**Assignment 1**

**LEARNING EXPERIENCE:**

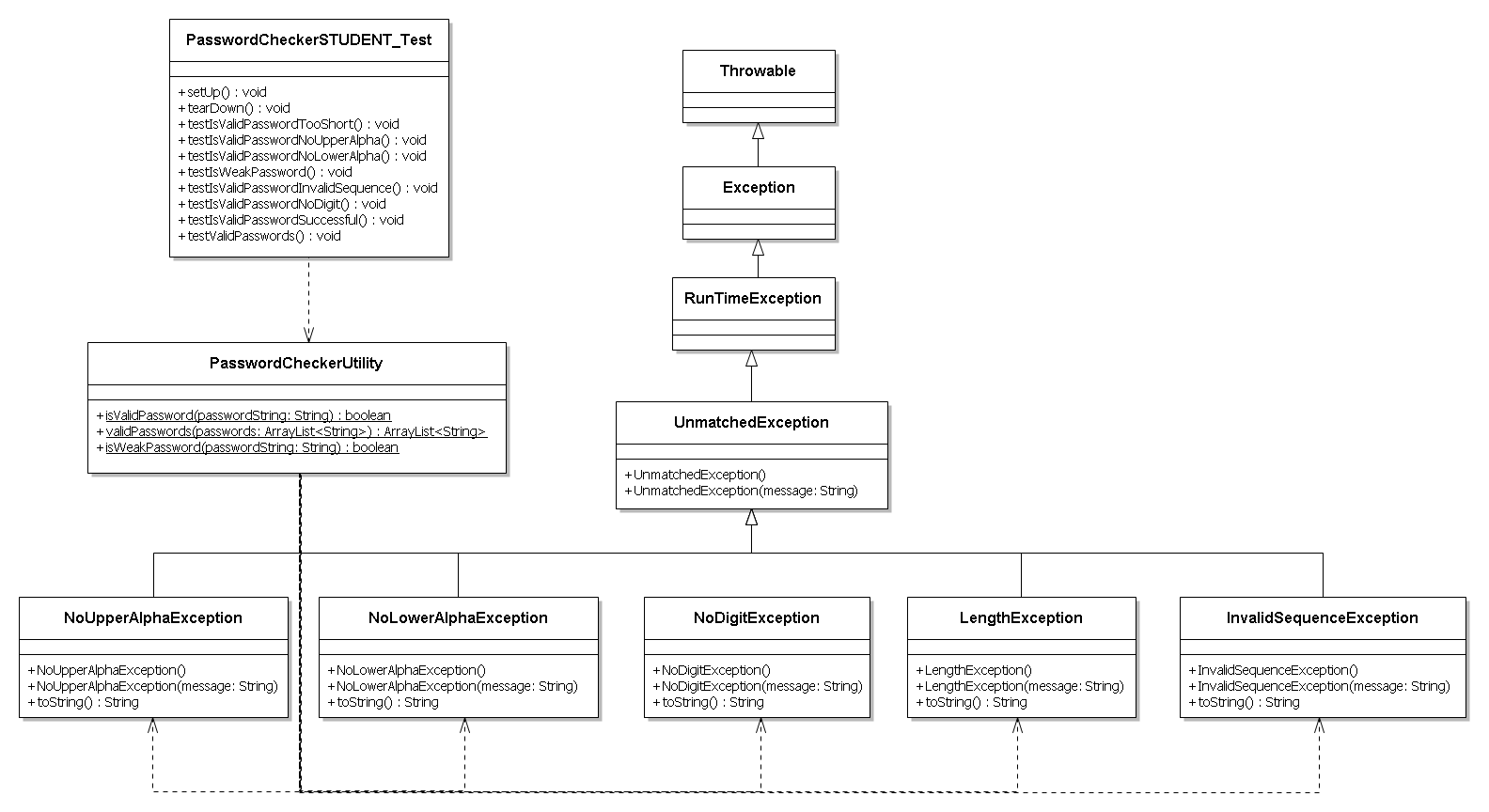
The main goal of this assignment is to practice exception handling and understanding inheritance when we handle error using Java. There are many concepts are covered in this assignment including inheritance, polymorphism, ArrayList, regular expressions, Junit tests and java documentation.

I spent few hours to review all concepts and complete the assignment. I got familiar with important concepts in Java including polymorphism, inheritance and exception handling. The most challenging part of the assignment was to design “PasswordCheckerUtility”; specially, to design a method which checks an “ArrayList” of passwords and return an ArrayList with the status of any invalid password. I used “try-catch” block inside a loop. “toString” method of each child class invokes during the run time and polymorphism mechanism helps me to add invalid password status into invalidPasswords ArrayList.

I noticed that when I ran my code against the PasswordCheckerTest, there was a discrepancy to the specification. According to the assignment if more than one error is presented in a password, NoUpperAlphaException has priority to OneDigitException. For example, if a password is “abcdef”, it fails rules 2 and 4 and Throw a NoUpperAlphaException, not a NoDigitException. I fixed the PasswordCheckerTest. The important lesson is following: always design and test cases must be verified against specification. If design only relies on test cases or vice versa, there is a possibility of introducing bug.

When I completed the assignment, I reviewed and learned with following concepts: Exceptions, Try-catch blocks, Inheritance, Polymorphism, ArrayList, Regular Expressions, Junit tests and Java documentation.

**Final Block Diagram of Design:**

Here is the UML diagram of the project. The UML diagrams are available in “doc” directory.

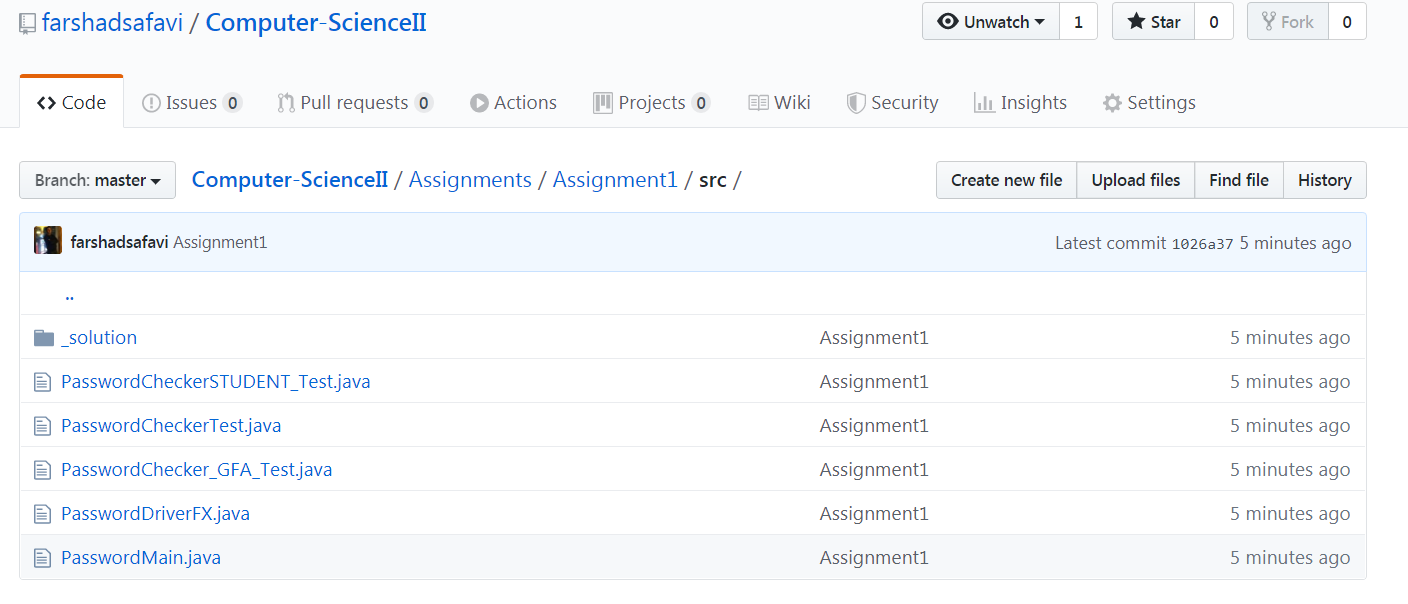
**Test Table:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case #** | **Input** | **Actual Input**  **Same as input** | **Expected Output** | **Actual Output**  **Same as expected** | **Did the test pass?** |
| 1 | 123Password |  | Password is valid |  |  |
| 2 | pawor |  | Error  Length Exception |  |  |
| 2 | password |  | Error  NoUpperAlphaException |  |  |
| 3 | PASSWORD |  | Error  NoLowerAlphaException |  |  |
| 4 | Password |  | Error  One Digit Exception |  |  |
| 5 | Passsword |  | Error  One Digit Exception |  |  |
| 6 | Passsword1 |  | Error  InvalidSequenceException |  |  |
| 7 | Pas123f |  | OK but Weak |  |  |
| 8 | 123456 |  | Error  NoUpperAlphaException |  |  |
| 9 | 123456A |  | Error:  NoLowerAlphaException |  |  |
| 10 | 123456AbaaAAbb |  | Password is valid |  |  |
| 11 | 123456AbaaAAbbb |  | Error  InvalidSequenceException |  |  |

**Screenshots:**

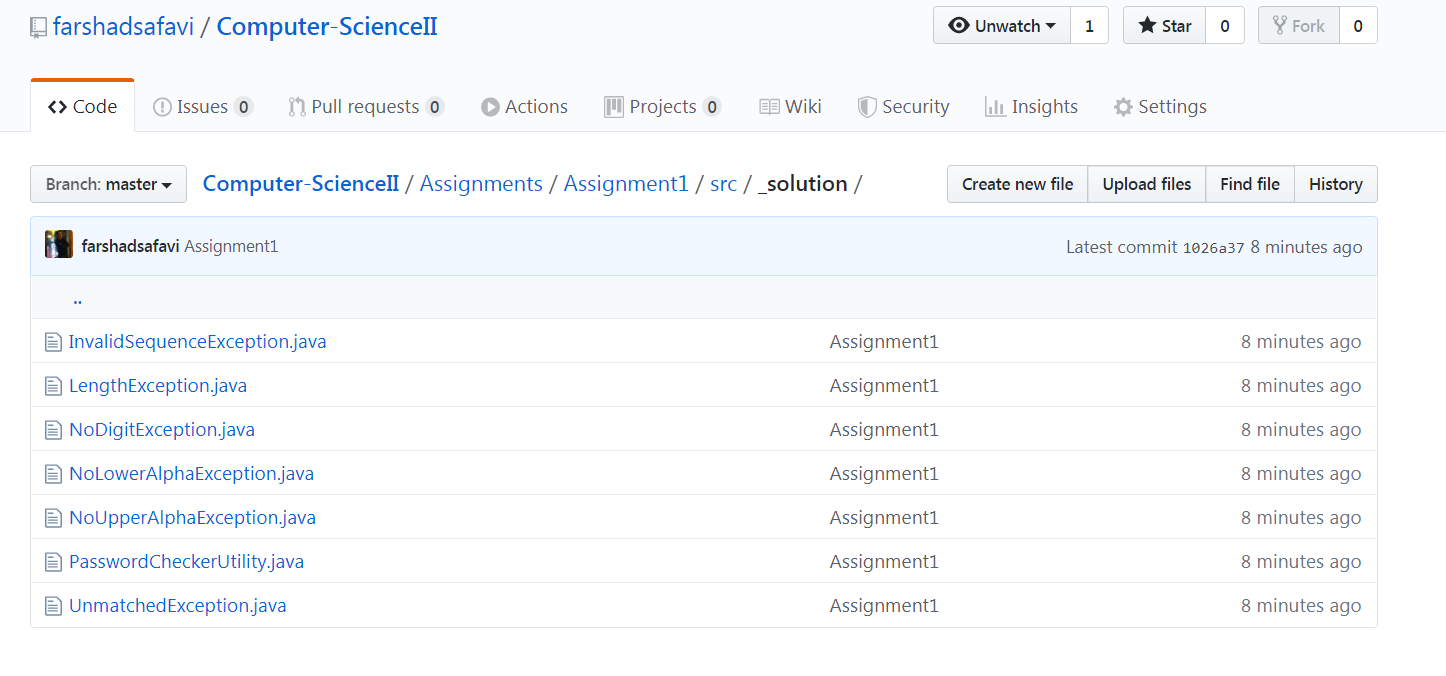
Here is a screenshot of the repository for the Assignment1/src. You can find all files at:

<https://github.com/farshadsafavi/Computer-ScienceII/tree/master/Assignments/Assignment1/src>



Here is a screenshot of the repository for the Assignment1/src/\_solutions. You can find all files at:

<https://github.com/farshadsafavi/Computer-ScienceII/tree/master/Assignments/Assignment1/src/_solution>



Here is the pseudo code for three methods in PasswordCheckerUtility class

**To write** *isValidPassword, isWeakPassword, invalidPasswords* **methods first we need to complete following classes**

1. Create Unmatched Exception Class Extends from Runtime Exception
2. Create a Length Exception Class Extends from Unmatched Exception
3. Create a No Digit Exception Class Extends from Unmatched Exception
4. Create a No Upper Alpha Exception Class Extends from Unmatched Exception
5. Create a No Lower Alpha Exception Class Extends from Unmatched Exception
6. Create an Inavalid Sequence Exception Class Extends from Unmatched Exception

**Pseudo code for isValidPassword:**

Create a function isValidPassword(password)

* If password is less than 6
  + Throw a Length Exception
* if password contains no digit
  + Throw a No digit Exception
* if password contains no uppercase alphabetic character
  + Throw a No Upper Alpha Exception
* if password contains no uppercase alphabetic character
  + Throw a No Lower Alpha Exception
* If password contains more than two of the same character in sequence.
  + Throw an Inavalid Sequence Exception
* Return True

**Pseudo code for isValidPassword:**

Create a function invalidPasswords (list of passwords)

* Declare Array list of InvalidPassowrds
* For loop iterate through all input passwords
  + Try block
    - passing each password to isValidPassword (password)
  + catch block
    - Add invalid password to array list InvalidPassowrds
* Return InvalidPasswords arraylist

**Pseudo code for isWeakPassword:**

Create a function isWeakPassword(password)

* Return passwords length <= 9