Password 🔍 Validator :

Check if a password meets certain security criteria (e.g., length, special characters).

This project validates a user's password based on defined security criteria, such as minimum length, the inclusion of uppercase letters, lowercase letters, numbers, and special characters. It informs the user whether the password is valid or needs improvement.

Input: A password string.
Output: Whether the password is valid based on criteria (e.g., length, special characters).

Example:

- Input: "Password123!"
- Output: "Valid password"
- Input: "pass"
- Output: "Invalid password"

Solution 1: Password Validator using Basic String Methods

Code:

```
import java.util.Scanner;

public class PasswordValidatorUsingStringMethods {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Taking password input from the user
        System.out.print("Enter a password: ");
        String password = scanner.nextLine();

        // Define password validation criteria
        boolean isValid = validatePassword(password);

        // Output result based on validation
        if (isValid) {
            System.out.println("Valid password");
        } else {
            System.out.println("Invalid password");
        }
```

```
            scanner.close(); // Close the scanner
        }

        // Method to validate the password based on multiple criteria
        public static boolean validatePassword(String password) {
            // Check for minimum length (at least 8 characters)
            if (password.length() < 8) {
                return false;
            }

            boolean hasUppercase = false;
            boolean hasLowercase = false;
            boolean hasDigit = false;
            boolean hasSpecialChar = false;

            // Check each character of the password for uppercase, lowercase, d
            for (int i = 0; i < password.length(); i++) {
                char ch = password.charAt(i);
                if (Character.isUpperCase(ch)) {
                    hasUppercase = true;
                } else if (Character.isLowerCase(ch)) {
                    hasLowercase = true;
                } else if (Character.isDigit(ch)) {
                    hasDigit = true;
                } else if ("!@#$%^&*()-+".contains(Character.toString(ch))) {
                    hasSpecialChar = true;
                }
            }

            // Return true if all criteria are met
            return hasUppercase && hasLowercase && hasDigit && hasSpecialChar;
        }
    }
```

Output:

```
Enter a password:  Go3(&#Pa12
Valid password
```

```
Enter a password:  myPassword
```

```
Invalid password
```

Explanation :

- **Input:**The user enters a password.
- **Validation Criteria:**
  - At least 8 characters long.
  - Contains uppercase, lowercase, digits, and special characters.
- **Character Check:** Loops through the password to check for each required type of character.
- **Validation Result:** Returns whether the password meets the criteria.
- **Output:** Displays whether the password is valid or invalid.

Solution 2: Password Validator using Regular Expressions (Regex)

Code:

Copy

```java
import java.util.Scanner;
import java.util.regex.Pattern;

public class PasswordValidatorUsingRegex {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Taking password input from the user
        System.out.print("Enter a password: ");
        String password = scanner.nextLine();

        // Define regex pattern for password validation
        boolean isValid = validatePassword(password);

        // Output result based on validation
        if (isValid) {
            System.out.println("Valid password");
        } else {
            System.out.println("Invalid password");
        }

        scanner.close(); // Close the scanner
    }
```

```
        // Method to validate password using regex
        public static boolean validatePassword(String password) {
            // Regular expression pattern to check:
            // - At least 8 characters long
            // - At least one uppercase, one lowercase, one digit, and one spec
            String regex = "^(?=.*[A-Z])(?=.*[a-z])(?=.*\\d)(?=.*[!@#$%^&*()-+]

            // Validate the password using the regex pattern
            return Pattern.matches(regex, password);
        }
    }
```

Output:

```
Enter a password:  Pass123
Invalid password
```

```
Enter a password:  Ohu$12sWe%
Valid password
```

Explanation: