

PROBLEM STATEMENT:

Exceptions must be handled in any commercial or real world application. The user must be informed regarding any exceptional/error situation and instructed as to what to do to rectify the situation (e.g., re-enter data or re-do an operation, if possible).

Notes:

For this assignment, the most relevant example in Chapter 13 of the textbook is the try-catch block presented in 13.4 and 13.6. The exception handling capabilities that you will implement in this assignment will be similar to those examples.

Be sure to handle all the exceptions that can possibly be thrown by the program for this lab exercise. Any exceptions that are not handled by your code will be displayed in the Output pane of NetBeans. In general, having unhandled exceptions is not acceptable.

CODE :

Obtain the file ExceptHandleAryDouble.java.
Create a NetBeans project for this assignment.
Compile and run the program. The program will run in a stand-alone window and it should function as follows:

The first JTextField is designed for user entry of a position or index number for the array of decimal numbers (type double). If you enter an integer in this first JTextField and hit the Enter key in this first JTextField, the program will present the value from the specified position of the array in the second JTextField.

The second JTextField is designed for user input or program output of the value of the array element at the position (index) specified in the first JTextField. If you enter an integer in the first JTextField and a double type number in the second JTextField and hit the Enter key in the second JTextField, the program will insert the value of the second JTextField into the array at the position indicated by the first JTextField.

The entire resulting modified array will be re-displayed in the JTextArea. When the user clicks the Compute button, the count, sum, and average of the entered numbers is computed and displayed

In the next steps, you will make modifications to the program that focus on exception handling.

Handle exceptions thrown because of the entered index (position) number in first JTextField.

If the user enters any illegal text in the first JTextField, an exception is thrown by the program when it tries to convert the text string to an integer. A proper integer number consists of digits only. That is, to create an integer, the user cannot enter a decimal point, comma, letter, other punctuation, space between digits, etc.

You must find all the places in which this text-to-int conversion takes place, and add a try-catch block to catch and handle any exception that might be thrown.

When you insert the try-catch blocks, use the example on textbook page 551 as an example.

The existing statement in the program that tries to convert text to an int must be placed in the try-clause (within the curly braces of the try block).

In the parenthesized parameter declaration of the catch-clause, just use Exception as the exception type, as shown in the example on textbook page 551. This will catch all exceptions since the Exception class is the topmost parent class of the entire hierarchy of exception classes. That is, all exception classes are subclasses of the Exception class, either directly or indirectly.

You only need to use one catch-clause for each try-clause in this program (to keep it simple).

You must write code in the catch-clause to handle any exception that might be thrown and bound to the parameter variable declared in the initial set of parentheses of the catch-clause. See the next paragraphs for details on the exception handling code.

Your code to handle the exception in the catch-clause must present a JOptionPane dialog that displays the following information, each on a separate line:

Statement telling the user that illegal data was entered.

Statement indicating whether the illegal data was entered in the first or second JTextField.

Statement telling the user which operation was being attempted, either (1) getting a decimal number from the array or (2) inserting a decimal number into the array.

Statement telling the user to please try to enter the data again.

The string returned by the toString method of the exception object that was caught (bound to the Exception parameter for the catch-clause). That is, assuming that the parameter is named e, then invoke e.toString(). The string returned by this toString method will specify the package and class of the exception object and a message providing additional information regarding the exception.

Handle exceptions thrown because of the number entered in the second JTextField for insertion into the array.

If the user enters any illegal text in the second JTextField, an exception is thrown by the program when it tries to convert the text string to a number of type double.

A proper number of type double consists only of digits, one decimal point, and an optional negative sign. That is, to create a double type number, the user cannot use more than one decimal point; and cannot use commas, letters, other punctuation, spaces between digits, etc.

Do all of the same tasks as listed under Step #5 above, but this time for the data entered in the second JTextField.

Handle exceptions thrown for index out of bounds.

If the user enters a number in the first JTextField that is not within the range of allowable index numbers for the array, an exception will be thrown.

You must find all the places in the code where this problem might arise, and add a trycatch block to catch and handle any exception that might be thrown.

Again, to keep the program simple, use just one catch-clause for the try-clause. Use type Exception in the parenthesized parameter declaration of the catch-clause so as to catch all index-out-of-bounds exceptions using the one catch-clause.

Your code to handle the exception in the catch-clause must present a JOptionPane dialog that displays the following information, each on a separate line. This is similar to the information displayed for the other exception cases described above:

Statement telling the user that the number entered was out of bounds.

Statement indicating whether the illegal data was entered in the first or second JTextField.

Statement telling the user to please try to enter the data again, and what the allowable range is. Reference the appropriate variable(s) to do this. Do not hardcode the range.

The string returned by the toString method of the exception object that was caught (bound to the Exception parameter for the catch block).

The string returned by this toString method will specify the package and class of the exception object and a message providing additional information regarding the exception.

Handle NaN result from division by zero.

Currently, if no numbers have been entered into the array and if the user clicks the button to compute the sum and average of the entered numbers, the result from the division operation is NaN. This symbol stands for a special constant of type double, called "Not-a- Number". This is because division by zero is undefined in mathematics. Although no exception is thrown in this situation, this situation should be handled in a better manner.

Your job is to modify the code of the actionPerformed method of the ButtonListener class so that it tests for the possibility of a zero-divisor or NaN result. If this situation occurs, the message displayed to the user should include an additional statement explaining the NaN result.

Compile, execute, and debug the program.

Make window captures of your executing program to demonstrate that the requirements of this assignment have been met. There are seven cases for which you need to make window captures, as listed below. For each of the first five cases, insert a pair of windows, namely (1) the main JFrame window showing the erroneous user input, and (2) the resulting JOptionPane window that is displayed showing the output text when the exception is detected and handled. You need to make window captures for the following seven cases:

1. Erroneous input in first JTextField when user hits the Enter key in the first JTextField.
2. Erroneous input in first JTextField when user hits the Enter key in the second JTextField.
3. Erroneous input in second JTextField when user hits the Enter key in second JTextField.
4. Index out of bounds when user hits the Enter key in first JTextField.
5. Index out of bounds when user hits the Enter key in second JTextField.
6. NaN output occurs.
7. Correct inputs and outputs, with no exceptions arising.

See the window captures below for examples.

Create batch file.

Create a batch file as you did for the previous assignments so that your program can be executed without the use of NetBeans.
Be sure that the batch file is within your top level NetBeans project folder.

As usual create ReadMe.pdf file.

Create a file named ReadMe.pdf

In this document, insert your name at the top, and on the next line insert the assignment number

Then enter any comments regarding the assignment and your program.

Then insert several window captures of windows showing the inputs and outputs from the execution of the program.

Insert at least two window captures of your JFrame window showing the inputs and output from the execution of the program.

The first window capture should be captured when you have entered data on the 3rd taxpayer; it should show the data in the JTextFields and selected items in the JComboBoxes.

The second window capture should show the updated list of taxpayers in the JTextArea that is the result of hitting the Enter key in one of the JTextFields for this new 3rd taxpayer.

Be sure the ReadMe file is within your top level project folder.

Zip the project folder and all its contents.

Change the name the zip file so that its name consists of your name along with the assignment number, as follows: " LastName_ Lab_11_cs209.zip".

Do not use spaces in the name of the file, use underscores or hyphens instead.

Deliverables:

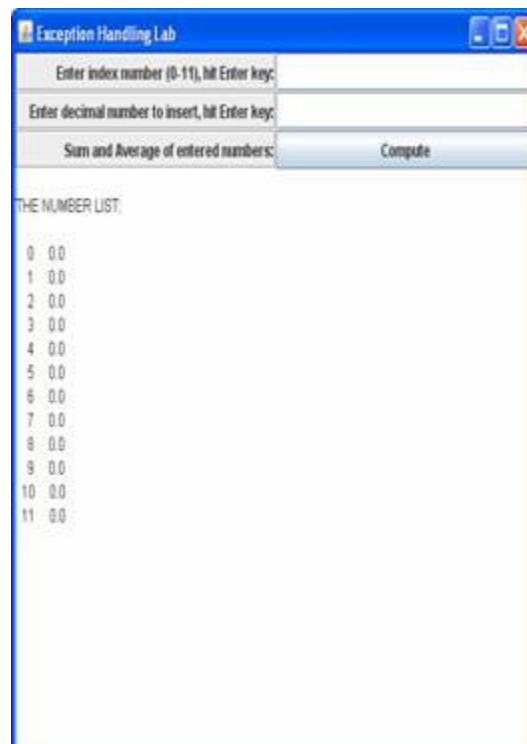
Send to duttat@ecc.edu an email this the exact subject

cs209_Lab_10

In this email attached the above named zip file

LastName_Lab_10_cs209.zip

Due Date : 5:00pm 6 November 2014



Enter index number (0-11), hit Enter key:	Enter decimal number to insert, hit Enter key:	Sum and Average of entered numbers:	Compute
THE NUMBER LIST:			
0	0.0		
1	0.0		
2	0.0		
3	0.0		
4	0.0		
5	0.0		
6	0.0		
7	0.0		
8	0.0		
9	0.0		
10	0.0		
11	0.0		

Exception Handling Lab

Enter index number (0-11), hit Enter key: a1


Enter decimal number to insert, hit Enter key: 0.0

Sum and Average of entered numbers: Compute

THE NUMBER LIST:

0	: 0.0
1	: 0.0
2	: 0.0
3	: 0.0
4	: 0.0
5	: 0.0
6	: 0.0
7	: 0.0
8	: 0.0
9	: 0.0
10	: 0.0
11	: 0.0

Message

 Illegal data entered for index in first textbox
Please try again to choose the location in the array
java.lang.NumberFormatException: For input string: "a1"

OK

Exception Handling Lab

Enter index number (0-11), hit Enter key: 2

Enter decimal number to insert, hit Enter key: 78ac3

Sum and Average of entered numbers:

THE NUMBER LIST:

0	0.0
1	0.0
2	0.0
3	0.0
4	0.0
5	0.0
6	0.0
7	0.0
8	0.0
9	0.0
10	0.0
11	0.0

Message

 Incorrect data type entered
Please try again to enter a real number into the array
java.lang.NumberFormatException: For input string: "78AC3"

Exception Handling Lab

Enter index number (0-11), hit Enter key:	15
Enter decimal number to insert, hit Enter key:	0.0
Sum and Average of entered numbers:	Compute

THE NUMBER LIST:

0	0.0
1	0.0
2	0.0
3	0.0
4	0.0
5	0.0
6	0.0
7	0.0
8	0.0
9	0.0
10	0.0
11	0.0



Exception Handling Lab

Enter index number (0-11), hit Enter key: 5

Enter decimal number to insert, hit Enter key: 34.08

Sum and Average of entered numbers: Compute

THE NUMBER LIST:

0	0.0
1	0.0
2	0.0
3	0.0
4	12.5
5	0.0
6	0.0
7	0.0
8	0.0
9	0.0
10	0.0
11	0.0

Number of entered decimal numbers is: 1
Sum of the entered numbers is: 12.5
Average of the entered numbers is: 12.5

Exception Handling Lab

Enter index number (0-11), hit Enter key:

Enter decimal number to insert, hit Enter key:

Sum and Average of entered numbers:

THE NUMBER LIST:

0	.00
1	.00
2	.00
3	.00
4	12.5
5	34.09
6	.00
7	.00
8	.00
9	.00
10	.00
11	.00

Number of entered decimal numbers is: 2
Sum of the entered numbers is: 46.59
Average of the entered numbers is: 23.295