

Appendix: Java 2 Input Examples

Example Program ReadChars.java

```
import java.io.*;

class ReadChars {

    public static void main(String args[ ]) {
        char c=' ';
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter characters ( period<Enter> stops:");
        do {
            try {
                c = (char) br.read( );
                System.out.println(c);
            } catch (IOException e) {
                System.out.println("IO Exception on Buffered Read");
            }
        } while (c != '.');
    }
}
```

Objects of class *BufferedReader* can produce an *IOException*. This must be thrown to an external exception handler or caught within the class. To handle any *IOExceptions* in the example, the flow of control statement *try and catch* is used. The *try* section contains a call to the *read* method the *BufferedReader* object *br*. The *catch* section contains the action to handle an *IOException*. Since the use of *try and catch* means that the variable *c* may not receive a value, it must be initialized before entering the *do-while* loop.

This example is compiled:

```
javac ReadChars.java
```

which produces the file *ReadChars.class*. It can then be run with the Java 2 program interpreter as follows (note that ".class" is not included in the command line):

```
java ReadChars
```

Example Program ReadLines.java

```
import java.io.*;

class ReadLines {

    public static void main(String args[ ]) {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        String str = " ";
        System.out.println("Enter information (stop quits):");
        do {
            try {
                str = br.readLine();
                System.out.println(str);
            } catch (IOException e) {
                System.out.println("IO Exception on Buffered Read");
            };
        } while (!str.equals("stop"));
    }
}
```

Objects of class *BufferedReader* can produce an *IOException*. This must be thrown to an external exception handler or caught within the class. To handle any *IOExceptions* in the example, the flow of control statement *try and catch* is used. The *try* section contains a call to the *read* method the *BufferedReader* object *br*. The *catch* section contains the action to handle an *IOException*. Since the use of *try and catch* means that the variable *str* may not receive a value, it must be initialized before entering the *do-while* loop.

This example is compiled:

```
javac ReadLines.java
```

which produces the file *ReadLines.class*. It can then be run with the Java 2 program interpreter as follows (note that ".class" is not included in the command line):

```
java ReadLines
```

Example Program ReadInt.java

```

import java.io.*;

class ReadInt {

    public static void main(String args[ ]) {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int i = 0;
        int t = 0;
        System.out.println("Enter information (<Enter> quits:");
        do {
            try {
                t = br.read( );

                // the character 0 has a value of 48
                // the character 9 has a value of 47
                // by subtracting 48 from t before adding it to i, the correct value is added
                // by multiplying i times 10, the decimal point of i is moved one place to the right
                // i*10+t-48 is used to accumulate the integer value typed in
                if (t>47 && t<58) i = i*10 + t-48;

            } catch (IOException e) {
                System.out.println("IO Exception on Buffered Read");
            }
        } while (t!=10 && t!=13);
        System.out.println(i);
    }
}

```

Objects of class *BufferedReader* can produce an *IOException*. This must be thrown to an external exception handler or caught within the class. To handle any *IOExceptions* in the example, the flow of control statement *try and catch* is used. The *try* section contains a call to the *read* method the *BufferedReader* object *br*. The *catch* section contains the action to handle an *IOException*. Since the use of *try and catch* means that the variable *t* may not receive a value, it must be initialized before entering the *do-while* loop.

This example is compiled:

```
javac ReadInt.java
```

which produces the file *ReadInt.class*. It can then be run with the Java 2 program interpreter as follows (note that ".class" is not included in the command line):

```
java ReadInt
```