Rochester Institute of Technology
National Technical Institute for the Deaf
Information and Computing Studies Department

Name: Edward Riley

NACA.161 Programming Fundamentals II
In-class Exercise #23 – User-Defined Exceptions

Overview

This exercise is designed to create and use your own exceptions classes.

Create an Exception Class

1) Create a class called **NumberOutOfRangeException** so that you use this class as an Exception class.

   What class do you need to extend?

   Exception

   What does your class header look like?

   public class NumberOutOfRangeException

   Save and compile the file.

2) Create a default constructor that passes a string to its super class. Use the following string: "The number is out of range".

   This string will be displayed whenever this exception is caught and its **toString** method is called. The error message is generic so that any class can use it. What line of code did you add to pass the string to its superclass?

   super("The Number is out of range");

3) Now provide the user of our class with more information. Instead of just saying the number is out of range, we will also let the user know what number was used.

   Create a 1-argument constructor. The argument is an integer value and contains the number that is out of range.

   What line of code did you add to pass a string with the integer value to its superclass?

   super("Number " + x + " is out of range.");

   Compile the code and make sure it compiles before continuing.

4) Create another 1-argument constructor that accepts a double value. Do the same thing you did for the previous constructor—the only difference is that this time you are using a double variable. This lets this class be used for double variables also.

   Make sure this class compiles before continuing.

5) Create another 1-argument constructor that accepts a string and passes it on the superclass. This lets the user of your class supply their own error message.

   Make sure this class compiles before continuing.

Create a Class That Uses the Your Exception Class

This class will used to implement your new exception class.

6) Create a class called MyClass with the following attributes:

Name	Type	Range of values
intVar	int	0 to 100 inclusive
dblVar	double	less than 40.0

7) Create an accessor for each attributes. Save and compile the file.

8) Create a mutator for intVar that only sets the attribute if the argument is in the acceptable range listed above. For now, don't do anything if the argument's value is invalid. Save and compile the file.

9) Now add code to the mutator from the previous step that will throw your newly created exception if the value is invalid. At this point, use the default constructor of the exception class. Don't do anything to the method header; just add the else-part of the code.

Write down the else-part of the mutator code:

throw new NumbersOutOfRange Exception

10) Save and compile the class.

What error did you get?

Unreported exception

11) How did you fix the error?

added " throws NumberOutOfRangeException " to the end of header

Save and compile the class.

12) Now create the mutator for **dblVar** as follows:
   a) Set the attribute if the argument is in the acceptable range.
   b) If the argument value is invalid, raise the **NumberOutOfRangeException** exception.
   c) Use the 1-argument constructor and pass the invalid value to the constructor.

Make sure the class compiles before continuing

13) Create a default constructor that sets **intVar** to 0 and **dblVar** to 0.0 using the mutators. Save and compile the code.

What error did you get?

No errors.

How do you fix the error?

Keine Fehler

Fix the code and make sure the class compiles before continuing.

14) Create a 2-argument constructor that sets **intVar** to the first parameter value and sets **dblVar** to the second parameter value using the mutators. Save and compile the code.

What error did you get?

NO ERRORS

How do you fix the error?

NEIN

Fix the code and make sure the class compiles before continuing.

15) Create a **toString** method to print the value of each parameter with appropriate labels. Save and compile the file.

Test Class Creation

16) Create a class called **MyTestClass** with a main method. Save and compile the code.

17) Create a MyClass object using the default constructor and compile the code.

What error did you get?

Nein

Why did this error occur?

No

How would you correct this error?

Ninguno

18) Fix the code and make sure the class compiles before continuing. You must catch the specific exception, not the generic Exception class.

19) In the try-block, set intVar to a value of –10 using the mutator.

20) In the catch-block print the **getMessage** method of the exception object.

21) Make sure the class compiles before continuing

22) Now run the program.

What message did you get?

ERROR: NumberOutOfRangeException: Number -10 is out of range

This should be the message you created in the default constructor in the exception class you created.

23) Inside the same try-block, add a line that tries to set the **dblVar** to 50.0. Add this line right after the line that calls the **intVar** mutator.

24) Compile and run the program.

Notice that you got the same error as before. In other words you only got the error message that refers to the **intVar** mutator.

Why didn't you get the error message for the **dblVar** mutator?

Because setIntVar triggered it first.

25) To be able to see both error messages, create another try/catch statement to call the dblVar mutator.

26) Compile and run the program.

What compile error did you get?

Error: NumberOutOfRangeException: 50 is out of range

27) Fix the problem with the MyClass object before continuing.

28) Run the program. If you did you see your 2 error messages, fix the code until you get 2 error messages.

What was the second error message (the one for the **dblVar** mutator)?

ERROR 50 out of range

Notice that the invalid number appears in the message. Why?

2 Triggers

29) Add code to the main so that the user can enter the value of the double to create another MyClass object. Assume the integer has a valid value of 50.

If the double value is invalid so that an exception is thrown, print the message from the exception and allow the user to enter another value for the double.

If the double value is valid, exit the loop and use the **toString** method to print the contents of this new MyClass object.

Signoff

When you complete all of the steps successfully and answer all of the questions, contact your instructor to check if your application(s) executes correctly and to review your code. We will initial the line below.

_____**Successful execution of code**

If you do not finish the program during the class period, contact your instructor to check to review your code and initial below.

_____**Code not completed during lab time**

You may then submit your work at the start of next class. You may not use the work period of the next class to complete this assignment. If you do not have a signature, then you cannot receive any points for this assignment.