**Implement the following projects which focus on different aspects of Java programming, including:**

Object-Oriented Programming principles

Exception handling

File operations

Data structures

Input validation

User interface design

6. Password Validator

Create a program that validates passwords based on specific criteria using object oriented programming in Java.

Key points:

- Check length (minimum 8 characters)

- Require numbers and special characters

- Use regular expressions

import java.util.ArrayList;

import java.util.List;

import java.util.regex.Pattern;

class PasswordValidationException extends Exception {

public PasswordValidationException(String message) {

super(message);

}

}

class ValidationRule {

private final String description;

private final Pattern pattern;

public ValidationRule(String description, String regex) {

this.description = description;

this.pattern = Pattern.compile(regex);

}

public boolean validate(String password) {

return pattern.matcher(password).matches();

}

public String getDescription() {

return description;

}

}

public class PasswordValidator {

private final List<ValidationRule> rules;

private static final int MIN\_LENGTH = 8;

public PasswordValidator() {

rules = new ArrayList<>();

initializeRules();

}

private void initializeRules() {

// Rule for minimum length

rules.add(new ValidationRule(

"Password must be at least " + MIN\_LENGTH + " characters long",

".{" + MIN\_LENGTH + ",}"

));

// Rule for at least one number

rules.add(new ValidationRule(

"Password must contain at least one number",

".\*[0-9].\*"

));

// Rule for at least one lowercase letter

rules.add(new ValidationRule(

"Password must contain at least one lowercase letter",

".\*[a-z].\*"

));

// Rule for at least one uppercase letter

rules.add(new ValidationRule(

"Password must contain at least one uppercase letter",

".\*[A-Z].\*"

));

// Rule for at least one special character

rules.add(new ValidationRule(

"Password must contain at least one special character",

".\*[!@#$%^&\*()\\[\\]{}\\-\_+=~`|:;\"'<>,.?/].\*"

));

}

public void validatePassword(String password) throws PasswordValidationException {

List<String> violations = new ArrayList<>();

for (ValidationRule rule : rules) {

if (!rule.validate(password)) {

violations.add(rule.getDescription());

}

}

if (!violations.isEmpty()) {

throw new PasswordValidationException(

"Password validation failed:\n- " +

String.join("\n- ", violations)

);

}

}

// Example usage with main method

public static void main(String[] args) {

PasswordValidator validator = new PasswordValidator();

String[] testPasswords = {

"weak",

"NoSpecialChar1",

"NoNumber@abc",

"Valid@Password123",

};

for (String password : testPasswords) {

try {

System.out.println("\nTesting password: " + password);

validator.validatePassword(password);

System.out.println("Password is valid!");

} catch (PasswordValidationException e) {

System.out.println(e.getMessage());

}

}

}

}

o/p

Testing password: weak

Password validation failed:

- Password must be at least 8 characters long

- Password must contain at least one number

- Password must contain at least one uppercase letter

- Password must contain at least one special character

Testing password: NoSpecialChar1

Password validation failed:

- Password must contain at least one special character

Testing password: NoNumber@abc

Password validation failed:

- Password must contain at least one number

Testing password: Valid@Password123

Password is valid!