EXCEPTIONS IN TEXT I/O

When `TextIO` reads a numeric value from the user, it makes sure that the user's response is legal, using a technique similar to the `while` loop and `try..catch` in the previous example. However, `TextIO` can read data from other sources besides the user. (See Subsection 2.4.5.) When it is reading from a file, there is no reasonable way for `TextIO` to recover from an illegal value in the input, so it responds by throwing an exception. To keep things simple, `TextIO` only throws exceptions of type *IllegalArgumentException*, no matter what type of error it encounters. For example, an exception will occur if an attempt is made to read from a file after all the data in the file has already been read. In `TextIO`, the exception is of type *IllegalArgumentException*. If you have a better response to file errors than to let the program crash, you can use a `try..catch` to catch exceptions of type *IllegalArgumentException*.

For example, suppose that a file contains nothing but real numbers, and we want a program that will read the numbers and find their sum and their average. Since it is unknown how many numbers are in the file, there is the question of when to stop reading. One approach is simply to try to keep reading indefinitely. When the end of the file is reached, an exception occurs. This exception is not really an error -- it's just a way of detecting the end of the data, so we can catch the exception and finish up the program. We can read the data in a `while (true)` loop and break out of the loop when an exception occurs. This is an example of the somewhat unusual technique of using an exception as part of the expected flow of control in a program.

To read from the file, we need to know the file's name. To make the program more general, we can let the user enter the file name, instead of hard-coding a fixed file name in the program. However, it is possible that the user will enter the name of a file that does not exist. When we use `TextIO.readfile` to open a file that does not exist, an exception of type *IllegalArgumentException* occurs. We can catch this exception and ask the user to enter a different file name. Here is a complete program that uses all these ideas:

```
/**
 * This program reads numbers from a file.  It computes the sum
and
 * the average of the numbers that it reads.  The file should
contain
 * nothing but numbers of type double; if this is not the case,
the
 * output will be the sum and average of however many numbers
were
 * successfully read from the file.  The name of the file will
be
 * input by the user.
 */

public class ReadNumbersFromFile {

   public static void main(String[] args) {

      while (true) {
         String fileName;  // The name of the file, to be input
by the user.
         TextIO.put("Enter the name of the file: ");
         fileName = TextIO.getln();
         try {
            TextIO.readFile( fileName );  // Try to open the
file for input.
            break;  // If that succeeds, break out of the loop.
         }
         catch ( IllegalArgumentException e ) {
            TextIO.putln("Can't read from the file \"" +
fileName + "\".");
            TextIO.putln("Please try again.\n");
         }
      }

      // At this point, TextIO is reading from the file.

      double number;  // A number read from the data file.
      double sum;      // The sum of all the numbers read so
far.
      int count;      // The number of numbers that were read.

      sum = 0;
      count = 0;

      try {
         while (true) { // Loop ends when an exception occurs.
             number = TextIO.getDouble();
             count++;  // This is skipped when the exception
occurs
             sum += number;
         }
      }
      catch ( IllegalArgumentException e ) {
         // We expect this to occur when the end-of-file is
encountered.
         // We don't consider this to be an error, so there is
nothing to do
         // in this catch clause.  Just proceed with the rest
of the program.
      }

      // At this point, we've read the entire file.

      TextIO.putln();
      TextIO.putln("Number of data values read: " + count);
      TextIO.putln("The sum of the data values: " + sum);
      if ( count == 0 )
         TextIO.putln("Can't compute an average of 0 values.");
      else
         TextIO.putln("The  average  of  the  values:    "  +
(sum/count));

   }

}
```

Source : http://math.hws.edu/javanotes/c3/s7.html