Appendix 5
ConsoleIn

```
1  import java.io.BufferedReader;
2  import java.io.InputStreamReader;
3  import java.io.IOException;

4  /**
5   Class for robust keyboard input.
6   If the user enters an improper input, that is, an input of the wrong type
7   or a blank line when the line should be nonblank, then the user is given
8   an error message and is prompted to reenter the input
9
10  @author Walter Savitch September 18, 2003
11 */
12 public class ConsoleIn
13 {

14     private static final BufferedReader inputObject =
15             new BufferedReader(new InputStreamReader(System.in));


16     /**
17      Reads a line of text and returns that line as a String value.
18      This will read the rest of a line if the line is already partially read.

19      @return The line input from the keyboard.
20     */
21     public static String readLine()
22     {
23         String inputLine = null;

24         try
25         {
26             inputLine = inputObject.readLine();
27         }
28         catch(IOException e)
29         {
30             System.out.println("Fatal Error.Aborting.");
31             System.exit(0);
32         }
```

```
33          return inputLine;
34      }


35      /**
36       The user is supposed to enter a whole number of type int on a line by
37       itself. There may be whitespace before and/or after the number.
38       Returns the number entered as a value of type int. The rest of the line
39       is discarded. If the input is not entered correctly, then in most cases,
40       the user will be asked to reenter the input. In particular, incorrect
41       number formats and blank lines result in a prompt to reenter the input.

42       @return Returns the integer input.
43      */
44      public static int readLineInt()
45      {
46          String inputString = null;
47          int number = 0;//To keep the compiler happy.
48          boolean done = false;

49          while (! done)
50          {
51              try
52              {
53                  inputString = readLine();
54                  number = Integer.parseInt(inputString.trim());
55                  done = true;
56              }
57              catch (NumberFormatException e)
58              {
59                  System.out.println(
60                           "Input number is not in correct format.");
61                  System.out.println("The input number must be");
62                  System.out.println("a whole number written as an");
63                  System.out.println("ordinary numeral, such as 42.");
64                  System.out.println("Do not include a plus sign.");
65                  System.out.println("Minus signs are OK,");

66                  System.out.println("Try again.");
67                  System.out.println("Enter a whole number:");
68              }
69          }

70          return number;
71      }
```

```
72      /**
73       The user is supposed to enter a whole number of type long
74       on a line by itself. There may be whitespace before
75       and/or after the number.
76       Returns the number entered as a value of type long.
77       The rest of the line is discarded. If the input is not
78       entered correctly, then in most cases, the user will be asked
79       to reenter the input. In particular, incorrect number formats
80       and blank lines result in a prompt to reenter the input.

81       @return Returns the integer input.
82      */
83      public static long readLineLong()
84      {
85          String inputString = null;
86          long number = 0;//To keep the compiler happy.
87          boolean done = false;

88          while (! done)
89          {
90              try
91              {
92                  inputString = readLine();
93                  number = Long.parseLong(inputString.trim());
94                  done = true;
95              }
96              catch (NumberFormatException e)
97              {
98                  System.out.println(
99                           "Input number is not in correct format.");
100                 System.out.println("The input number must be");
101                 System.out.println("a whole number written as an");
102                 System.out.println("ordinary numeral, such as 42");
103                 System.out.println("Do not include a plus sign.");
104                 System.out.println("Minus signs are OK,");

105                 System.out.println("Try again.");
106                 System.out.println("Enter a whole number:");
107             }
108         }

109          return number;
110      }
```

```
111     /**
112      The user is supposed to enter a whole number of type byte
113      on a line by itself. There may be whitespace before
114      and/or after the number.
115      Returns the number entered as a value of type byte.
116      The rest of the line is discarded. If the input is not
117      entered correctly, then in most cases, the user will be asked
118      to reenter the input. In particular, incorrect number formats
119      and blank lines result in a prompt to reenter the input.

120      @return Returns the integer input.
121     */
122     public static byte readLineByte()
123     {
124         String inputString = null;
125         byte number = 0;//To keep the compiler happy.
126         boolean done = false;

127         while (! done)
128         {
129             try
130             {
131                 inputString = readLine();
132                 number = Byte.parseByte(inputString.trim());
133                 done = true;
134             }
135             catch (NumberFormatException e)
136             {
137                 System.out.println(
138                         "Input number is not in correct format.");
139                 System.out.println("The input number must be");
140                 System.out.println("a whole number written as an");
141                 System.out.println("ordinary numeral, such as 42");
142                 System.out.println("Do not include a plus sign.");
143                 System.out.println("Minus signs are OK,");

144                 System.out.println("Try again.");
145                 System.out.println("Enter a whole number:");
146             }
147         }

148         return number;
149     }
```

```
150     /**
151      The user is supposed to enter a whole number of type short
152      on a line by itself. There may be whitespace before
153      and/or after the number.
154      Returns the number entered as a value of type short.
155      The rest of the line is discarded. If the input is not
156      entered correctly, then in most cases, the user will be asked
157      to reenter the input. In particular, incorrect number formats
158      and blank lines result in a prompt to reenter the input.

159      @return Returns the integer input.
160     */
161     public static short readLineShort()
162     {
163         String inputString = null;
164         short number = 0;//To keep the compiler happy.
165         boolean done = false;
166 
167         while (! done)
168         {
169             try
170             {
171                 inputString = readLine();
172                 number = Short.parseShort(inputString.trim());
173                 done = true;
174             }
175             catch (NumberFormatException e)
176             {
177                 System.out.println(
178                         "Input number is not in correct format.");
179                 System.out.println("The input number must be");
180                 System.out.println("a whole number written as an");
181                 System.out.println("ordinary numeral, such as 42");
182                 System.out.println("Do not include a plus sign.");
183                 System.out.println("Minus signs are OK,");

184                 System.out.println("Try again.");
185                 System.out.println("Enter a whole number:");
186             }
187         }

188         return number;
189     }
```

```
190     /**
191      The user is supposed to enter a number of type double on a line by
itself.
192      There may be whitespace before and/or after the number.
193      Returns the number entered as a value of type double. The rest of the
line
194      is discarded. If the input is not entered correctly, then in most cases,
195      the user will be asked to reenter the input. In particular, incorrect
196      number formats and blank lines result in a prompt to reenter the input.

197      @return The floating point number input.
198     */
199     public static double readLineDouble()
200     {
201         String inputString = null;
202         double number = 0;//To keep the compiler happy.
203         boolean done = false;

204         while (! done)
205         {
206             try
207             {
208                 inputString = readLine();
209                 number = Double.parseDouble(inputString.trim());
210                 done = true;
211             }
212             catch (NumberFormatException e)
213             {
214                 System.out.println(
215                             "Input number is not in correct format.");
216                 System.out.println("The input number must be");
217                 System.out.println("an ordinary number either with");
218                 System.out.println("or without a decimal point,");
219                 System.out.println("such as 42 or 41.999");
220                 System.out.println("Try again.");
221                 System.out.println("Enter a number:");
222             }
223         }

224         return number;
225     }
```

```
226     /**
227      The user is supposed to enter a number of type float
228      on a line by itself. There may be whitespace before
229      and/or after the number.
230      Returns the number entered as a value of type float.
231      The rest of the line is discarded. If the input is not
232      entered correctly, then in most cases, the user will be asked
233      to reenter the input. In particular, incorrect number formats
234      and blank lines result in a prompt to reenter the input.

235      @return The floating point number input.
236     */
237     public static float readLineFloat()
238     {
239         String inputString = null;
240         float number = 0;//To keep the compiler happy.
241         boolean done = false;

242         while (! done)
243         {
244             try
245             {
246                 inputString = readLine();
247                 number = Float.parseFloat(inputString.trim());
248                 done = true;
249             }
250             catch (NumberFormatException e)
251             {
252                 System.out.println(
253                         "Input number is not in correct format.");
254                 System.out.println("The input number must be");
255                 System.out.println("an ordinary number either with");
256                 System.out.println("or without a decimal point,");
257                 System.out.println("such as 42 or 42.999");
258                 System.out.println("Try again.");
259                 System.out.println("Enter a number:");
260             }
261         }

262         return number;
263     }
```

```
264     /**
265      Returns the first non-whitespace character on the input line
266      The rest of the input line is discarded. If the line contains
267      only whitespace, the user is asked to reenter the line.

268      @return The first non-whitespace character on the input line.
269     */
270     public static char readLineNonwhiteChar()
271     {
272         boolean done = false;
273         String inputString = null;
274         char nonWhiteChar = ' ';//To keep the compiler happy.

275         while (! done)
276         {
277             inputString = readLine();
278             inputString = inputString.trim();
279             if (inputString.length() == 0)
280             {
281                 System.out.println("Input is not correct.");
282                 System.out.println("The input line must contain at");
283                 System.out.println("least one non-whitespace character.");
284                 System.out.println("Try again.");
285                 System.out.println("Enter input:");
286             }
287             else
288             {
289                 nonWhiteChar = inputString.charAt(0);
290                 done = true;
291             }
292         }

293         return nonWhiteChar;
294     }


295     /**
296      Input should consist of a single word on a line, possibly surrounded by
297      whitespace. The line is read and discarded. If the input word is "true" or
298     "t", then true is returned. If the input word is "false" or "f", then false
299      is returned. Uppercase and lowercase letters are considered equal. If the
300      user enters anything else, the user is asked to reenter the input.
```

```
301         @return The boolean value entered.
302       */
303       public static boolean readLineBoolean()
304       {
305           boolean done = false;
306           String inputString = null;
307           boolean valueReturned = false;//To keep the compiler happy.

308           while (! done)
309           {
310               inputString = readLine();
311               inputString = inputString.trim();
312               if (inputString.equalsIgnoreCase("true")
313                       || inputString.equalsIgnoreCase("t"))
314               {
315                   valueReturned = true;
316                   done = true;
317               }
318               else if (inputString.equalsIgnoreCase("false")
319                           || inputString.equalsIgnoreCase("f"))
320               {
321                   valueReturned = false;
322                   done = true;
323               }
324               else
325               {
326                   System.out.println("Input is not correct.");
327                   System.out.println("The only valid inputs are:");
328                   System.out.println("the word true,");
329                   System.out.println("the word false,");
330                   System.out.println("the letter T,");
331                   System.out.println("the letter F.");
332                   System.out.println("Any combination of upper- and");
333                   System.out.println("lowercase letters is acceptable");
334                   System.out.println("Try again.");
335                   System.out.println("Enter input:");
336               }
337           }

338           return valueReturned;
339       }

340   }
```