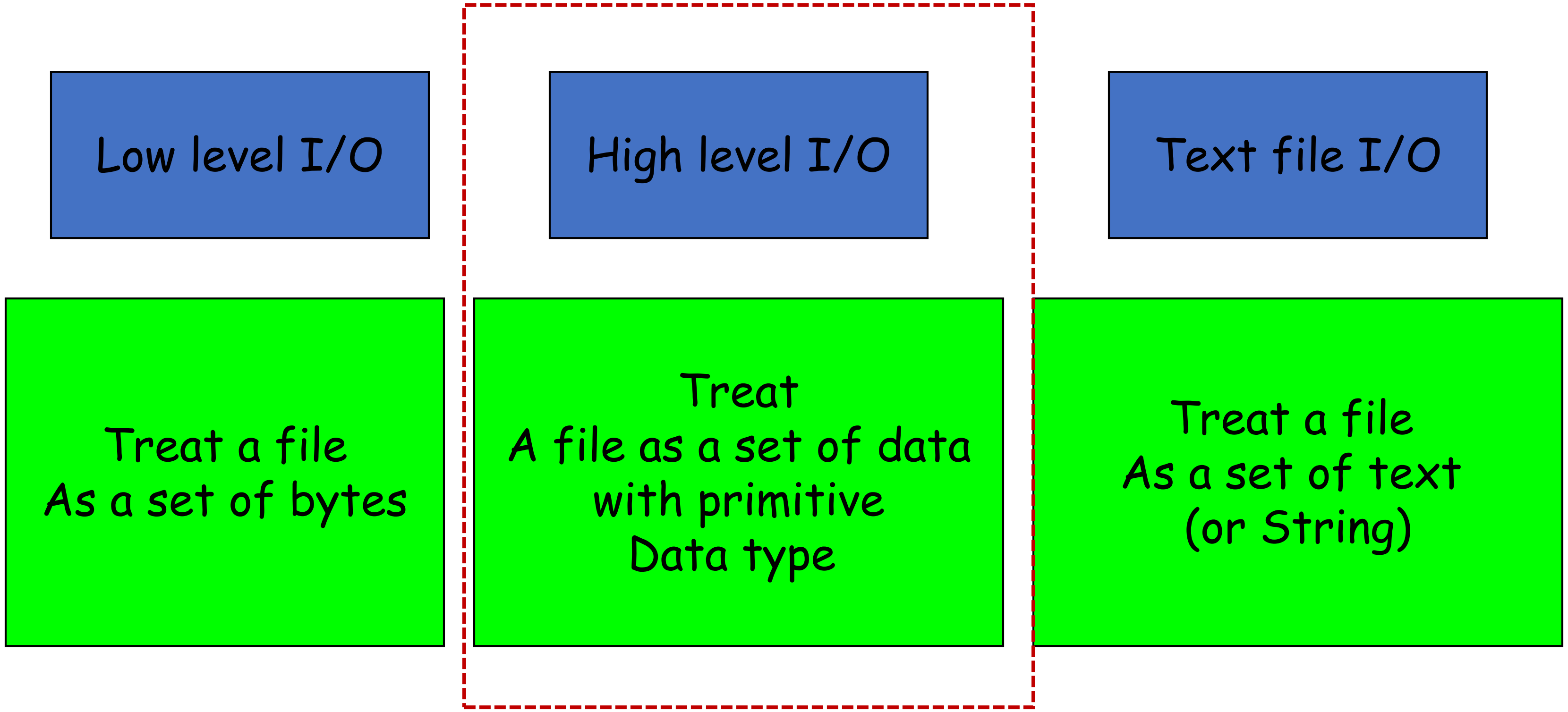
Sungchul Lee

Learning Object

- ➢High-Level File Input and Output
  - ❑Java.io package
  - ❑DataOutputStream Class
  - ❑DataInputStream Class

Type of File I/O

- Three type of file I/O



Low level I/O

Treat a file
As a set of bytes

High level I/O

Treat
A file as a set of data
with primitive
Data type

Text file I/O

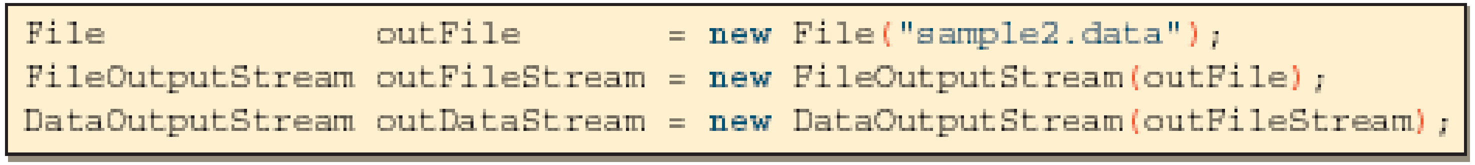
Treat a file
As a set of text
(or String)

High-Level File I/O

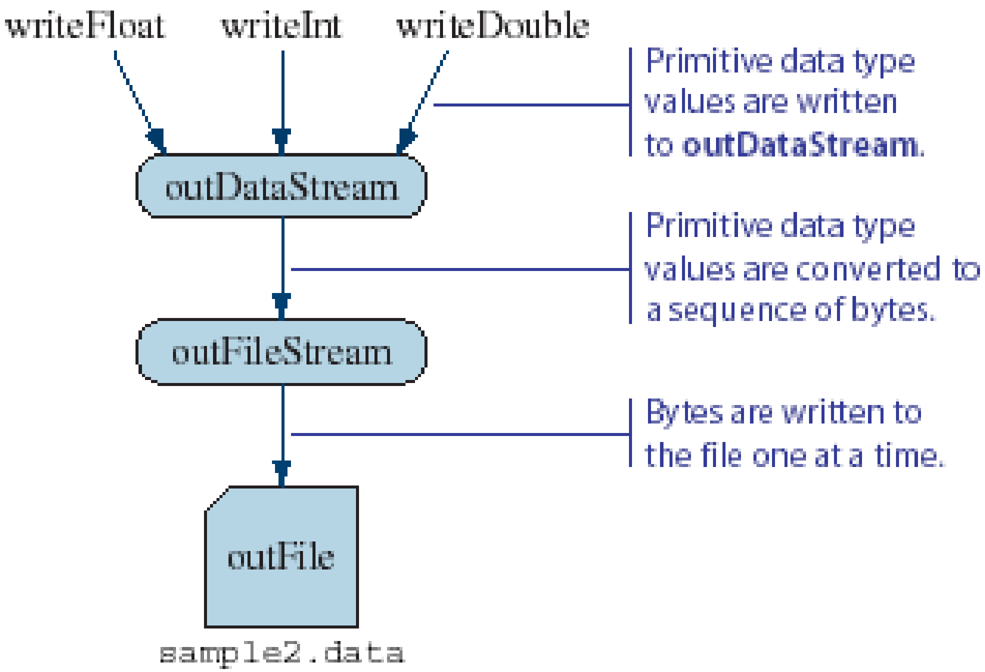
- java.io package
  - import java.io.*;
- Write Primitive data (int, double, char, etc)
  - **FileOutputStream** is used to handle raw binary data.
  - **DataOutputStream** lets an application write primitive data types
- Read Primitive data
  - **FileInputStream** is used to read bytes from file
  - **DataInputStream** allows an application to read primitive data
- Note: To read the data back correctly, we **must know the order of the data stored** and their **data types**

Setting up DataOutputStream

- A standard sequence to set up a DataOutputStream object:



```
File             outFile       = new File("sample2.data");
FileOutputStream outFileStream = new FileOutputStream(outFile);
DataOutputStream outDataStream = new DataOutputStream(outFileStream);
```



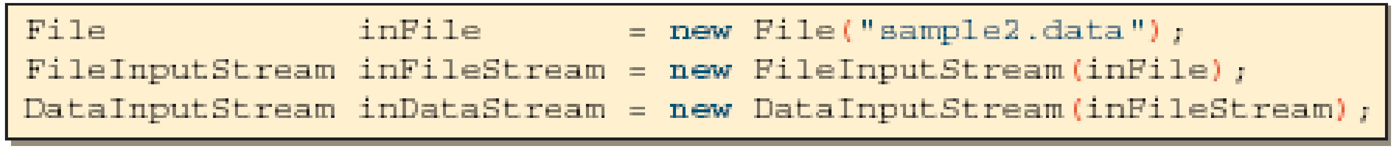
Primitive data type values are written to **outDataStream**.

Primitive data type values are converted to a sequence of bytes.

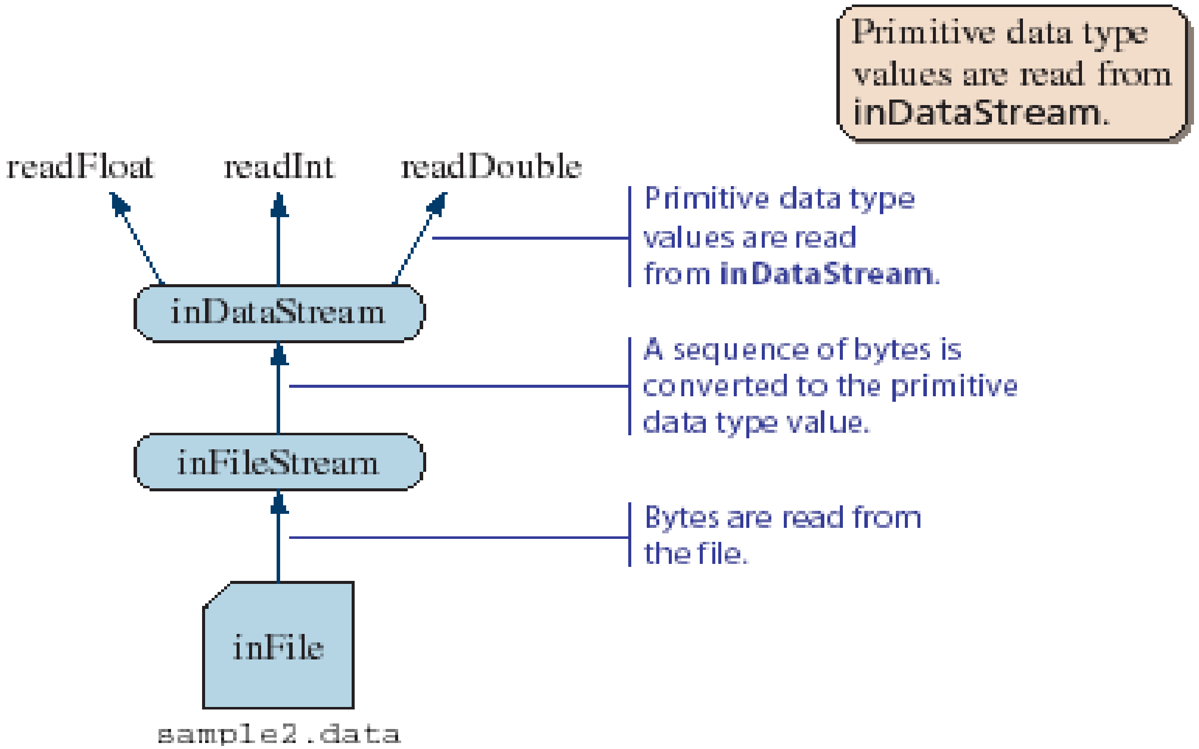
Bytes are written to the file one at a time.

Setting up DataInputStream

- A standard sequence to set up a DataInputStream object:



```
File            inFile       = new File("sample2.data");
FileInputStream inFileStream = new FileInputStream(inFile);
DataInputStream inDataStream = new DataInputStream(inFileStream);
```



Primitive data type values are read from inDataStream.

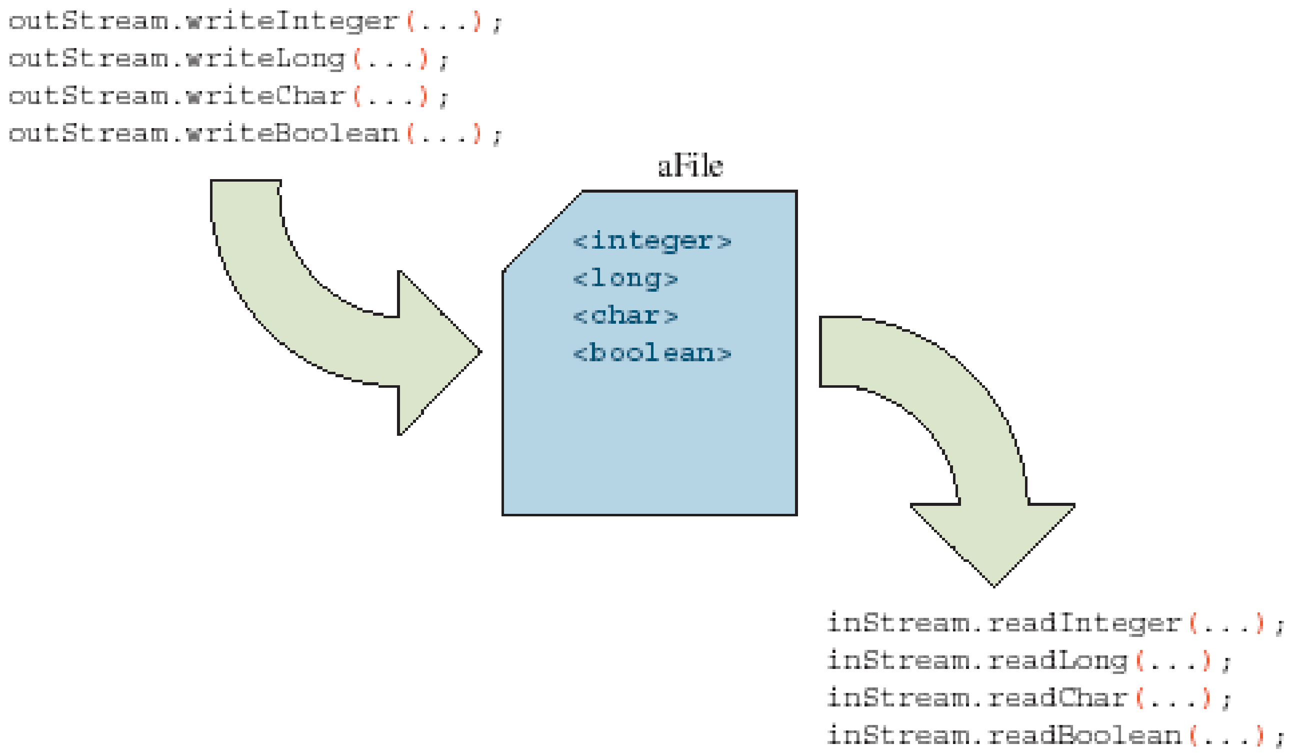
Primitive data type values are read from **inDataStream**.

A sequence of bytes is converted to the primitive data type value.

Bytes are read from the file.

Reading Data Back in Right Order

- The order of write and read operations must match in order to read the stored primitive data back correctly



```
outStream.writeInteger(...);
outStream.writeLong(...);
outStream.writeChar(...);
outStream.writeBoolean(...);
```

aFile

```
<integer>
<long>
<char>
<boolean>
```

```
inStream.readInteger(...);
inStream.readLong(...);
inStream.readChar(...);
inStream.readBoolean(...);
```

Practice

1. Make a new project (Reference: Create Project and Class File)
   - ❑Project name: High_Level_IO
2. Create a new Class File
   - ❑Class name: Main
3. Coding:
   - ❑import java.io.*;
     - ❖File
     - ❖FileOutputStream and DataOutputStream
     - ❖FileInputStream and DataInputStream

Practice – code (Write)

```
import java.io.*;
public class Main {
    public static void main (String[] args) throws IOException {
        //set up outDataStream
        File outFile = new File("sample2.dat");
        FileOutputStream outFileStream = new FileOutputStream(outFile);
        DataOutputStream outDataStream = new
        DataOutputStream(outFileStream);
        //write values of primitive data types to the stream
        outDataStream.writeInt(123);
        outDataStream.writeLong(11L);
        outDataStream.writeFloat(22F);
        outDataStream.writeDouble(33D);
        outDataStream.writeChar('A');
        outDataStream.writeBoolean(true);
```

```
//output done, so close the stream
outDataStream.flush();
outDataStream.close();
outFileStream.flush();
outFileStream.close();
```

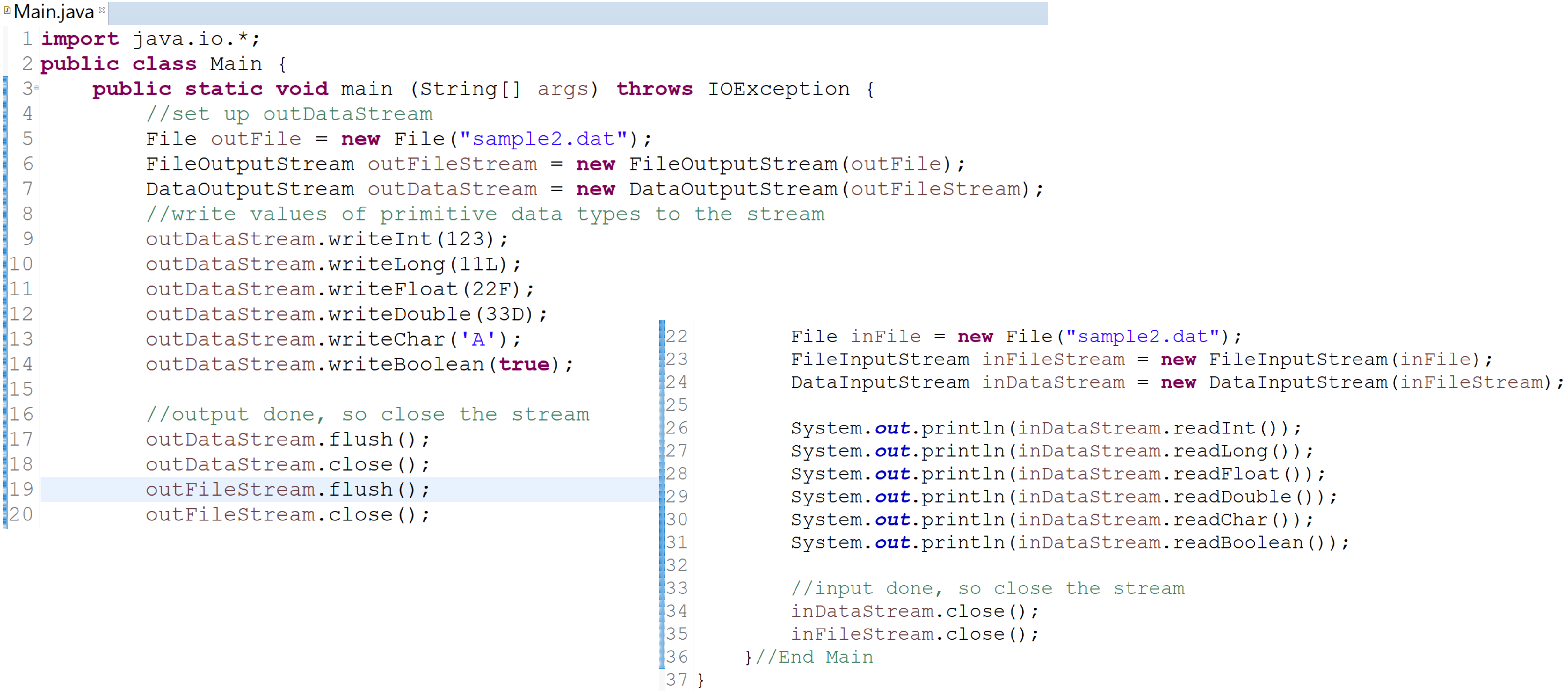
Practice – code (Read)

```
        File inFile = new File("sample2.dat");
        FileInputStream inFileStream = new FileInputStream(inFile);
        DataInputStream inDataStream = new DataInputStream(inFileStream);

        System.out.println(inDataStream.readInt());
        System.out.println(inDataStream.readLong());
        System.out.println(inDataStream.readFloat());
        System.out.println(inDataStream.readDouble());
        System.out.println(inDataStream.readChar());
        System.out.println(inDataStream.readBoolean());

          //input done, so close the stream
         inDataStream.close();
         inFileStream.close();
    }//End Main
}
```

Practice – Code



```java
import java.io.*;
public class Main {
    public static void main (String[] args) throws IOException {
        //set up outDataStream
        File outFile = new File("sample2.dat");
        FileOutputStream outFileStream = new FileOutputStream(outFile);
        DataOutputStream outDataStream = new DataOutputStream(outFileStream);
        //write values of primitive data types to the stream
        outDataStream.writeInt(123);
        outDataStream.writeLong(11L);
        outDataStream.writeFloat(22F);
        outDataStream.writeDouble(33D);
        outDataStream.writeChar('A');
        outDataStream.writeBoolean(true);

        //output done, so close the stream
        outDataStream.flush();
        outDataStream.close();
        outFileStream.flush();
        outFileStream.close();
```

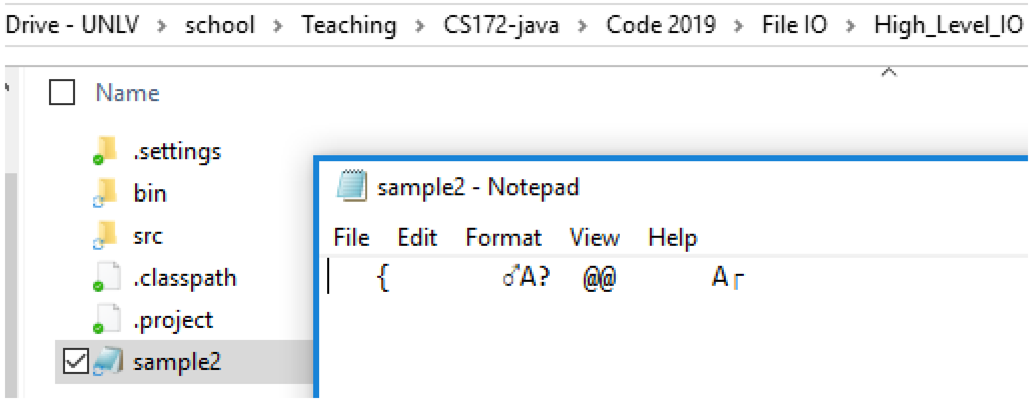
```java
        File inFile = new File("sample2.dat");
        FileInputStream inFileStream = new FileInputStream(inFile);
        DataInputStream inDataStream = new DataInputStream(inFileStream);

        System.out.println(inDataStream.readInt());
        System.out.println(inDataStream.readLong());
        System.out.println(inDataStream.readFloat());
        System.out.println(inDataStream.readDouble());
        System.out.println(inDataStream.readChar());
        System.out.println(inDataStream.readBoolean());

        //input done, so close the stream
        inDataStream.close();
        inFileStream.close();
    }//End Main
}
```

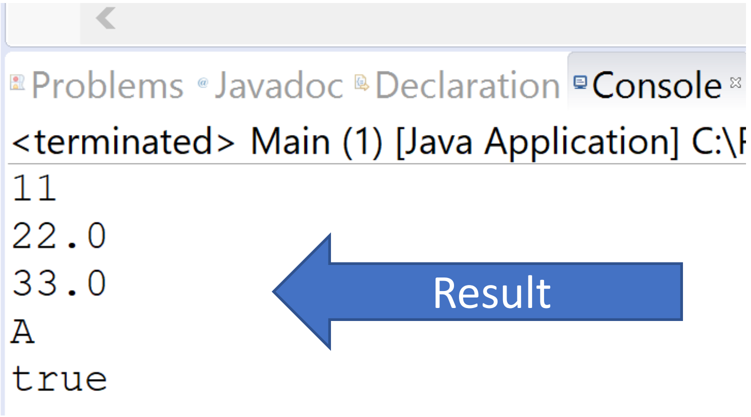
Practice –Result

- ➢Sample2.txt
  - ❑In Project folder
  - ❑Byte type



Drive - UNLV › school › Teaching › CS172-java › Code 2019 › File IO › High_Level_IO

```
    {           ♂A?  @@        A┌
```

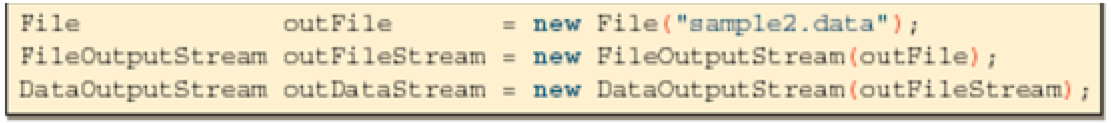


```
<terminated> Main (1) [Java Application] C:\F
11
22.0
33.0
A
true
```

Result

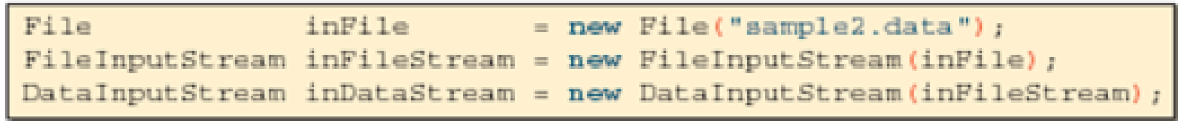
Summary

- ➢High-Level File Input and Output
  - ❑Java.io package
  - ❑Write



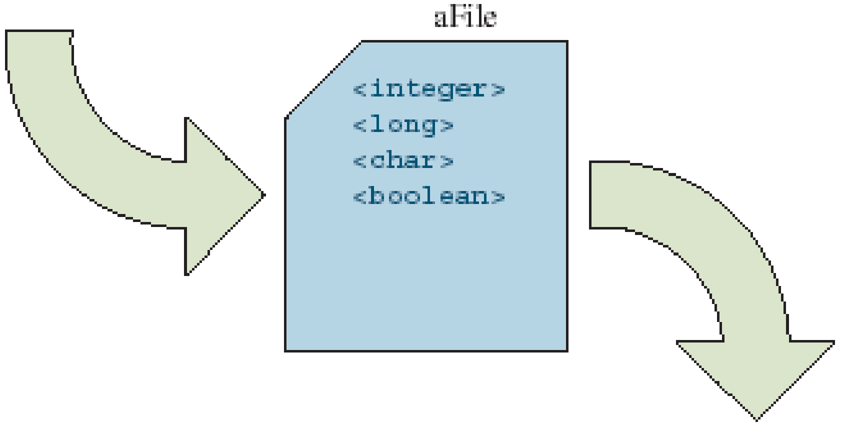
```
File             outFile       = new File("sample2.data");
FileOutputStream outFileStream = new FileOutputStream(outFile);
DataOutputStream outDataStream = new DataOutputStream(outFileStream);
```

  - ❑Read



```
File            inFile       = new File("sample2.data");
FileInputStream inFileStream = new FileInputStream(inFile);
DataInputStream inDataStream = new DataInputStream(inFileStream);
```

```
outStream.writeInteger(...);
outStream.writeLong(...);
outStream.writeChar(...);
outStream.writeBoolean(...);
```



```
inStream.readInteger(...);
inStream.readLong(...);
inStream.readChar(...);
inStream.readBoolean(...);
```