The code base for the assignment has files representing 3 classes – PasswordCriteria, ProgramTest and TestRunner.

The assignment has two tasks:

1. Write the specification for the class PasswordCriteria – Add pre-conditions and post-conditions for the public methods and the Invariant for the class.
2. Write methods in the ProgramTest class to test PasswordCriteria class based on the specification.

PasswordCriteria represents an abstraction of criteria for acceptance of passwords. It captures certain rules and length requirements and provides means to check the validity of a password based on the criteria. The password may contain only letters (both cases) and numbers. Special characters are not allowed.

The class allows a particular range of password length bounded by the attributes (minLength and maxLength). It also provides four criteria for the characters in the password –

- *hasLetters* – Password contains letters in the alphabet (a-z). This attribute alone only implies lower case letters.
- *hasMixedCase* – Password contains both lower and upper case letters. If this is false and hasLetters is true, then the class only allows password with lower case letters. If both *hasLetters* and *hasMixedCase* are true, then it implies that the password must have both cases.
- *hasNumbers* – Password contains digits 0-9.
- *hasAllDifferent* – Password has all unique characters.

The object is created with an initial value of true for *hasLetters* and the rest are initialized to false. There are several public methods of the class which are used to set/get various attributes and test the validity of the password. The details are provided in the comments of these methods.

The contracts are required only for the **public methods**. Some of the private methods of the class can be used to write the contracts.

The concept is similar to Norton password generator - https://identitysafe.norton.com/password-generator

The specification is based on the requirements below:

1. Password must be of size greater than 3.
2. The minimum length of the password must not exceed the maximum length.
3. *hasMixedCase* can be enabled only if *hasLetters* is enabled.
4. Both *hasLetters* and *hasNumbers* must **not** be false at the same time. At least one of them must be true.

In addition to the contracts implied by the above requirements, there are contracts which can be derived indirectly from the requirements e.g. length constrains based on the rules – *Does the maximum length have any constraints based on the rules?*

There are certain contracts based on design restriction e.g. the password argument must not be null or empty.

The task should address the kinds of contracts explained above. For simplicity we can make the assumption that none of the methods make any irrelevant changes to member attributes. E.g. in the method getMaxLength() – we can assume that the method does not alter the values of the attribute minLength, maxLength or any other member attribute. Thus, the contracts need not consider these scenarios. Same applies for other similar methods.

```
/*
* Getter for max length
*/
public Integer getMaxLength() {
	maxLength = 0;
	minLength = 1;
	return maxLength;
}
```

Contracts are already provided partly for some methods as an example. Please add further changes as required.

The tests are in ProgramTest.java file. Two test methods are provided as an example, add more tests for the contracts. If changes are made in the code to test the post condition, state the change in the comments in the corresponding test method. Please do not change the source code unless required for testing. Execution of the test is done using the same command as practices in the lab sessions.

In Windows –

```
>testrun.bat “<path to JDK>”
```

```
e.g. testrun.bat "C:\Program Files\Java\jdk1.7.0_79"
```

in Linux –

```
$sh ./testrun.sh
```

The output of the test will also display the **stack trace** to help identify the function where the contract failed. The details of the trace may be ignored.

The assignment will be graded based on the number of contracts addressed and tests performed to verify them. The assignment can be done individually or in group. Group work is encouraged and group size must not be more than 4 members.

Please ensure that source files can be compiled and execute with the unmodified TestRunner.java and the build scripts. Upload the two source files (PasswordCriteria and ProgramTest) to Moodle latest **by 9th Dec 23:55.**