Pseudocode for PasswordCheckerUtility:

isValidPassword(String) method:

* Local Variables:
  + passwordArray: String[] – Turns the password string to an array to loop through every item later
  + containsNumber: Boolean – Used to check if the password contains a number
  + containsUpperAlpha: Boolean – Used to check if the password contains upper case letters
  + containsLowerAlpha: Boolean – Used to check if the password contains a lower case letter
  + foundTheSame: Boolean – Used to check if the same character was found in a sequence
  + lastCharacter: char – Used to save the last character when checking the sequence
* Can throw the following exception:
  + PasswordException – The superclass of all the exceptions listed below:
    - LengthException – This occurs when the length of the password is less than 6 characters.
    - NoDigitException – This occurs when the password contains no digits.
    - NoUpperAlphaException – This occurs when there aren’t any uppercase letters in the password
    - NoLowerAlphaException – This occurs when there aren’t any lowercase letters in the password
    - NoSpecialSymbolException – This occurs when there aren’t any special symbols in the password.
    - InvalidSequenceException – This occurs when the password has more than 2 of the same character in a sequence
* Check the length of the password passed into the method by calling the String length method
  + If the length is less than 6:
    - throw a new instance of LengthException.
    - Return false? (Might get an UnreachableCode Exception)
* Check if the password contains a number by:
  + Loop through the passwordArray
    - Use switch to check if the current item is a number
      * If it has a number, change containsNumber to true
      * If it does not, keep it as false
  + Check if the containsNumber is false
    - If it is false:
      * Throw a new instance of NoDigitException
* Check if the password contains an uppercase letter by:
  + Loop through the passwordArray:
    - Use switch to check if the current item is an uppercase letter
      * If it is an uppercase letter, set containsUpperAlpha to true
      * If it does not, keep it to false
  + Check if containsUpperAlpha is false
    - If it is false
      * Throw a new instance of NoUpperAlphaException.
* Check if the password contains a lowercase letter by:
  + Loop through the passwordArray:
    - Use switch to check if the current item is a lowercase letter
      * If it is a lowercase letter, set containsLowerAlpha to true
      * If it does not, keep it to false
  + Check if containsLowerAlpha is false
    - If it is false
      * Throw a new instance of NoLowerAlphaException.
* Check if the password contains a special character by:
  + Create a variable for a Regex pattern to check the following characters: [!@#$%^&\*()-=\_+`~[]\{};:'",<.>/?|](mailto:!@#$%^&*()-=_+`~[]\{};:'",<.>/?|)
  + Create a variable for the matcher to check if the password contains the characters
  + Check if the password contains a symbol with the regex
    - If it is false
      * Throw a new instance of NoSpecialSymbolException.

Check if the password has more than two of the same characters in a sequence by:

* + Loop through the passwordArray
    - Check if the current item is equal to the lastCharacter
      * If it is, add 1 to foundTheSame
      * If it isn’t, set foundTheSame to 1, as the current character would technically be found as the same.
    - set the lastCharacter to the current item
    - Check if foundInSequence is greater than or equal to three
      * If it is:
        + create a new instance of the InvalidSequenceException.
* Return true because to get to this point, all past tests must pass.

isWeakPassword(String) method

* Check if the length of the password passed is greater than or equal to 10 with the String length method
  + If it is:
    - Return true
  + If it isn’t:
    - Return false

getInvalidPasswords(ArrayList<String>) method:

* Create a new ArrayList of type String called invalidPasswords to store the invalid passwords
* Loop through the passed list of passwords
  + Try to do the following:
    - Pass the current password to the isValidPassword method
  + Catch anything under PasswordException:
    - Add the password to invalidPasswords ArrayList in the following format:
      * <password> <Message of the exception>
* Add a blank string to the arraylist or you can get an error if there aren’t any invalid passwords
* Return invalidPasswords

UML Class Diagram for created classes:

A screenshot of a cell phone

Description automatically generated

Test Cases:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Case #** | **Test** | **Expected Result** | **Actual Result** | **Pass?** | **Reflection** |
| Case 1 | Password 1:  E3f3r  Password 2:  L@dG6  Password 3:  Go123 | All passwords should fail due to the length being less than 6 characters | All passwords failed due to the length of the password is less than 6. | Y | Test passed |
| Case 2 | Password 1:  bp.ww@13  Password 2:  qqrg533~.f  Password 3:  [ro34.6r@fb3](mailto:ro34.6r@fb3) | All passwords should fail because there aren’t any uppercase letters | All passwords failed because there aren’t any uppercase letters. | Y | Test passed |
| Case 3 | Password 1:  bP.Ww@13  Password 2:  L@pG6..f$  Password 3:  AAaa@@a44 | All passwords should be valid passwords, but they will be considered a weak password | All passwords passed and were also weak | Y | Test passed |