Assignment 2 (Exception Handling)
Submission: via Blackboard
Points: 50

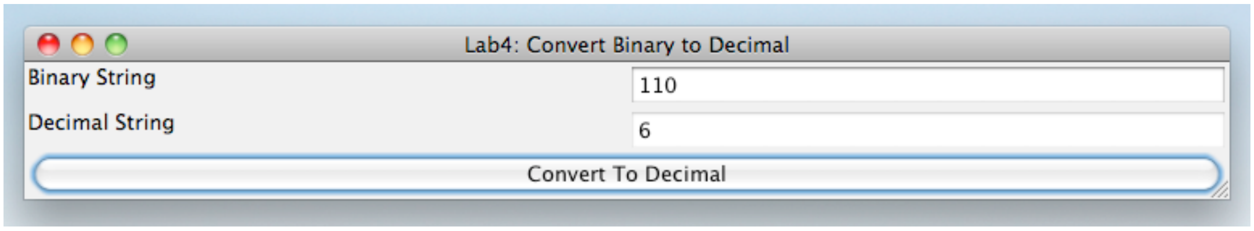
Objectives:

1. Follow software specifications provided to design and implement the following program
2. Perform a code review using the checklist provided
3. Provide appropriate Javadoc documentation and generate HTML documentation files
4. Thoroughly test your program and provide 4 test cases (for different possible inputs)

Program Specification:

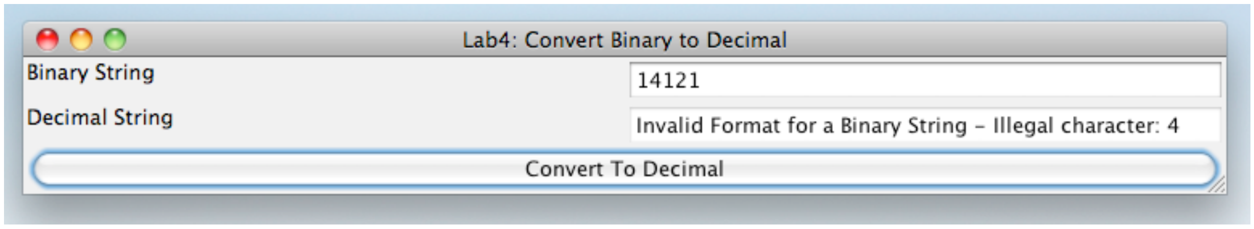
Implement a method **parseBinary(String binaryString)**, which converts a binary string into a decimal number. Implement **parseBinary** method to throw a **NumberFormatException** if the string is not a binary string. NumberFormatException is predefined in the Java. Create a program called Assignment2.java with UI as shown in figure 1 and 2 with two TextField's to receive the user's input binary string and display the decimal string or a error message in case an exception occurs. This program should use the parseBinary method to perform the conversion and display the results.

(Hint: For the UI, refer to last semester's notes on GUI. Create a Frame with a panel on it. Add 2 text fields and a button on the panel)



Binary String 110
Decimal String 6

Figure 1: Screenshot showing valid inputs



Binary String 14121
Decimal String Invalid Format for a Binary String – Illegal character: 4

Figure 2: Screenshot showing the NumberFormatException message

SUBMISSION:

Submit your assignment via Blackboard The submission package should have the following:

- Source code: Assignment2.java
- Code Review checklists (marked)
- HTML documentation: HTML and supporting files
- 4 test cases with inputs and expected outputs

Code Review Checklist – Java

1. Specification / Design

[ ] Is the functionality described in the specification fully implemented by the code?

[ ] Is there any excess functionality in the code but not described in the specification?

2. Initialization and Declarations

[ ] Are all local and global variables initialized before use?

[ ] Are variables and class members of the correct type and appropriate mode

[ ] Are variables declared in the proper scope?

[ ] Is a constructor called when a new object is desired?

[ ] Are all needed import statements included?

3. Method Calls

[ ] Are parameters presented in the correct order?

[ ] Are parameters of the proper type for the method being called?

[ ] Is the correct method being called, or should it be a different method with a similar name?

[ ] Are method return values used properly? Cast to the needed type?

4. Arrays

[ ] Are there any off-by-one errors in array indexing?

[ ] Can array indexes ever go out-of-bounds?

[ ] Is a constructor called when a new array item is desired?

5. Object Comparison

[ ] Are all objects (including Strings) compared with "equals" and not "=="?

6. Output Format
[ ] Are there any spelling or grammatical errors in displayed output?
[ ] Is the output formatted correctly in terms of line stepping and spacing?

7. Computation, Comparisons and Assignments
[ ] Check order of computation/evaluation, operator precedence and parenthesizing
[ ] Can the denominator of a division ever be zero?
[ ] Is integer arithmetic, especially division, ever used inappropriately, causing unexpected truncation/rounding?
[ ] Check each condition to be sure the proper relational and logical operators are used.
[ ] If the test is an error-check, can the error condition actually be legitimate in some cases?
[ ] Does the code rely on any implicit type conversions?

8. Exceptions

[ ] Are all relevant exceptions caught?
[ ] Is the appropriate action taken for each catch block?
[ ] Are all appropriate exceptions thrown?

9. Flow of Control

[ ] In a switch statement is every case terminated by break or return?
[ ] Do all switch statements have a default branch?
[ ] Check that nested if statements don't have “dangling else” problems.
[ ] Are all loops correctly formed, with the appropriate initialization, increment and termination expressions?
[ ] Are open-close parentheses and brace pairs properly situated and matched?

10. Files

[ ] Are all files properly declared and opened?
[ ] Are all files closed properly, even in the case of an error?
[ ] Are EOF conditions detected and handled correctly?
[ ] Are all file exceptions caught?