CodInClub

Powered By BridgeLabz

Here are **practice problems** for **Java Regular Expressions (Regex)**. These problems will help you **validate, extract, replace, and manipulate strings** using regex in Java.

📝 Basic Regex Problems

1️⃣ Validate a Username

- A valid username:
  - Can only contain **letters (a-z, A-Z), numbers (0-9), and underscores (_)**
  - Must start with a letter
  - Must be **between 5 to 15 characters long**

🔹 **Example Inputs & Outputs**

✅ `"user_123"` → **Valid**

❌ `"123user"` → **Invalid** (starts with a number)

❌ `"us"` → **Invalid** (too short)

```java
import java.util.Scanner;

public class ValidateUsername {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a username: ");
        String username = sc.nextLine();
        if (username.matches("^[a-zA-Z][a-zA-Z0-9_]{4,14}$")) {
            System.out.println("Valid");
        } else {
            System.out.println("Invalid");
        }
    }
}
```

2️⃣ Validate a License Plate Number

- License plate format: **Starts with two uppercase letters, followed by four digits.**
- Example: `"AB1234"` is **valid**, but `"A12345"` is **invalid**.

```java
import java.util.Scanner;

public class ValidateLicensePlate {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter license plate: ");
        String plate = sc.nextLine();
        if (plate.matches("^[A-Z]{2}\\d{4}$")) {
            System.out.println("Valid");
        } else {
            System.out.println("Invalid");
        }
    }
}
```

3️⃣ Validate a Hex Color Code

- A valid **hex color**:
    - Starts with a **#**
    - Followed by **6 hexadecimal characters** (`0-9`, `A-F`, `a-f`).

🔹 **Example Inputs & Outputs**

✅ `"#FFA500"` → **Valid**

✅ `"#ff4500"` → **Valid**

❌ `"#123"` → **Invalid** (too short)

```java
import java.util.Scanner;

public class ValidateHexColor {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter hex color code: ");
        String color = sc.nextLine();
        if (color.matches("^#[0-9a-fA-F]{6}$")) {
            System.out.println("Valid");
        } else {
            System.out.println("Invalid");
        }
    }
}
```

🔍 Extraction Problems

4️⃣ Extract All Email Addresses from a Text

🔹 **Example Text:**

`"Contact us at support@example.com and info@company.org"`

🔹 **Expected Output:**

```
support@example.com

info@company.org
```

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

```java
public class ExtractEmails {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the text:");
        String input = sc.nextLine();
        Matcher matcher =
Pattern.compile("[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}").matcher(
input);
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

---

5 Extract All Capitalized Words from a Sentence

🔹 **Example Text:**

`"The Eiffel Tower is in Paris and the Statue of Liberty is in New York."`

🔹 **Expected Output:**

`Eiffel`, `Tower`, `Paris`, `Statue`, `Liberty`, `New`, `York`

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

public class ExtractCapitalizedWords {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the sentence:");
        String input = sc.nextLine();
        Matcher matcher =
Pattern.compile("\\b[A-Z][a-z]*\\b").matcher(input);
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

---

6 Extract Dates in dd/mm/yyyy Format

- **Example Text:**

"The events are scheduled for 12/05/2023, 15/08/2024, and 29/02/2020."

- **Expected Output:**

12/05/2023, 15/08/2024, 29/02/2020

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

public class ExtractDates {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter text with dates:");
        String input = sc.nextLine();
        Matcher matcher =
Pattern.compile("\\b\\d{2}/\\d{2}/\\d{4}\\b").matcher(input);
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

7️⃣ Extract Links from a Web Page

- 🔹 **Example Text:**

`"Visit https://www.google.com and http://example.org for more info."`

- 🔹 **Expected Output:**

`https://www.google.com`, `http://example.org`

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

public class ExtractLinks {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter text with URLs:");
        String input = sc.nextLine();
        Matcher matcher = Pattern.compile("https?://\\S+").matcher(input);
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

🔄 Replace and Modify Strings

8️⃣ Replace Multiple Spaces with a Single Space

- 🔹 **Example Input:**

`"This is an example with multiple spaces."`

🔹 **Expected Output:**

```
"This is an example with multiple spaces."
```

```java
import java.util.Scanner;

public class ReplaceMultipleSpaces {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter text with multiple spaces:");
        String input = sc.nextLine();
        String result = input.replaceAll("\\s{2,}", " ");
        System.out.println(result);
    }
}
```

---

9 Censor Bad Words in a Sentence

- Given a **list of bad words**, replace them with `****`.

🔹 **Example Input:**

```
"This is a damn bad example with some stupid words."
```

🔹 **Expected Output:**

```
"This is a **** bad example with some **** words."
```

```java
import java.util.Scanner;

public class CensorBadWords {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String[] badWords = {"damn", "stupid"};
        System.out.println("Enter sentence:");
        String input = sc.nextLine();
        for (int i = 0; i < badWords.length; i++) {
            input = input.replaceAll("(?i)\\b" + badWords[i] + "\\b",
"****");
        }
        System.out.println(input);
    }
}
```

🔥 Advanced Problems

🔟 Validate an IP Address

- A valid **IPv4 address** consists of **four groups of numbers (0-255) separated by dots**.

```
import java.util.Scanner;

public class ValidateIPAddress {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter IPv4 address: ");
        String ip = sc.nextLine();
        String pattern = "^((25[0-5]|2[0-4]\\d|1\\d{2}|[1-9]?\\d)\\.){3}"
                       + "(25[0-5]|2[0-4]\\d|1\\d{2}|[1-9]?\\d)$";
        if (ip.matches(pattern)) {
            System.out.println("Valid");
        } else {
            System.out.println("Invalid");
        }
    }
}
```

1️⃣1️⃣ Validate a Credit Card Number (Visa, MasterCard, etc.)

- A **Visa** card number starts with 4 and has **16 digits**.
- A **MasterCard** starts with 5 and has **16 digits**.

```java
import java.util.Scanner;

public class ValidateCreditCard {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter credit card number: ");
        String card = sc.nextLine();
        if (card.matches("^4\\d{15}$")) {
            System.out.println("Valid Visa Card");
        } else if (card.matches("^5\\d{15}$")) {
            System.out.println("Valid MasterCard");
        } else {
            System.out.println("Invalid Card");
        }
    }
}
```

1️⃣2️⃣ Extract Programming Language Names from a Text

🔹 **Example Text:**

`"I love Java, Python, and JavaScript, but I haven't tried Go yet."`

🔹 **Expected Output:**

`Java`, `Python`, `JavaScript`, `Go`

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

public class ExtractLanguages {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter a sentence:");
        String input = sc.nextLine();
        Matcher matcher =
Pattern.compile("\\b(Java|Python|JavaScript|Go)\\b",
Pattern.CASE_INSENSITIVE).matcher(input);
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

1️⃣3️⃣ Extract Currency Values from a Text

- **Example Text:**

`"The price is $45.99, and the discount is 10.50."`

- **Expected Output:**

`$45.99`, `10.50`

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

public class ExtractCurrencyValues {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter text with currency values:");
        String input = sc.nextLine();
        Matcher matcher =
Pattern.compile("\\$?\\d+\\.\\d{2}").matcher(input);
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
    }
}
```

1️⃣4️⃣ Find Repeating Words in a Sentence

🔹 **Example Input:**

```
"This is is a repeated repeated word test."
```

🔹 **Expected Output:**

```
is, repeated
```

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

public class FindRepeatingWords {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter a sentence:");
        String input = sc.nextLine();
        Matcher matcher = Pattern.compile("\\b(\\w+)\\b\\s+\\1\\b",
Pattern.CASE_INSENSITIVE).matcher(input);
        while (matcher.find()) {
            System.out.println(matcher.group(1));
        }
    }
}
```

1️⃣5️⃣ Validate a Social Security Number (SSN)

🔹 **Example Input:**

`"My SSN is 123-45-6789."`

🔹 **Expected Output:**

✅ `"123-45-6789"` is **valid**

❌ `"123456789"` is **invalid**

```java
import java.util.Scanner;

public class ValidateSSN {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter SSN: ");
        String ssn = sc.nextLine();
        if (ssn.matches("^\\d{3}-\\d{2}-\\d{4}$")) {
            System.out.println("Valid SSN");
        } else {
            System.out.println("Invalid SSN");
        }
    }
}
```

Here are **practice problems** for **Java Regular Expressions (Regex)**. These problems will help you **validate, extract, replace, and manipulate strings** using regex in Java.

🔍 Extraction Problems

4️⃣ Extract All Email Addresses from a Text

🔹 **Example Text:**

`"Contact us at support@example.com and info@company.org"`

🔹 **Expected Output:**

`support@example.com`

`info@company.org`

5️⃣ Extract All Capitalized Words from a Sentence

🔹 **Example Text:**

`"The Eiffel Tower is in Paris and the Statue of Liberty is in New York."`

🔹 **Expected Output:**

`Eiffel`, `Tower`, `Paris`, `Statue`, `Liberty`, `New`, `York`

6️⃣ Extract Dates in `dd/mm/yyyy` Format

🔹 **Example Text:**

`"The events are scheduled for 12/05/2023, 15/08/2024, and 29/02/2020."`

🔹 **Expected Output:**

`12/05/2023`, `15/08/2024`, `29/02/2020`

7️⃣ Extract Links from a Web Page

🔹 **Example Text:**

`"Visit https://www.google.com and http://example.org for more info."`

🔹 **Expected Output:**

`https://www.google.com`, `http://example.org`

🔄 Replace and Modify Strings

8️⃣ Replace Multiple Spaces with a Single Space

🔹 **Example Input:**

`"This is an example with multiple spaces."`

🔹 **Expected Output:**

`"This is an example with multiple spaces."`

9️⃣ Censor Bad Words in a Sentence

- Given a **list of bad words**, replace them with `****`.

🔹 **Example Input:**

`"This is a damn bad example with some stupid words."`

🔹 **Expected Output:**

`"This is a **** bad example with some **** words."`

🔥 Advanced Problems

🔟 Validate an IP Address

- A valid **IPv4 address** consists of **four groups of numbers (0-255) separated by dots**.

1️⃣1️⃣ Validate a Credit Card Number (Visa, MasterCard, etc.)

- A **Visa** card number starts with `4` and has **16 digits**.
- A **MasterCard** starts with `5` and has **16 digits**.

1️⃣2️⃣ Extract Programming Language Names from a Text

🔹 **Example Text:**

`"I love Java, Python, and JavaScript, but I haven't tried Go yet."`

🔹 **Expected Output:**

`Java`, `Python`, `JavaScript`, `Go`

1️⃣3️⃣ Extract Currency Values from a Text

🔹 **Example Text:**

`"The price is $45.99, and the discount is 10.50."`

🔹 **Expected Output:**

`$45.99`, `10.50`

1️⃣4️⃣ Find Repeating Words in a Sentence

🔹 **Example Input:**

`"This is is a repeated repeated word test."`

🔹 **Expected Output:**

`is`, `repeated`

1️⃣5️⃣ Validate a Social Security Number (SSN)

🔹 **Example Input:**

`"My SSN is 123-45-6789."`

🔹 **Expected Output:**

✅ `"123-45-6789"` is **valid**

❌ `"123456789"` is **invalid**

- A valid username:
  - Can only contain **letters (a-z, A-Z), numbers (0-9), and underscores (_)**
  - Must start with a letter
  - Must be **between 5 to 15 characters long**

🔹 **Example Inputs & Outputs**

✅ `"user_123"` → **Valid**

❌ `"123user"` → **Invalid** (starts with a number)

❌ `"us"` → **Invalid** (too short)

2️⃣ Validate a License Plate Number

- License plate format: **Starts with two uppercase letters, followed by four digits.**
- Example: `"AB1234"` is **valid**, but `"A12345"` is **invalid**.

3️⃣ Validate a Hex Color Code

- A valid **hex color**:
  - Starts with a **#**

  - Followed by **6 hexadecimal characters** (`0-9`, `A-F`, `a-f`).

🔹 **Example Inputs & Outputs**

✅ `"#FFA500"` → **Valid**

✅ `"#ff4500"` → **Valid**

❌ `"#123"` → **Invalid** (too short)

🔍 Extraction Problems

4️⃣ Extract All Email Addresses from a Text

🔹 **Example Text:**

`"Contact us at support@example.com and info@company.org"`

🔹 **Expected Output:**

`support@example.com`

`info@company.org`

5️⃣ Extract All Capitalized Words from a Sentence

🔹 **Example Text:**

`"The Eiffel Tower is in Paris and the Statue of Liberty is in New York."`

🔹 **Expected Output:**

`Eiffel`, `Tower`, `Paris`, `Statue`, `Liberty`, `New`, `York`

6️⃣ Extract Dates in `dd/mm/yyyy` Format

🔹 **Example Text:**

`"The events are scheduled for 12/05/2023, 15/08/2024, and 29/02/2020."`

🔹 **Expected Output:**

`12/05/2023`, `15/08/2024`, `29/02/2020`

7️⃣ Extract Links from a Web Page

🔹 **Example Text:**

`"Visit https://www.google.com and http://example.org for more info."`

🔹 **Expected Output:**

`https://www.google.com`, `http://example.org`

🔄 Replace and Modify Strings

8️⃣ Replace Multiple Spaces with a Single Space

🔹 **Example Input:**

`"This is an example with multiple spaces."`

🔹 **Expected Output:**

`"This is an example with multiple spaces."`

9️⃣ Censor Bad Words in a Sentence

- Given a **list of bad words**, replace them with `****`.

🔹 **Example Input:**

`"This is a damn bad example with some stupid words."`

🔹 **Expected Output:**

`"This is a **** bad example with some **** words."`

🔥 Advanced Problems

🔟 Validate an IP Address

- A valid **IPv4 address** consists of **four groups of numbers (0-255) separated by dots**.

1️⃣1️⃣ Validate a Credit Card Number (Visa, MasterCard, etc.)

- A **Visa** card number starts with `4` and has **16 digits**.
- A **MasterCard** starts with `5` and has **16 digits**.

1️⃣2️⃣ Extract Programming Language Names from a Text

🔹 **Example Text:**

`"I love Java, Python, and JavaScript, but I haven't tried Go yet."`

🔹 **Expected Output:**

`Java`, `Python`, `JavaScript`, `Go`

1️⃣3️⃣ Extract Currency Values from a Text

🔹 **Example Text:**

`"The price is $45.99, and the discount is 10.50."`

🔹 **Expected Output:**

`$45.99`, `10.50`

1️⃣4️⃣ Find Repeating Words in a Sentence

🔹 **Example Input:**

`"This is is a repeated repeated word test."`

🔹 **Expected Output:**

`is`, `repeated`

1️⃣5️⃣ Validate a Social Security Number (SSN)

🔹 **Example Input:**

`"My SSN is 123-45-6789."`

🔹 **Expected Output:**

✅ `"123-45-6789"` is **valid**

❌ `"123456789"` is **invalid**
