Welcome to Exceptions and Command-Line Lab

Learning Outcomes

By the end of this lab:

- Trace, explain and write code with exceptions including creating, throwing and catching.
- Trace, explain and write code for handling command line arguments.

Exercise A: Trace Exceptions 

1. Running TraceMe results in the Output. Trace the program and fill in the __blanks__ in the Output. Check your answer using an IDE.
   Output:

```
Exception in thread "main" _java.lang.RuntimeException_: _This is the
message._
      at TraceMe.methodC_(TraceMe.java:_6_)
      at TraceMe.main(TraceMe.java:_3_)
```

```
1 public class TraceMe{
2    public static void main(String[] args) {
3        methodC();
4    }
5    public static void methodC() {
6        throw new RuntimeException("This is the message.");
7    }
8}
```

Exception in thread "main" java.lang.RuntimeException: This is the message.
at TraceMe.methodC(TraceMe.java:6)
at TraceMe.main(TraceMe.java:3)

2. Answer the following questions, before running ExploreExceptions.java.
    a. **What will the output be for the following method calls in the main method:**
        i. **process(null);** A, B D F G
        ii. **process("abc");** A, B, C, F,
        iii. **process("");** A B E F 0 length key
    b. **Is the finally block always executed whether or not there is an exception?**
    Finally block always executed
    c. **Is the finally block executed when there is a return within the try block?**
    Finally block executed when there is a return within the block.
    d. **Is it possible to rethrow a caught exception? Where is an example of this?**

```
import java.io.IOException;
public class ExploreExceptions {

   static String lookupData(String key) throws Exception,IOException {
        if (key == null) {
            throw new IOException("null key");
        } else if (key.length() == 0) {
            throw new Exception("0 length key");
        }
        return key;
    }

    static void process(String key) throws Exception {
        System.out.print("A");
        try {
            System.out.print("B");
            String data = lookupData(key);
            System.out.print("C");
            return;
        } catch (IOException e) {
            System.out.print("D");
        } catch (Exception e) {
            System.out.print("E");
            throw e;
        } finally {
            System.out.print("F");
        }
        System.out.print("G");
    }

    public static void main(String[] args) throws Exception {
        process(null);
        System.out.println();
        process("abc");
        System.out.println();
        try {
```

```
            process("");
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
        System.out.println();
    }
}
```

3. The following program has a compilation error message on line 4 "Unhandled exception type Exception".

```
public class ExceptionHandling() throws Exception {
    public static int strLength(String str) {
        if (str == null) {
            throw new Exception();  //compilation error here
        } else {
            return str.length();
        }
    }
    public static void main(String[] args) {

        System.out.println(strLength(null));

        System.out.println("end of main");
    }
}
```

   a. **What tool/program verifies that checked exceptions are being handled by the programmer?**
      The Compiler
   b. **Does having this compiler error message indicate that Exception is a checked or unchecked exception?**
      It is checked exception because compiler error message is “Unhandled exception”
   c. **If in line 4, Exception was changed to RuntimeException, would the compiler error message go away? Why or why not?**
      The error message will be handled by the compiler if changed to Runtime.
   d. **How would you change the code such that the main method catches the exception, prints out a message, and continues normal execution printing out "end of main" at the end?**
      Adding a catch between the print statements should allow code to progress.
      Add throw

Exercise B: Debug CountWords 

Using a systematic process, fix the bugs in CountWords.java and note specific changes you make. The program is supposed to produce the following output when run from the command-line with the various arguments. The arguments follow the program's name on the command-line.

Recall 10 L | Command-Line Arguments describes how to pass arguments in Eclipse and IntelliJ.

Command:	`java CountWords`
Current and expected output:	`Usage: java CountWords [-letters] word(s)` `  -letters means count the number of letters and double letters.`

Command:	`java CountWords the words`

Current output:	`the words`
	`wordCount=2`
	`letterCount=8 doubleLetterCount=0`

Expected output:	`the words` `wordCount=2`

Command:	`java CountWords -letters wheelbarrow woodpecker windjammer`

Current output:	`-letters wheelbarrow woodpecker windjammer`
	`wordCount=4`
	`letterCount=38 doubleLetterCount=6`

Expected output:	`wheelbarrow woodpecker windjammer`
	`wordCount=3`
	`letterCount=31 doubleLetterCount=4`

a. **Were you able to resolve counting just the 3 words and not the -letters switch?**

    i. **Did you choose to change the loop from an enhanced for loop to regular for loop or come up with another solution?**

b. **Did you discover and resolve the bug where the letterCount is always output even when the -letters switch is not specified on the command-line?**
c. **Did you discover and resolve the bug where the w at the end of wheelbarrow and beginning of woodpecker are counted as double letters when they shouldn't be?**

Exercise TA: Demonstration and Discussion

This exercise is utilized to determine 3 points of the lab grade. We suggest reviewing these before you discuss these with your TA.

1. Exercise A: Tracing Exceptions: 1, 2 and 3.
    a. Demonstrate tracing and discuss answers.
2. Exercise B: Debug Count Words
    a. Discuss running a program and passing command line arguments and the process of systematically debugging.

Exercise C: Write a Program to Create the Exception 

Complete the WriteMe program so that it will generate exactly the error message below.

```
public class WriteMe {
      //?
    public static void main(String[] args) {
      //?
    }
}
```

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 0
      at WriteMeSol.methodA(WriteMeSol.java:9)
      at WriteMeSol.methodB(WriteMeSol.java:4)
      at WriteMeSol.main(WriteMeSol.java:13)
```

a. Exactly what does each line number in the exception message refer to?

Exercise D: Parsing Command-Line Arguments 

Within Translate write the code for parsing the command-line arguments.

```
L:\User>java Translate
Usage: java Translate [-u | -p] [-r] (-i | sentence)
  Options may be in any order, sentence last.
  Any non-option argument is the beginning of a sentence.
  -u means upper case
  -p means proper case
     if both -u and -p then only do first
  -r means reverse
  -i means interactive mode ('quit' to end)
  sentence means the words to translate (ignored if interactive mode)
```

```
L:\User>java Translate -p university of wisconsin
University Of Wisconsin

L:\User>java Translate -u university of wisconsin
UNIVERSITY OF WISCONSIN

L:\User>java Translate -r university of wisconsin
nisnocsiw fo ytisrevinu

L:\User>java Translate -p -r university of wisconsin
nisnocsiW fO ytisrevinU

L:\User>java Translate -p "university of wisconsin"
University Of Wisconsin

L:\User>java Translate -p -r -i
The quick brown fox jumped over the lazy dog.
.goD yzaL ehT revO depmuJ xoF nworB kciuQ ehT
The lazy dog didn't even move.
.evoM nevE t'ndiD goD yzaL ehT
quit
```

Exercise E: Trace and Explain Encode 

Read the Encode program to determine what it is doing. Add comments to the code to guide a reader that knows Java but isn't familiar with encoding methods such as ROT13.

What is the output for each of the following?

```
java Encode
java Encode -f the message
java Encode -d hello there
java Encode -E 3
java Encode -e # the message
java Encode -e 2 the message
java Encode -d 2 vjg oguucig
```

Did you improve the Usage message to describe what the program is supposed to do?

Additional Learning Materials

When you have mastered everything in this lab, *and previous labs,* then you are welcome to learn from additional learning resources available on the web and beyond this course:

- https://codingbat.com/java
- https://www.khanacademy.org/computing/computer-programming
- https://techdevguide.withgoogle.com/
- https://code.org/
- http://programmingbydoing.com/

Note: Team Lab is focused on pair programming and open discussion and so it is not appropriate to work on individual programming assignments.

Contributions by Marc Renault