Lab 11 (15 points)

There are several learning objectives to this assignment

- Creating Interface
- Creating Class that implements Interface
- Creating an exception
- Using exceptions to handle user input errors and divide by zero errors
- Creating a driver class to test your classes

Using object oriented design:

1) Create both RationalInterface and Rational class that implements RationalInterface. Rational class has the following instance params (int numerator, int denom, double result). Create a no arg constructor that sets num and denom to 0 and result to 0.0. RationalInterface and Rational has the following methods
    a. doRational()–primary method in Rational class. Does not return anything (void):
        i. Parameters – none
        ii. Throws InputMismatchException and DivideByZeroException (passes the buck to the calling method which is main()). Since main() is try, catching, you do not need to throws for main())
        iii. Flow of method - Calls setUserInput() individually to first set inst var numerator and then second set instance var denom. After setting numerator and denom, doRational() calls calcRational(). Instance var result is assigned the output of calcRational() and the following is written to output - "With numerator <numerator> and denominator <denom>, the result is <result>". The result should be rounded out to three (3) decimal places. doRational() should then ask is the user wants to calculate another number ("enter y for yes or n for no"). The coder should ensure that y, Y, n, or N are valid responses. HINT: most of the code of doRational will need to run in a do-while loop
    b. setUserInput() – Returns an integer that assigns the result of setUserInput() to either numerator or denom instance vars. ***NOTE: This is public for the purpose of this lab to be included in the interface, but technically could be a private helper method***
        i. Parameters – none
        ii. Throws InputMismatchException (passes the buck to the calling method)
        iii. Flow of method – asks user for integer input and returns an integer to the calling method.
    c. calcRational() – calculates and returns a double based on numerator / denominator. If the denominator is 0, calcRational will throw a DivideByZeroException. ***NOTE: This is public for the purpose of this lab to be included in the interface, but technically could be a private helper method***
        i. Parameters – none
        ii. Throws DivideByZeroException (passes the buck to the calling method)
        iii. Flow of Method – test to see if denominator is 0, if so throw/ create a DivideByZeroException with a message something like "That's not kewl, don't divide by zero" with the constructor call. Otherwise, calculate and return a double to the calling method. Remember to cast here since you are doing int division!

2) Create an exception class called DivideByZeroException that extends Exception. You should create two constructors. One that is a no-arg and one that accepts a String param and calls super with the String param. NOTE-You do not have to create InputMismatchException, but you will need to import InputMismatchException for both

RationalInterface and Rational.

3) Create a driver class called RationalDemo that does the following:
   a. Creates a Rational object called rat1.
   b. Invokes doRational() based on rat1 created above in a protected block including a finally block (HINT – a try, two (2) catches, and a finally).
   c. If an InputMismatchException is caught, notify the user with a message - something like ("try running the program again, this time use integers for num and denom") in a print statement. Permission granted to be creative with the message to the user.
   d. If a DivideByZeroException is caught, write to output the exception object's message (ie e.getMessage()). This was set in Rational during the throw and constructor invocation in step 1ciii above.
   e. In order to let the user know that the program has completed running, write an output such as "Thanks for using RationalDemo, CU soon" before exiting main() in the finally block. Permission granted to be creative here.

Use Cases
1 - Run RationalDemo enter 2 (num), 3 (denom), y or Y (to play again). Then enter a non-integer input (ie decimal or String) to create a InputMismatchException throw.

```
Enter an int (whole number) for the numerator (ie 3): 2
Enter an int (whole number) for the denominator, but don't enter 0: 3
With numerator 2 and denominator 3 the result is: 0.667
Enter y to calculate another number or enter n to stop: y
Enter an int (whole number) for the numerator (ie 3): 3.4
try running again, this time use integers for num and denom
Thanks for using RationalDemo, CU soon
```

2 - Run RationalDemo enter 7 (num), 0 (denom) to create a DivideByZeroException throw

```
Enter an int (whole number) for the numerator (ie 3): 7
Enter an int (whole number) for the denominator, but don't enter 0: 0
That's not kewl, I said not to divide by zero
Thanks for using RationalDemo, CU soon
```

3 - Run RationalDemo 1 (num), 3 (denom), n or N.

```
Enter an int (whole number) for the numerator (ie 3): 1
Enter an int (whole number) for the denominator, but don't enter 0: 3
With numerator 1 and denominator 3 the result is: 0.333
Enter y to calculate another number or enter n to stop
n
Thanks for using RationalDemo, CU soon
```

Submitting your work
For all labs you will need to provide a copy of all .java files. **No need to provide .class files. I cannot read these.** ***NOTE – For Replit, please update Main.java to another name such as TempProb.java, ProChall3.java, etc.*** In addition to your .java files, you will need to provide output files of your console. The name of the output file should match the class name and have the .txt extension such as TempProbOut.txt, ProChall3Output.txt. For GUIs such as JOptionPane, you will instead need to create screenshots. For Windows users, Snipping Tool is a great way to do this. Chromebook - Shift+Ctrl+Show Windows. Mac OS users, you can see how to take screenshots using the following url - https://support.apple.com/en-us/HT201361.