Chapter 13

File Processing

Reading from a text file

```
File tf = new File("t1.txt");
Scanner inFile = new Scanner(tf);
```

or

```
Scanner inFile =
   new Scanner(new File("t1.txt"));
```

Read from file whose name is command line argument

```
// Create Scanner object that reads from the keyboard.
Scanner kb = new Scanner(System.in);

// Prompt user for file name.
System.out.println("Enter file name");

// Read file name from keyboard
String fileName = kb.next();

// Create Scanner object for file
Scanner inFile = new Scannner(new File(fileName));


while (inFile.hasNextInt())    // any numbers left?
{
   x = inFile.nextInt();       // read number
   System.out.println(x);      // display number
 }

inFile.close();
```

Writing a text file

```
PrintWriter outFile = new
PrintWriter("numbers.txt");

for (int i = 1; i <= 100; i++)
   outFile.println(i);

outFile.println(i);

outFile.close();
```

Handling IOException

```
public void f() throws IOException
{
    Scanner inFile =
       new Scanner(new File("t1.txt"));
    ...
}
```

Alternative

```
public void f()
{
   Scanner inFile = null; // need init value
   try
   {
       inFile = new Scanner(new File("t1.txt"));
   }
   catch (IOException e)
   {
      ...
   }
   ...
}
```

Compile-time error

```
 1 public void f()
 2 {
 3    int x;
 4    Scanner inFile; // inFile not initialized
 5
 6    try
 7    {
 8        inFile = new Scanner(new File("t1.txt"));
 9    }
10    catch (IOException e)
11    {
12       System.out.println("File may not exist");
13    }
14    // following code executed even if line 8 fails
15    x = inFile.nextInt();// inFile may be undefined
16 }
```

```
 1 import java.util.Scanner;
 2 import java.io.*;   // for IOException, PrintWriter
 3 class IOExample1
 4 {
 5    public static void main(String[] args)
 6    {
 7       Scanner inFile = null;
 8       PrintWriter outFile = null;
 9
10       try
11       {
12          inFile = new Scanner(new File("t1.txt"));
13          outFile = new PrintWriter("t2.txt");
14       }
15       catch (IOException e)
16       {
17          System.out.println(e.getMessage()); // display error
18          System.exit(1);                     // terminate
19       }
20
21       String s;
22       while (inFile.hasNextLine())
23       {
24          s = inFile.nextLine();    // read one line
25          outFile.println(s);       // write this line
26       }
27
28       inFile.close();              // close files
29       outFile.close();
30    }
31 }
```

Use `throws`

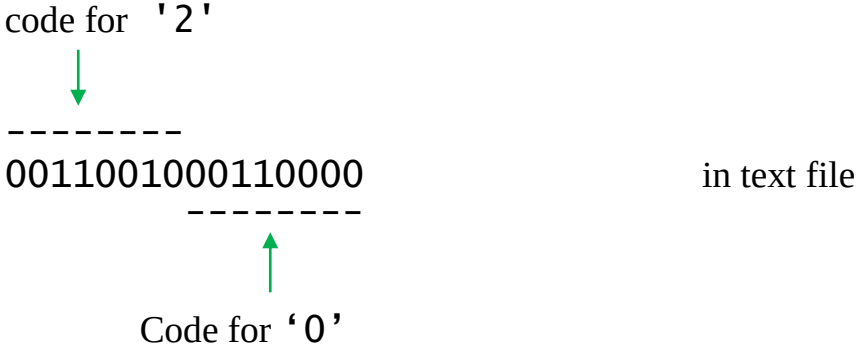
```
 1 import java.util.Scanner;
 2 import java.io.*;              // java.io has IOException
 3 class IOExample2
 4 {
 5    public static void main(String[] args) throws IOException
 6    {
 7       Scanner inFile = new Scanner(new File("t1.txt"));
 8       PrintWriter outFile = new PrintWriter("t2.txt");
 9
10       String s;
11       while (inFile.hasNextLine())
12       {
13          s = inFile.nextLine();     // read one line
14          outFile.println(s);        // write this line
15       }
16
17       inFile.close();               // close files
18       outFile.close();
19    }
20 }
```

Binary Files

```
x = 20;
```

Binary number equal to decimal 20 is

`00000000000000000000000000010100` in binary file



in text file

Writing to an output file

```
 1 import java.io.*;
 2 import java.util.Scanner;
 3 class TextandBinOutExample
 4 {
 5      public static void main(String[] args) throws IOException
 6      {
 7         PrintWriter textOut = new PrintWriter("t3.txt");
 8         int x = 20;
 9         textOut.println(x);       // output text
10         textOut.close();
11
12         DataOutputStream binOut = new DataOutputStream(
13                                   new FileOutputStream("b1.bin"));
14         binOut.writeInt(x);       // output binary
15         binOut.close();
16      }
17 }
```

Reading a binary file

```
 1 import java.io.*;
 2 import java.util.Scanner;
 3 class BinInExample
 4 {
 5      public static void main(String[] args) throws IOException
 6      {
 7         int x;
 8
 9         DataInputStream binIn =
10            new DataInputStream(new FileInputStream(args[0]));
11
12         try
13         {
14            while (true)
15            {
16               x = binIn.readInt();
17               System.out.println(x);
18            }
19         }
20         catch (EOFException e)
21         {
22            binIn.close();
23         }
24      }
25 }
```

DataInputStream

```
byte readByte(byte b)
short readShort(short s)
int readInt(int i)
long readLong(long l)
float readFloat(float f)
double readDouble(double d)
char readChar(int c)
boolean readBoolean(boolean b)
String readUTF(String s)
void close()
```

DataOutputStream

```
void writeByte(byte b)
void writeShort(short s)
void writeInt(int i)
void writeLong(long l)
void writeFloat(float f)
void writeDouble(double d)
void writeChar(int c)
void writeBoolean(boolean b)
void writeUTF(String s)
void close()
```

Exception Hierarchy

Fig. 13.8 shows the hierarchy among the exception classes you are likely to encounter.