**Institute of Technology Tralee**

**Computing Department**

**Object Oriented Programming 1**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Session 6 – Review of some past CAs on Input Validation**

A Java program is required that will read in a single date of birth value in the form dd-mm-yyyy.

This date of birth value must be partially validated. We will take it here that only the first 6 characters in the date of birth require validation, we will assume that the user supplies a valid value for the year section of the date of birth.

In order for the date of birth value to be considered valid in our case,

* It must be exactly 10 characters in length
* The 3rd and 6th characters must both be dashes
* The first 2 characters must both be digits
* The day part must convert to an integer in the range 1-31
* The 4th and 5th characters must both be digits
* The month part must convert to an integer in the range 1-12
* The maximum value for the day part must correspond to the month part for the date of birth concerned. To validate this condition you must use the fact that there are 31 days in months 1,3,5,7,8,10,12, 30 days in months 4,6,9 and 11 and for simplicity here we will take it that month 2 has 28 days.

If an invalid date of birth is supplied, the user should receive a suitable error message indicating what was wrong with the date of birth and be asked to re-enter.

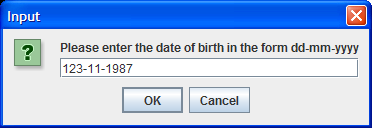
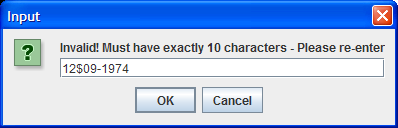
Once a valid date of birth has been supplied, a message dialog will display the valid date of birth entered.

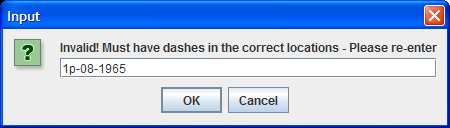
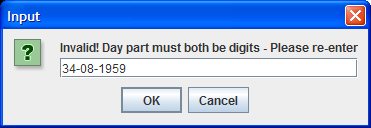
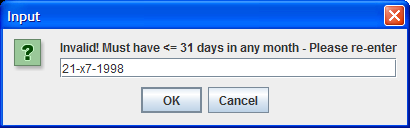
For full marks here your program should, along with a logically correct solution for the problem above, include the usual **single-line** and **multi-line comment** at the top of the program. The multi-line comment should briefly explain the purpose of the program.

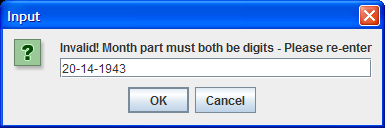
Your program should run as indicated in the following sample screenshots. You should use the values indicated in the screenshots when testing your program.

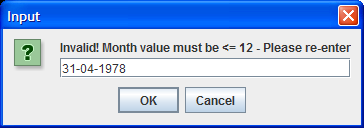
**Sample Screenshots**

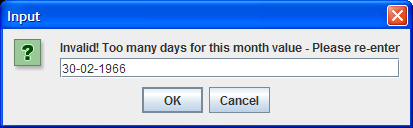
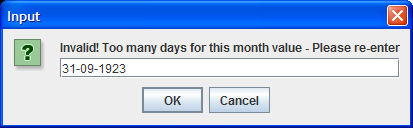
**The user begins by entering a number of invalid dates of birth – they are told why the date of birth supplied is invalid and asked to re-enter:**

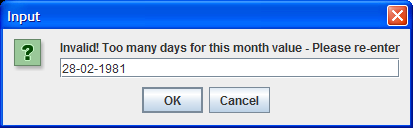
 

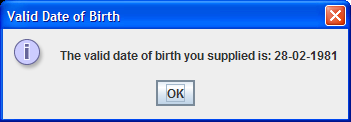






**… eventually the user enters a valid date of birth and it is displayed on a message dialog**



**(b)**

Stock market indices, such as the ISEQ in Ireland and the NASDAQ in the US are always given in the format:

XXXXXXX.YY

Where XXXXXXX refers to any amount of digits (at least one) and YY refers to exactly 2 digits

Therefore a valid index must

* have at least 4 characters
* begin with a digit
* have a decimal point (dot) as the 3rd last character
* end with 2 digits
* contain all digits before the decimal point

Examples of valid index values would be 0.96, 123.45, 7689.33 and 52.71

You must write a Java program that reads in an arbitrary number of index values (you can take it here that the sentinel value will be the string “x”). When a value is entered, it should be firstly tested to ensure that it contains at least 4 characters and, if it passes this test it should then be checked that it begins with a digit, then if it passes this test it should proceed to carry out the other tests as required, one by one, as indicated above. If it fails on any of the tests, the user should receive an error message at the first point of failure, telling them why it failed and be asked to re-enter the index.

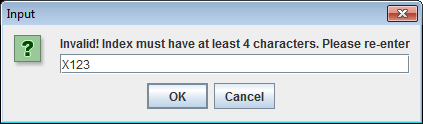
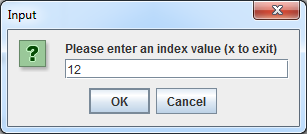
If it passes all tests, then the index should be added to a list of valid indices. Then the user will get asked for the next index and the main loop will stop whenever the user supplies the value “x” for the index. At this stage a message dialog will appear with a list of the valid indices.

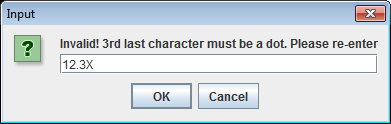
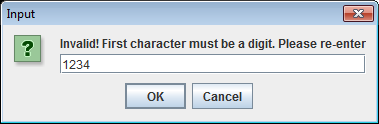
For full marks here, your program should, along with a logically correct solution for the problem above, include the usual **single-line** and **multi-line comment** at the top of the program. The multi-line comment should briefly explain the purpose of the program.

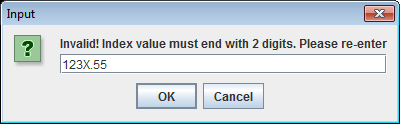
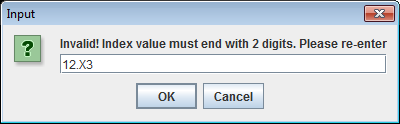
Your program should run as indicated in the following sample screenshots. Also, you can use the test values indicated when testing your own program.

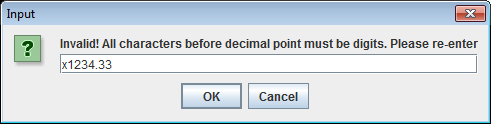
**Sample Screen Shots**

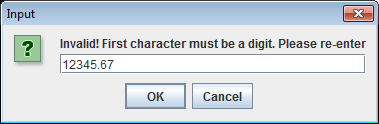
**The main loop begins and here the user enters several bad index values before eventually entering a valid one. The user gets told what was wrong with the index each time. Once a good index is entered the validation loop finishes**



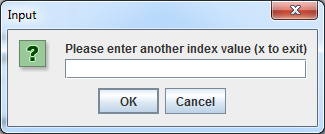








**The good index value is then added to a list of valid indices and then the next iteration of the main loop begins**



**…… and so it continues**

**When the user finally enters “x” for the index value the main loop stops and a message dialog appears displaying the list of valid indices:**

