Programming I

---The Validation Program

EHSAN, KHOSHNAVAZ MOTLAGH ID: 2340457

LI QIAN ID: 2340667

2023

Table of Contents

[Planning and development of the code 2](#_Toc129162202)

[Basic descriptive process of each method 2](#_Toc129162203)

[Individual team assignments 5](#_Toc129162204)

[Tools used during development 5](#_Toc129162205)

[Overview of the project 6](#_Toc129162206)

[Appendix: Daily Scrum Meeting Logs 6](#_Toc129162207)

# Planning and development of the code

The planning and development of the code is conducted from the following aspects and steps:

1. Identify the inputs and functions. The methods will be used to validate 3 functions.
2. Email Validation
3. Username Validation
4. Password Validation
5. The second step is to identify the validation criteria. It involves specifying the rules that the input must follow to be considered valid. For instance, if the character is a letter of the English alphabet or a number, if the charter is an acceptable special character etc.
6. Once the validation criteria have been identified, the next step is to determine the algorithm that the program will use to validate. The sets of if-else statements are used to check if the input meets the criteria. In order not to repeat code, some methods are called in other methods.
7. Develop the code and write methods to return a Boolean value. The result should meet the criteria and indicate whether the input is valid or not.
8. Some private methods implemented to be used in this class to prevent repetition and high readability of the original method.
9. Test, modify code and bug fixed. Testing is done by providing various inputs manually and verifying the outputs. it is important to test it thoroughly to ensure that it strictly follows the setting rules.
10. Documentation. The word document is necessary because the other developers or user can understand how it works and how to use it. This documentation should include information on the basic descriptive process of each method, validation criteria, the validation algorithm and other relevant details.

# Basic descriptive process of each method

1. isAlphaNum()

This method is to check if a character is alphanumeric. It takes a char as its only argument. Conditional statement is used to check if the given character is a letter of the English alphabet or a digit between 0 and 9

1. isSpecialChar()

This method is to check if a character is an acceptable special character. It takes a char, and boolean as its argument. “if” statement is used to add criteria. It returns true if such character is a dash (-) or a period (.), and it also allow for the underscore (\_) to return true if the boolean argument is true.

1. isPrefixChar()

This method is to check if a character is allowed in the prefix. It takes a char as its only argument. “if” statement is used to add criteria. It returns true if a character in the prefix is alphanumeric characters, dashes, periods, or underscores.

1. isDomainChar()

This method is to check if a character is allowed in the domain. It takes a char as its only argument. We call methods of isAlphaNum() and isSpecialChar() to realize this function. It returns true if a character in the domain is alphanumeric characters, dashes, or periods

1. singleAtSign()

This method is to check if a String contain a single at sign (@). It will return false as soon as number of @ is more than 1.

1. fetchBeforeAt()

This method is to get the first part of an email address which is before @. It takes a String (full email address assumingly) as its only argument. The result will return a string containing the portion before the @ symbol.

1. fetchAfterAt()

This method is to get the second part of an email address. It takes a String (full email address) as its only argument. The result will return a string containing the portion after the @ symbol.

1. isPrefix()

This method is to check if the string of first part before @ is a valid email prefix. It takes a string as its only argument. It will return true if:

* there is at least 1 character in the string,
* Only alphanumeric and underscore, period, dash is allowed,
* Underscore, period, dash must be followed by 1 alphanumeric char,
* First character is alphanumeric.

1. isDomain()

This method is to check if the string of second part is a valid email domain. It takes a string as its only argument. the method takes a string as its only argument and it will return true if:

* if there is only one period,
* string in before period
* contains at least 1 character
* only alphanumeric and period and dash
* period and dash must be followed by alphanumeric.
* Starts with only alphanumeric
* String in after period
  + contains at least two characters,
  + contains only English alphabet.

1. isEmail()

This method is to check if a string is a valid email address. It takes a String as its only argument. Firstly, the method “singleAtSign()” is called to check if the string is a valid email address. If email address is in acceptable formats, then the other 2 methods “fetchBeforeAt()” and “fetchAfterAt()” are called to split email address into 2 parts ( prefix and suffix). The third step is to call methods “isPrefix()” and “isDomain()” to verify if prefix and suffix are following the rules. Return true if everything meets conditions.

1. isUsername()

This method is to check if a string is a valid username. It takes a String as its only argument. Conditional statement, for loops and arrays are used to check if the string meets below conditions. Return lowercase valid username if all rules are met.

* Contains at least one alphanumeric charter
* Contains seven or less characters
* Contains only alphanumeric characters, periods, dashes or an exclamation point(!)
* Start with a period or dash
* A period or dash must always be followed by at least one alphanumeric characters.

1. safePassword()

This method is to check if the string is safe to use. . It takes a String as its only argument and returns true if:

* Contains at least one alphanumeric characters
* Contains a minimum 7 characters and maximum 15 characters
* Contain at least one uppercase letter, one lowercase letter, one number, and one period, dash or underscore
* The same character must never be repeated more than twice

1. isAlphabetic()

This method takes a char as only argument and returns true if the char is an English alphabet.

1. onlyOneChar()

This method takes a char and a string as arguments and looks up the char inside the string returns true if the occurrence is only one.

1. splitByChar()

This method takes a char and a string as arguments and returns a string array where the value of first index is the first portion of string split by the char and second index is the second portion. Also returns and empty string in first index when there is no given char in the string.

1. hasCharInString()

This method takes a char and a string as arguments and returns true if the char appears in the string.

1. hasMinRequirements()

This method takes a string as argument and returns true if there are at least one of each bellow:

* uppercase letter
* lowercase letter
* number
* period, dash or underscore

1. hasConsecutiveLetters()

This method takes a string as argument and returns true if there is no repeating letters in the string.

1. isAlphaNum()

This method is an overload method. It takes a string as argument and returns false if there is no alphanumeric character in the string.

# Individual team assignments

|  |  |  |
| --- | --- | --- |
| **Role** | **Name** | **Assignments** |
| Development Team | Ehsan, Khoshnavaz Motlagh | The methods of singleAtSign(), fetchBeforeAt(), fetchAfterAt(), isPrefix(), isDomain(), safePassword() and documentation |
| Scrum Master  /Development Team | Li QIAN | The methods of isAlphaNum(), isSpecialChar(), isPrefixChar(); isDomainChar(); isEmail(); isUsername() and documentation |

## Tools used during development

|  |  |
| --- | --- |
| **Tasks** | **Tools** |
| Develop Java code and debug | Eclipse and IntelliJ IDE for Java |
| File transfer and update | Github/Teams |
| Daily scrum meeting | Teams |

# Overview of the project

This project is about to write a series of methods to validate email addresses, username and password. All methods are in a class named Validator. The methods will check if a given string meet all conditions. For example, if a char is alphanumeric or a special character, if a char is allowed in the prefix or domain of an email address, and if only a single "@" symbol in the email address. The project also involves methods to check if the prefix and domain of an email address are valid according to certain criteria. There are total 19 methods (12 public methods and 7 private methods) in our project. All methods are tested to make sure they meet all conditions.

Daily Scrum meetings were hold everyday to review the progress of the project and share the experience between teams when developing code. Each team member works hard towards the completion of the project.

We also faced problems during the development of the project. Some conditions or explanation of the functions are not very clear or easy to understand. Different people may have different understanding on some conditions. Finally, the team worked together and finished it based on our best understanding or assumption.

# Appendix: Daily Scrum Meeting Logs

|  |  |
| --- | --- |
| **Date:** | March 4, 2023 Friday 9:15am |
| **Attendees:** | Ehsan, Khoshnavaz Motlagh, Li QIAN |
| **Agenda:** | * Review the requirement of the project * Next step: Li will develop the first 4 methods   Ehsan will develop the other 5 methods  Both two will process the last 3 methods  Impediments: None |

|  |  |
| --- | --- |
| **Date:** | March 6, 2023 Monday 14:00 pm |
| **Attendees:** | Ehsan, Khoshnavaz Motlagh, Li QIAN |
| **Agenda:** | * Review the methods we finished * Discuss the combination of total methods and finish the last 3 methods. * Next step: Ehsan modify the codes   Li starts to prepare word documentation  Impediments: None |

|  |  |
| --- | --- |
| **Date:** | March 7, 2023 Tuesday 12:00 pm |
| **Attendees:** | Ehsan, Khoshnavaz Motlagh, Li QIAN |
| **Agenda:** | * Review the code we finished * Improve our document * Next step: finish Validator.java file and word documentation together   Impediments: None |

|  |  |
| --- | --- |
| **Date:** | Mar 8, 2023 Wednesday 10:00 am |
| **Attendees:** | Ehsan, Khoshnavaz Motlagh, Li QIAN |
| **Agenda:** | * Test the code again for more examples * Review the total documents will be submitted * Submitted the project before 14:30pm |