Assignment A3 (45 marks)

Focus: *Exception Handling, Text I/O*

Q1. [15 marks] Write a program that creates an integer array with 50 random values, prompts the user to enter the index of an element in the array between 0 and 49, then displays the corresponding element value. If the specified index is out of bounds, display an error message (e.g. "Out of Bounds") and ask the user to enter another index. Use a while loop that will keep prompting the user until a valid input is received. To handle invalid inputs, write **two versions of your program**: one that uses **exception handling**, and one that uses **defensive programming**. Assume a user will always enter numbers.

Sample run

```
Enter an index: 77
Out of Bound. Try again:98
Out of Bound. Try again:43
The element is 4
```

Q2. [10 marks] "*An* `InputMismatchException` *is thrown by a Scanner to indicate that the token retrieved does not match the pattern for the expected type, or that the token is out of range for the expected type*". For example, if you use the `nextDouble` to read a token that is not of the type `double`.

Write a program that prompts the user to enter a mathematical formula (e.g., 4.3 + 5.1) and then displays the result. Your program should prompt the user to try again if the input has an invalid number or invalid operator (i.e., valid operators are: +, -, /, and *). Assume the user enters spaces between the operands and the operator. **Read the numbers using the `Scanner`'s `nextDouble` method, and the operator using the `next` method.**

Hints:

- Use a while loop that will keep prompting the user for input as long as it is invalid.
- Use a try-catch statement to check for the numbers' validity.
- Use a selection statement (e.g., `if`, `switch`, etc.) to check for the operator's validity.

Sample run

```
Enter a simple mathematical formula: 4.5 plus 5.3
Invalid Operator. Try again.
4.5a + 5.3
Invalid number format. Try again.
4.5 + 5.3
Result: 9.8
```

Q3. [10 marks] Repeat Q2 using the Scanner's next method to read the numbers as strings and then convert them to `double` using `Double.parseDouble()`. Run your program again and enter an invalid number (e.g., 4.5a). What is the type of Exception that is thrown? When does this exception happen? Modify your code to catch and handle the new exception.

Q4. [10 marks] Consider the program skeleton below:

```
import java.io.*;
import java.util.*;

public class CopyFileCapitalized {
   public static void main(String[] args) throws Exception {
      String censoredWords[] = {"ABC", "XYZ"};
      //add code (1)
   }
   private static String replaceCensoredWords(String line, String[] censoredWords){
      //add code (2)
   }
}
```

- Write code at (2) to check `line` word by word and replace those which are listed in the `censoredWords` array with `"..."` (three dots). The method should eventually return the same line of text after replacing the censored words. One way to check the words in `line` is to use the following statement and then read the words from the `input` stream using `next()`.

  ```
  Scanner input = new Scanner(line);
  ```

- Write code at (1) to read the contents of a text file (e.g., `source.txt`) line-by-line, replace censored words using the `replaceCensoredWords` method, convert the text to uppercase, and write it to a destination file (e.g., `destination.txt`).

Don't worry too much about writing code to handling exceptions (just declare them in method headers). Samples for source.txt and destination.txt can be downloaded from Canvas.

Grading:

Q1:

+7 for defensive programming,
+8 for exception handling

Q2, Q4:

15% for logic explanation
70% for % for proper code structure and logic
15% for correct syntax and formatting

Q3:

50% for logic explanation
50% for proper code structure, correct syntax and formatting

Submission Instructions

For programming questions, explain in few, simple sentences **the algorithm you used to tackle the problem**. Add these sentences as a **block comment at the beginning of your program**. For coding questions, make sure to use appropriate code formatting and structure (e.g., indentation, brackets, etc.).

For this assignment, you need to do the following:

1- Create a Java project of which name consists of your student number followed by the assignment number, e.g., "1234567_A3".
2- Create one class for each question and write your answer inside that class. Your classes should have the same name as the question number (e.g., Q1)
3- After solving all questions, open Windows Explorer (or any other file explorer).
4- Navigate to your Java project folder (can be found inside your Eclipse workspace folder).
5- Locate the "src" folder for this project (the folder that includes the source code for all questions).
6- Zip the "src" folder and rename the zipped file to match your project name (e.g., 1234567_A2.zip).
7- Submit the zipped file to Canvas.

Note that you can resubmit an assignment, but the new submission overwrites the old submission and receives a new timestamp.