Requirements

Java Runtime Environment 1.8

Compilation instructions

Step 1: Unzip the code and change directory to BinaTech_PasswordValidator

Step 2: Compile the code. Enter the following command –

```
javac com/gaurav/rules_interface/IRule.java
com/gaurav/rules_validator_interface/IPasswordValidator.java
com/gaurav/password_validator/PasswordValidator.java
com/gaurav/advanced_validator/AdvancedPasswordValidator.java
```

Step 3: Run the program using following command –

```
java com.gaurav.password_validator.PasswordValidator [arg 1 Password] [arg 2 Rules]
```

```
e.g.:
For Password Validator -
java com.gaurav.password_validator.PasswordValidator Abcdef1 1,2,3

For Advanced Password Validator –
java com.gaurav.advanced_validator.AdvancedPasswordValidator  aaaaweroch 1,3,9
```

Note on future code changes and extension

Other programmers can use Delegate Design Pattern to add and define more rules for validating passwords. IPasswordValidator interface can be implemented and PasswordValidator instance can be used as the delegate object. This way the task of maintaining password rules and password validation can be delegated to PasswordValidator object. IPasswordValidator provides addRule() API for adding more rules to PasswordValidator library. Please check AdvancedPasswordValidator.java to see the code.

Rules are defined using lambda expressions. Rule definitions implement the functional interface IRule and must return a boolean value.

PasswordValidator class has been implemented as final to avoid inheritance which may result in tightly coupled and hard to maintain code. Also PasswordValidator class is a specialized class and not designed for inheritance. Instead programmers can use composition to achieve code reuse.