**Validación de contraseñas en JAVA**  
  
import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class PasswordValidator {

public static void main(String[] args) {

String password = "MiPassword#2023";

if (isValidPassword(password)) {

System.out.println("La contraseña es segura");

} else {

System.out.println("La contraseña no cumple con los requisitos de seguridad");

}

}

public static boolean isValidPassword(String password) {

// Al menos una mayúscula

Pattern uppercasePattern = Pattern.compile("[A-Z]");

Matcher uppercaseMatcher = uppercasePattern.matcher(password);

boolean hasUppercase = uppercaseMatcher.find();

// Al menos un símbolo

Pattern symbolPattern = Pattern.compile("[^\\w\\s]");

Matcher symbolMatcher = symbolPattern.matcher(password);

boolean hasSymbol = symbolMatcher.find();

// Al menos un número

Pattern numberPattern = Pattern.compile("\\d");

Matcher numberMatcher = numberPattern.matcher(password);

boolean hasNumber = numberMatcher.find();

// Longitud mayor a 8 caracteres

boolean hasLength = password.length() > 8;

return hasUppercase && hasSymbol && hasNumber && hasLength;

}

}

En este ejemplo, se define un método **isValidPassword** que toma como argumento una cadena **password** y devuelve **true** si cumple con los requisitos de seguridad mencionados anteriormente y **false** en caso contrario.

Para cada uno de los requisitos, se utiliza una expresión regular para buscar si la cadena contiene el tipo de carácter requerido. Luego, se utiliza el método **find** del objeto **Matcher** para verificar si se encontró alguna coincidencia.

Finalmente, se devuelven los resultados combinados de todas las verificaciones utilizando operadores lógicos **&&**.

**Forma Alternativa**

import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class PasswordValidator {

public static void main(String[] args) {

String password = "MiPassword#2023";

if (isValidPassword(password)) {

System.out.println("La contraseña es segura");

} else {

System.out.println("La contraseña no cumple con los requisitos de seguridad");

}

}

public static boolean isValidPassword(String password) {

Pattern uppercasePattern = Pattern.compile("[A-Z]");

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

Pattern numberPattern = Pattern.compile("\\d");

Matcher uppercaseMatcher = uppercasePattern.matcher(password);

Matcher symbolMatcher = symbolPattern.matcher(password);

Matcher numberMatcher = numberPattern.matcher(password);

boolean hasUppercase = uppercaseMatcher.find();

boolean hasSymbol = symbolMatcher.find();

boolean hasNumber = numberMatcher.find();

boolean hasLength = password.length() > 8;

return hasUppercase && hasSymbol && hasNumber && hasLength;

}

}

En este ejemplo, la validación de la contraseña se realiza en el método isValidPassword. Primero, se crean tres patrones de expresión regular para buscar una mayúscula, un símbolo y un número en la contraseña.

Luego, se crean tres objetos Matcher que corresponden a cada patrón y se utiliza el método find para buscar si hay coincidencias en la contraseña.

Finalmente, se verifica que la contraseña tenga una mayúscula, un símbolo, un número y una longitud mayor a 8 caracteres utilizando variables booleanas y se devuelve el resultado de la validación utilizando el operador lógico &&.