```java
// Typical I/O stream configurations.
import java.io.*;

public class IOStreamDemo {
    // Throw exceptions to console:
    public static void main(String[] args)
        throws IOException {
        // 1. Reading input by lines:
        BufferedReader in =
            new BufferedReader(
                                   new FileReader("IOStreamDemo.java"));
        String s, s2 = new String();
        while((s = in.readLine())!= null)
            s2 += s + "\n";
        in.close();

        // 1b. Reading standard input:
        BufferedReader stdin =
            new BufferedReader(
                                   new InputStreamReader(System.in));
        System.out.print("Enter a line:");
        System.out.println(stdin.readLine());

        // 2. Input from memory
         StringReader in2 = new StringReader(s2);
         int c;
         while((c = in2.read()) != -1)
             System.out.print((char)c);

         // 3. Formatted memory input
         try {
             DataInputStream in3 =
                 new DataInputStream(
                                         new ByteArrayInputStream(s2.getBytes()));
             while(true)
                 System.out.print((char)in3.readByte());
         } catch(EOFException e) {
             System.err.println("End of stream");
         }

         // 4. File output
         try {
             BufferedReader in4 =
                 new BufferedReader(
                                        new StringReader(s2));
             PrintWriter out1 =
                 new PrintWriter(
                                     new BufferedWriter(
                                                        new FileWriter("IODemo.out"))
             int lineCount = 1;
             while((s = in4.readLine()) != null )
                 out1.println(lineCount++ + ":" + s);
             out1.close();
         } catch(EOFException e) {
             System.err.println("End of stream");
         }

         // 5. Storing & recovering data
         try {
             DataOutputStream out2 =
                 new DataOutputStream(
                                          new BufferedOutputStream(
                                                                   new FileOutputStr
             out2.writeDouble(3.14159);
             out2.writeCharswriteUTF("That was pi\n");
             out2.writeBytes("That was pi\n");
             out2.writeDouble(1.41413);
             out2.writeUTF("Square root of 2");
             out2.close();
```



```java
             DataInputStream in5 =
                 new DataInputStream(
                                         new BufferedInputStream(
                                                                   new FileInputStream
             BufferedReader in5br =
                 new BufferedReader(
                                        new InputStreamReader(in5));
             // Must use DataInputStream for data:
             System.out.println(in5.readDouble());
             // Only readUTF() will recover the
             // Java-UTF String properly:
             // Can now use the "proper" readLine():
             System.out.println(in5br.readLinereadUTF());
             // But the line comes out funny.
             // The one created with writeBytes is OK:
             System.out.println(in5br.readLine());
             // Read the following double and String:
             System.out.println(in5.readDouble());
             System.out.println(in5.readUTF());
         } catch(EOFException e) {
             System.err.println("End of stream");
         }

         // 6. Reading/writing random access files
         RandomAccessFile rf =
             new RandomAccessFile("rtest.dat", "rw");
         for(int i = 0; i < 10; i++)
             rf.writeDouble(i*1.414);
         rf.close();

         rf =
             new RandomAccessFile("rtest.dat", "rw");
         rf.seek(5*8);
         rf.writeDouble(47.0001);
         rf.close();

         rf =
             new RandomAccessFile("rtest.dat", "r");
         for(int i = 0; i < 10; i++)
             System.out.println(
                                   "Value " + i + ": " +
                                   rf.readDouble());
         rf.close();
    }
}
```