IST 256
Lab
Week 8 – Tuesday, March 8, 2011

1. Understanding how to write a method

Write a method according to the following

1. There is only one parameter and it is an integer
2. The method does the following:
    - If the value of the integer is between 0 to 99, then return the message "o.k."
    - If the value of the integer is smaller then 0 or larger then 99, then return the message "Mistake!"

First write the method header line for this method. You will need to decide on a method name and what is the type of the result.

Next write the method body to do the computation described. Do you need any local variables? Don't forget to include a return statement.

2. Write a program with exception handling for NumberFormatExceptions

In this lab, we will write a simple program that reads two numbers and displays them in a label. But we will add exception handling for converting one number to an integer and the other to a double.

a. Here is a sketch of the user form:

```
                Testing NumberFormat Exceptions

        Enter an integer              |_______________|  (textfield)
        Enter a decimal number         |_______________|  (textfield)

                        |__  Check Numbers  __|    (button)

                Result:                         (label)
```

Create the Event -> Action -> actionPerformed method for the button.

b. In the code for the button actionPerformed method, declare an integer and a double variable.

Read the two textfields into these variables, converting the first one to an integer and the second one to a double.

Display the two numbers in the result label.

c. After the variable declaration and **before** the line which uses Integer.parseInt, add lines with the beginning of the **try** block:

```
        try
        {
```

Indent the rest of the button code and put the matching bracket at the end:

```
        }
```

The reason that we are putting all the button code inside the try block is that we want to skip the rest of the button and return to the user to let them try again. Put the catch code:

```
        catch (NumberFormatException e)
        {
                // print out the Exception
                System.out.println(e.toString());
                // if a number error occurs, show message and return to the user
```

```
            javax.swing.JOptionPane.showMessageDialog(null,
                    "Please enter valid numbers");
    }
```

d. Test this program.

Try several ways to type in numbers, letters or special characters that will not be either a valid integer or double. Note the message in the output pane as well as the dialog box.

Show examples which you tried and gave NumberFormatExceptions:

Examples:

Hand in your lab sheet by next Tuesday, March 15.