CMSC204 – Prof. Gary Thai

Project 1 – PasswordCheckerUtility

**Lessons Learned**

This project focused on building a utility class to perform operations to validate one or more passwords (and return different data depending on whether those validation attempts were successful or not), a JavaFX GUI to allow a user to graphically interact with the program, and a JUnit test class that would automatically run tests on the utility class to verify its functionality.

In building the utility class, I learned more about creating custom Exception classes and how to throw them. I also learned about planning out the logic in my program, to decide what happens when. In this case, I decided to have my utility class handle data processing and have the logic of just what to do with the data happen in JavaFX (ex: Exception handling), effectively combining the JavaFX GUI with my main driver class.

On the GUI end, I deepened my understanding of JavaFX with implementing mnemonics and tooltips and revisited creating custom Alert windows when catching Exceptions. I also practiced using ArrayLists when reading a file of unknown length. In building my JUnit test cases, I discovered that I could tell a test case to expect an exception to be thrown by using the (expected = ) tag after the @Test flag.

One issue that I ran into was the IDE insisting that I serialize my nested inner Exception classes. This is something that I haven’t run into previously and will need to take a deeper look to see what makes the IDE insist on flagging my classes this way. A quick fix in the meantime has been adding a @SuppressWarnings(“serial”) flag to my nested classes.

**Assumptions**

* No restrictions on alphabet or symbols. The only restrictions would be the established password complexity rules
* Files to be read will all be .txt format
* Files to be read will all have passwords in a list, with one password per line
* Passwords can include whitespace (not including newline characters)
* “Weak” but otherwise valid passwords will not be recognized as a problem by the GUI
* Since the function of the program is to check passwords for validity, passwords will be kept in plaintext and will not need character masking

**Pseudocode**

In class PasswordCheckerUtility:

* Method isValidPassword
  + If password’s length is fewer than 6 characters, throw an exception
  + Call containsUpperCaseLetter() to verify that the password contains an upper case character
    - Throw an exception if false
  + Call containsLowerCaseLetter() to verify that the password contains a lower case character
    - Throw an exception if false
  + Call containsNumber() to verify the password contains a number
    - Throw an exception if false
  + Call hasTriples() to verify if the password contains a sequence of 3 identical characters
    - Throw an exception if true
  + If no exceptions are thrown due to the above, then return true to report the password as valid
* Method containsNumber
  + Examine all characters in the String
  + Return true if a numerical digit character is encountered
* Method containsUpperCaseLetter
  + Examine all characters in the String
  + Return true if an uppercase character is encountered
* Method containsLowerCaseLetter
  + Examine all characters in the String
  + Return true if a lowercase character is encountered
* Method hasTriples
  + Examine all characters in the String
  + Return true if a character is followed by two more identical characters
* Inner exception class LengthException
  + Extend RuntimeException
  + To be thrown if a read password has fewer than 6 characters
  + Error message: “The password must be at least 6 characters long”
* Inner exception class NoUpperAlphaException
  + Extend RuntimeException
  + To be thrown if a read password has no uppercase characters
  + Error message: “The password must contain at least one uppercase alphabetic character”
* Inner exception class NoLowerAlphaException
  + Extend RuntimeException
  + To be thrown if a read password has no lowercase characters
  + Error message: “The password must contain at least one lowercase alphabetic character”
* Inner exception class NoDigitException
  + Extend RuntimeException
  + To be thrown if a read password has no numerical digits
  + Error message: “The password must contain at least one digit”
* Inner exception class InvalidSequenceException
  + Extend RuntimeException
  + Error message: “The password cannot contain more than two of the same character in sequence”

In class PasswordMain:

* Method readFile
  + Open a FileChooser window for user to select file for reading
    - User limited to .txt files
  + Scanner reads through the file
  + Each line of file is added to an ArrayList as a new String element
  + When there are no more lines to be read, return the ArrayList to the method caller
* Text Fields passwordText and passwordAText
  + Text fields for user to enter passwords for testing
* Button checkPwdButton
  + Check that the text in passwordText and passwordAText matches
    - Throw an Alert if they do not match
  + Call isValidPassword to validate the entered password
  + Pop up an Alert window:
    - If isValidPassword returned True, the Alert will report the valid password
    - If isValidPassword threw an exception, the exception will be caught and the error message displayed
  + Button has a tooltip to briefly explain its function
  + Mnemonic is ‘p’
* Button exitButton
  + Closes the program
  + Has a tooltip to explain its function
  + Mnemonic is ‘x’
* Button checkPwdsInFileButton
  + Calls validPasswords to open and read a .txt file
    - If no file is found, displays an Alert
    - If the user closes the FileChooser window (NullPointerException), do nothing
    - If the returned ArrayList is populated with invalid passwords, display an Alert window of the invalidated passwords and their error messages
    - If the returned ArrayList is empty, display an Alert stating that all passwords are valid
  + Has a tooltip to explain its function
  + Mnemonic is ‘f’

|  |
| --- |
| **NoLowerAlphaException** |
|  |
|  |

**UML Diagrams**

|  |
| --- |
| **PasswordCheckerUtility** |
|  |
| +PasswordCheckerUtility()  +isValidPassword (String) : boolean  +containsNumber(String) : boolean  +containsUpperCaseLetter(String) : boolean  +containsLowerCaseLetter(String) : boolean  +hasTriples(String) : boolean  +validPasswords(ArrayList<String>) : ArrayList<String>  +isWeakPassword(String) : boolean |

|  |
| --- |
| **NoUpperAlphaException** |
|  |
|  |

|  |
| --- |
| **LengthException** |
|  |
|  |

|  |
| --- |
| **NoDigitException** |
|  |
|  |

|  |
| --- |
| **InvalidSequenceException** |
|  |
|  |

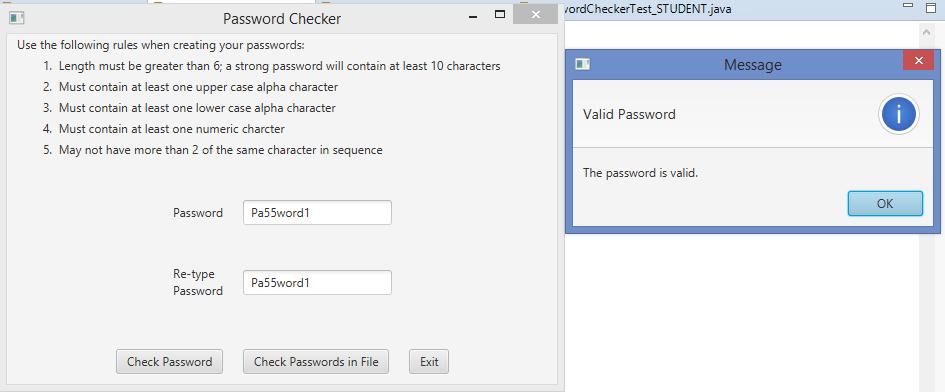
|  |
| --- |
| **PasswordMain** |
|  |
| +readFile() : ArrayList<String> |

|  |
| --- |
| **PasswordDriverFX** |
|  |
| +main(String) : void  +start(Stage) : void |

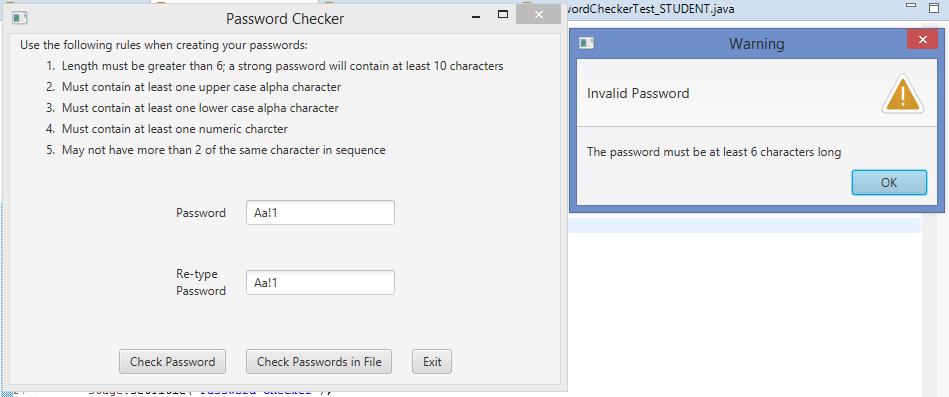
**Test Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test # | Input | Expected Output | Actual Output | Pass? |
| 1 | Entry: Pa55word1 Re-entry: Pa55word1 | Valid password window | Valid password window | Y |
| 2 | Entry: Aa!1 Re-entry: Aa!1 | Fail: not 6 characters | Fail: not 6 characters | Y |
| 3 | Entry: SuperSecureExtraLongPassword Re-entry: SuperSecureExtraLongPassword | Fail: no digit | Fail: no digit | Y |
| 4 | Entry: password1! Re-entry: password1! | Fail: no uppercase letter | Fail: no uppercase letter | Y |
| 5 | Entry: TEST5!@# Re-entry: TEST5!@# | Fail: no lowercase letter | Fail: no lowercase letter | Y |
| 6 | Entry: Pa555word1  Re-entry: Pa555word1 | Fail: too many characters in sequence | Fail: too many characters in sequence | Y |
| 7 | Entry: Pa55word1  Re-entry: Pa55word2 | Fail: passwords must match | Fail: passwords must match | Y |
| 8 | Entry: (blank)  Re-entry: (blank) | Fail: must be at least 6 characters long | Fail: must be at least 6 characters long | Y |
| 9 | Read File: passwordTest.txt Note: file contains all valid passwords | Pass: No invalid passwords found | Pass: No invalid passwords found | Y |
| 10 | Read File: passwordTest2.txt Note: file contains four invalid passwords and one valid password | Pop up window with invalid passwords and their Exceptions. Valid password is not included | Pop up window with invalid passwords and their Exceptions. Valid password is not included (see screenshot) | Y |

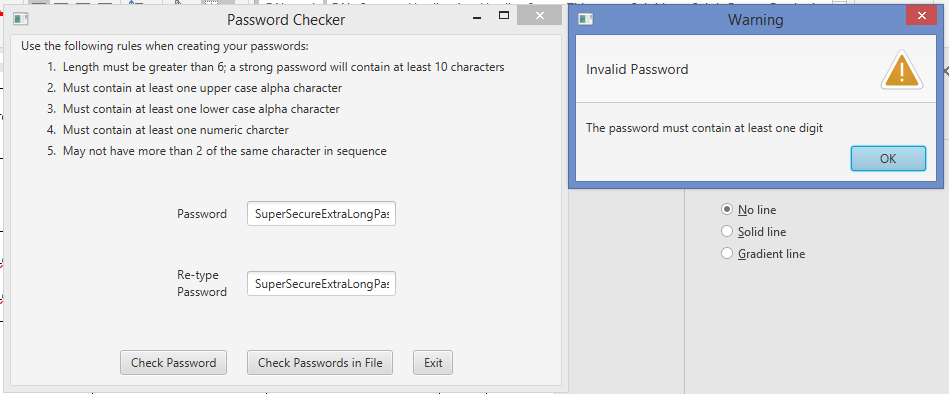
**Test Screenshots**

Test 1

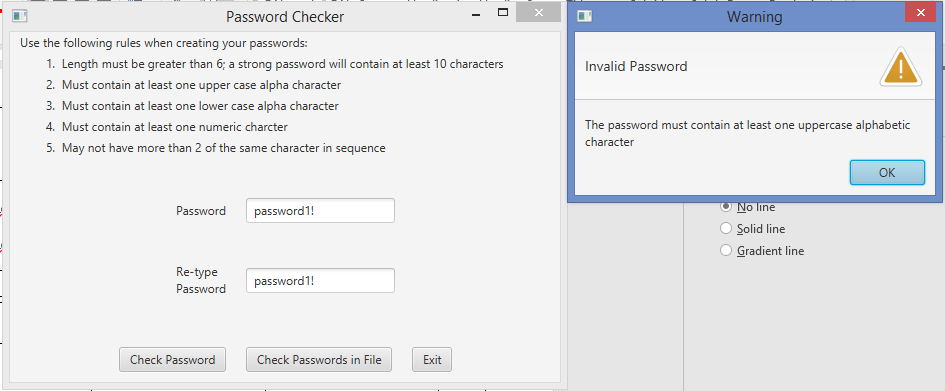
Test 2



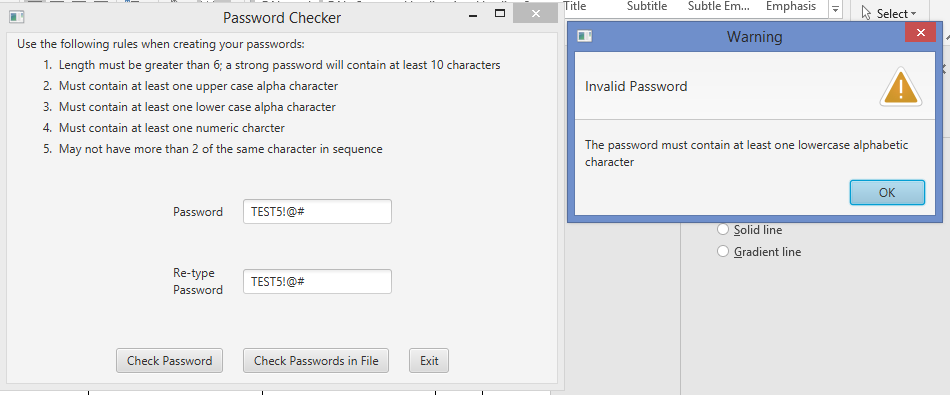
Test 3



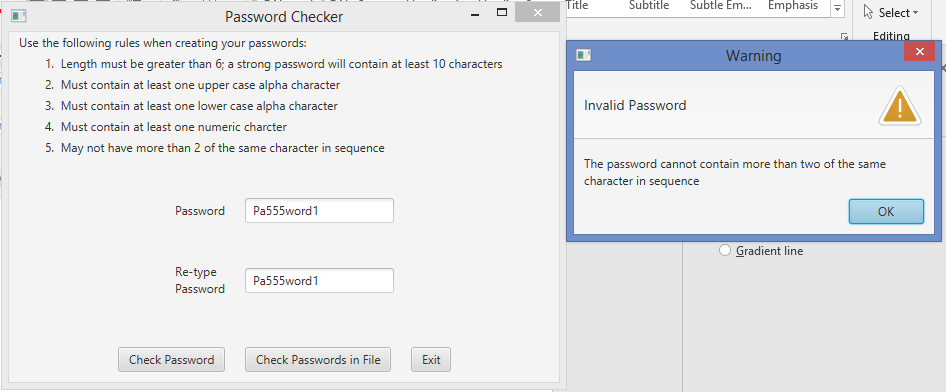
Test 4



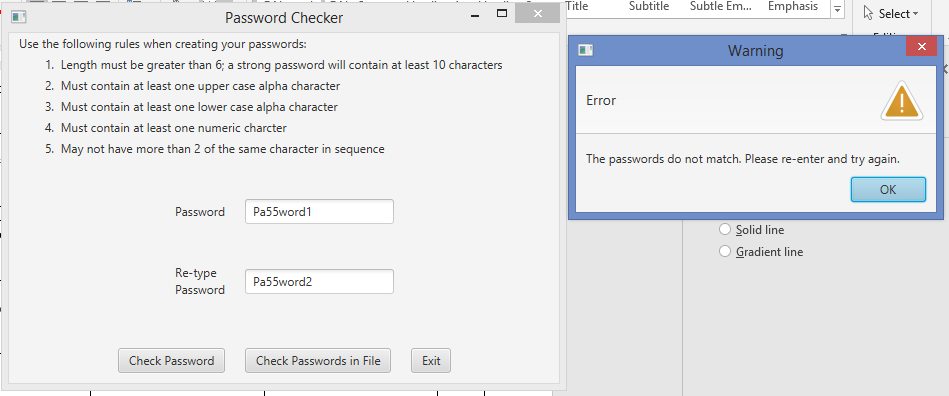
Test 5



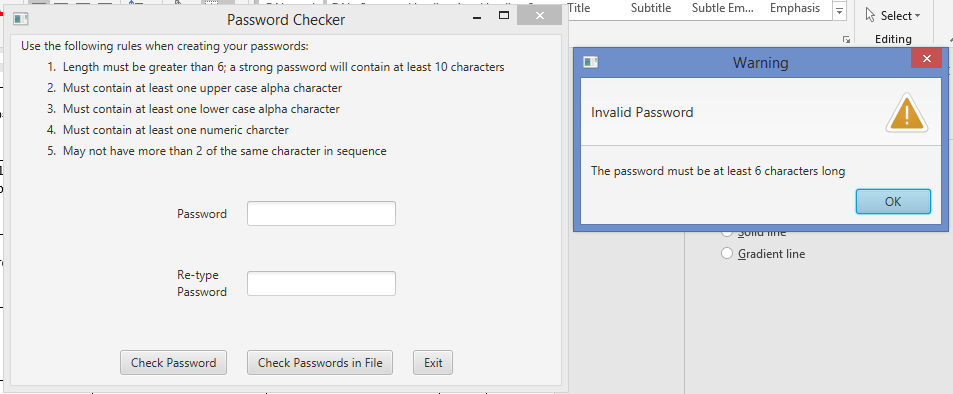
Test 6



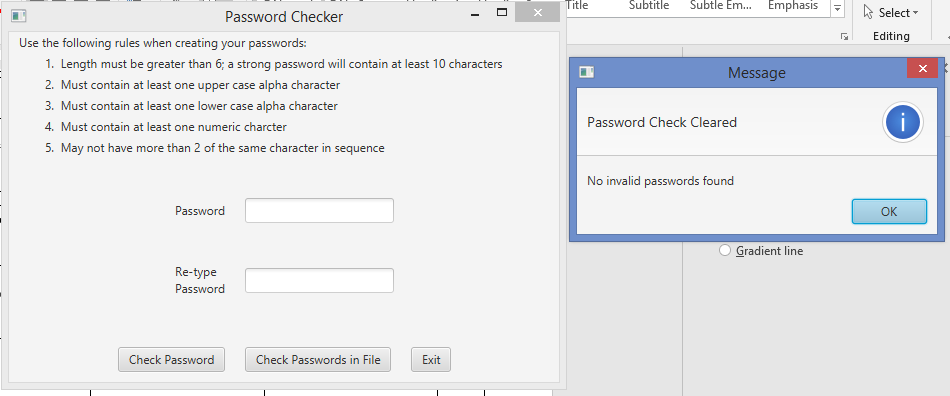
Test 7



Test 8



Test 9



Test 10

