Lab4 part B: Exceptions

CSSSKL162: Programming Methodology Skills,

Summary

In this part of the lab, you will fix errors and modify the code to handle exceptions for three different cases below.

1. **Simple exception handling**
   a. Download the following file from the class website: **TestExceptions.java**
   b. Compile the code and notice the compilation error message.
   c. Modify the main() method to propagate the *IOException* (this is an ancestor of *FileNotFoundException*). This is done using a throws clause when the method is first declared.
   d. While there are multiple exceptions that might be thrown by this code, this change should propagate all of them, since they all inherit from *IOException*.
   e. Execute the program and notice the run-time error.
   f. Modify the code to handle the exceptions using a try-catch block **instead** of propagating the exception. When the exception occurs print: "The file you have requested cannot be found."
   g. Test your changes.

2. **Handling exceptions**
   a. Download the following file from the class website: **CallStack.java**
   b. Examine the code to determine what it does.
   c. Compile and execute the code.
   d. Modify the main() method to handle the exception that is propagated to it. Use a try-catch block to display a meaningful error message when the exception occurs.
   e. Test your code. Notice that, although the exception was thrown in func2, it is caught by the catch block in the main method.

3. **Handling exceptions with a finally clause**
   a. Download the following file from the class website: **TestFinally.java**

b. Examine the code to determine what it does.
c. Compile the code and notice the error.
d. Modify TestFinally.java to handle the exception following these instructions (and those embedded in the code):
    i. Embedded in the code is a note showing the location for the try statement.
    ii. The first line of the catch block should look like this:
        catch(IOException e) {
    iii. Display a meaningful error message to indicate the error that has occurred and include the following statement in your catch clause:
        System.out.println("The exception is: " + e);
    iv. Even when an error occurs in a method, there may be clean-up activities that are required before the method or program terminates. In this case, any open files should be closed prior to terminating the program.
    v. Include a *finally* block that closes the files using notes 4 and 5 in the code that show where to place the beginning and end of the block.
    vi. You will need to add a try-catch block inside the finally block to handle any IOExceptions associated with the file close operation.
e. Test your code. When it is working, you will generate another type of exception.
f. Add a second catch clause inside the finally block with a meaningful message to handle this new exception.

(Thanks are due to the faculty at University of Nebraska-Lincoln who created an exceptions lab from which this draws upon almost entirely.)