```
class FileInputStreamDemo {
public static void main(String args[]) throws IOException {

int size;

InputStream f = new FileInputStream("FileInputStreamDemo.java");

System.out.println("Total Available Bytes: " + (size = f.available()));

int n = size/40;

System.out.println("First " + n + " bytes of the file one read() at a
time");

for (int i=0; i < n; i++) {
    System.out.print((char) f.read());
}

System.out.println("\nStill Available: " + f.available());

System.out.println("Reading the next " + n + " with one read(b[])");

byte b[] = new byte[n];
```

```
if (f.read(b) != n) {
    System.err.println("couldn't read " + n + " bytes.");
}

System.out.println(new String(b, 0, n));
System.out.println("\nStill Available: " + (size = f.available()));
System.out.println("Skipping half of remaining bytes with skip()");
f.skip(size/2);

System.out.println("Still Available: " + f.available());
System.out.println("Reading " + n/2 + " into the end of array");
if (f.read(b, n/2, n/2) != n/2) {
    System.err.println("couldn't read " + n/2 + " bytes.");
}

System.out.println(new String(b, 0, b.length));
System.out.println("\nStill Available: " + f.available());
f.close();
}
}
```

```
Total Available Bytes: 1433
First 35 bytes of the file one read() at a time
// Demonstrate FileInputStream.
im
Still Available: 1398
Reading the next 35 with one read(b[])
port java.io.*;
class FileInputS
Still Available: 1363
Skipping half of remaining bytes with skip()
Still Available: 682
Reading 17 into the end of array
port java.io.*;
read(b) != n) {
S
Still Available: 665
```