```
readFile {
     open the file;
      determine its size;
      allocate that much memory;
      read the file into memory;
      close the file;
}
```

```
errorCodeType readFile {
  initialize errorCode = 0;
  open the file;
  if (theFileIsOpen) {
    determine the length of the file;
    if (gotTheFileLength) {
      allocate that much memory;
      if (gotEnoughMemory) {
        read the file into memory;
        if (readFailed) {
          errorCode = -1;
        }
      } else {
        errorCode = -2;
      }
    } else {
      errorCode = -3;
    }
    close the file;
    if (theFileDidntClose && errorCode == 0) {
      errorCode = -4;
    } else {
      errorCode = errorCode and -4;
    }
  } else {
    errorCode = -5;
  }
  return errorCode;
}
```

```
readFile {
  try {
    open the file;
    determine its size;
    allocate that much memory;
    read the file into memory;
    close the file;
  } catch (fileOpenFailed) {
    doSomething;
  } catch (sizeDeterminationFailed) {
    doSomething;
  } catch (memoryAllocationFailed) {
    doSomething;
  } catch (readFailed) {
    doSomething;
  } catch (fileCloseFailed) {
    doSomething;
  }
}
```

```
method1 {
  call method2;
}

method2 {
  call method3;
}

method3 {
  call readFile;
}
```

```
method1 {
  errorCodeType error;
  error = call method2;
  if (error)
    doErrorProcessing;
  else
    proceed;
}

errorCodeType method2 {
  errorCodeType error;
  error = call method3;
  if (error)
    return error;
  else
    proceed;
}

errorCodeType method3 {
  errorCodeType error;
  error = call readFile;
  if (error)
    return error;
  else
    proceed;
}
```

```
method1 {
  try {
    call method2;
  } catch (exception) {
    doErrorProcessing;
  }
}

method2 throws exception {
  call method3;
}

method3 throws exception {
  call readFile;
}
```

```
// Note: This class won't compile by design!
import java.io.*;

public class InputFile {

  private FileReader in;

  public InputFile(String filename) {
    in = new FileReader(filename);
  }

  public String getWord() {
    int c;
    StringBuffer buf = new StringBuffer();

    do {
      c = in.read();
      if (Character.isWhitespace((char)c))
        return buf.toString();
      else {
        buf.append((char)c);
      } while (c != -1);

    return buf.toString();
  }
}
```

```
InputFile.java:15: Warning: Exception java.io.IOException must be caught,
or it must be declared in throws clause of this method.
                    while ((c = in.read()) != -1) {
                                        ^
```

```
public class InputFile {

  private FileReader in;

  public InputFile(String filename) throws FileNotFoundException {
    in = new FileReader(filename);
  }

  public String getWord()throws IOException {
    int c;
    StringBuffer buf = new StringBuffer();

    do {
      c = in.read();
      if (Character.isWhitespace((char)c))
        return buf.toString();
      else {
        buf.append((char)c);
      } while (c != -1);

    return buf.toString();
  }
}
```

```
// Note: This class won't compile by design!
// See ListOfNumbersDeclared.java or ListOfNumbers.java
// for a version of this  class that will compile.
import java.io.*;
import java.util.Vector;

public class ListOfNumbers {
  private Vector victor;
  private static final int size = 10;

  public ListOfNumbers () {
    victor = new Vector(size);
    for (int i = 0; i < size; i++)
      victor.addElement(new Integer(i));
  }

  public void writeList() {
    PrintWriter out = new PrintWriter(new FileWriter("OutFile.txt"));

    for (int i = 0; i < size; i++)
      out.println("Value at: " + i + " = " + victor.elementAt(i));

    out.close();
  }
}
```

```
...
public void writeList() {
  PrintWriter out = null;

  try {
    System.out.println("Entering try statement");
    out = new PrintWriter(new FileWriter("OutFile.txt"));

    for (int i = 0; i < size; i++)
      out.println("Value at: " + i + " = " + victor.elementAt(i));

  } catch (ArrayIndexOutOfBoundsException e) {
      System.err.println("Caught ArrayIndexOutOfBoundsException: " +
                                        e.getMessage());
  } catch (IOException e) {
      System.err.println("Caught IOException: " + e.getMessage());
  } finally {
      if (out != null) {
        System.out.println("Closing PrintWriter");
        out.close();
      } else {
        System.out.println("PrintWriter not open");
      }
  }
}
...
```

```
...
public void writeList()
            throws IOException, ArrayIndexOutOfBoundsException {
    PrintWriter out = new PrintWriter(new FileWriter("OutFile.txt"));

    for (int i = 0; i < size; i++)
      out.println("Value at: " + i + " = " + victor.elementAt(i));

    out.close();
  }
...
```