### **Basic Symbols in Regex**

Here are some of the most common symbols in regular expressions:

#### **. (Dot)**

* The dot . matches **any single character** except newlines.

**Example:**

* a.b
  + This will match:
    - acb, a1b, a b, a@b
  + It won’t match:
    - ab (because there's no character between a and b).

#### **\d (Digit)**

* \d matches any **digit** (0-9).

**Example:**

* \d
  + This will match:
    - 1, 5, 9, 0
  + It won’t match:
    - a, @, or any non-digit character.

#### **\w (Word Character)**

* \w matches any **alphanumeric character** (letters and numbers) and the underscore \_.

**Example:**

* \w
  + This will match:
    - a, 1, \_, z, A
  + It won’t match:
    - @, #, spaces, etc.

#### **\s (Whitespace)**

* \s matches any **whitespace** character (like spaces, tabs, and newlines).

**Example:**

* \s
  + This will match:
    - A space between words, or a tab character.
  + It won’t match:
    - Letters or numbers.

#### **^ (Caret)**

* The caret ^ matches the **start of a string**.

**Example:**

* ^a
  + This will match:
    - apple, a123 (anything starting with a).
  + It won’t match:
    - banana (doesn't start with a).

#### **$ (Dollar Sign)**

* The dollar sign $ matches the **end of a string**.

**Example:**

* a$
  + This will match:
    - ba (because it ends with a).
  + It won’t match:
    - abc (because abc doesn’t end with a).

### **3. Quantifiers (Repetition)**

#### **\* (Asterisk)**

* \* matches **zero or more** occurrences of the preceding character.

**Example:**

* a\*
  + This will match:
    - ``, a, aa, aaa, ... (zero or more as).
  + It won’t match:
    - b (because there's no a).

#### **+ (Plus)**

* + matches **one or more** occurrences of the preceding character.

**Example:**

* a+
  + This will match:
    - a, aa, aaa, ...
  + It won’t match:
    - `` (empty string, because at least one a is needed).

#### **? (Question Mark)**

* ? makes the preceding character **optional**. It matches zero or one occurrence.

**Example:**

* colou?r
  + This will match:
    - color, colour
  + It won’t match:
    - colouur (extra u is not allowed).

### **4. Character Classes and Groups**

#### **[] (Character Set)**

* [] defines a **set of characters** to match. It matches any single character inside the brackets.

**Example:**

* [aeiou]
  + This will match:
    - Any vowel: a, e, i, o, u.
  + It won’t match:
    - b, c, etc.

#### **() (Grouping)**

* () is used for **grouping**. It allows you to apply quantifiers to a set of characters as a unit.

**Example:**

* (abc)+
  + This will match:
    - abc, abcabc, abcabcabc, ...
  + It won’t match:
    - ab, ac (the whole group abc is required).

Examples

1.Match a 3-digit number:

^\d{3}$

2. Match a name with alphabets and spaces:

^[a-zA-Z\s]+$

3. Match a CNIC number:

^\d{5}-\d{7}-\d{1}$

#### **Example Validating a Phone Number (XXX-XXX-XXXX)**

To match a phone number like 123-456-7890

^\d{3}-\d{3}-\d{4}$

**Matching an IP Address (IPv4)**

**Valid: 192.168.0.1, 255.255.255.255**

**^(\d{1,3}\.){3}\d{1,3}$**

**Explanation:**

* **^ – start of the string.**
* **(\d{1,3}\.) – match 1 to 3 digits followed by a dot (.). This part is repeated 3 times.**
* **{3} – repeat the previous pattern exactly 3 times.**
* **\d{1,3} – match 1 to 3 digits (the last group).**
* **$ – end of the string.**

**9. Matching Hexadecimal Color Codes**

**To match a hexadecimal color code like #A3C1F7:**

### **Matching a Decimal Number (Positive or Negative)**

**To match any decimal number, including positive and negative numbers:**

**12.34, -5.67, +0.99**

**^[+-]?\d+\.\d+$**

Practice problem.

Match a 4-digit year (2023).

Match a credit card number (1234-5678-9876-5432).

Match a date (31/12/2023).

Matching Time (HH:MM:SS) (12:45:30)

—------------------------------------------------------------------------------------------------------------

### **Match a Date in MM-DD-YYYY Format**

#### **Requirements:**

We need to validate a date in the MM-DD-YYYY format:

* The month should be between 01 and 12.
* The day should be between 01 and 31.
* The year should be exactly 4 digits.

### **Solution:**

#### **Regular Expression:**

^(0[1-9]|1[0-2])-(0[1-9]|[12][0-9]|3[01])-\d{4}$

#### **Explanation:**

* ^ – start of the string.
* (0[1-9]|1[0-2]) – matches a month from 01 to 12.
* - – matches the hyphen separator.
* (0[1-9]|[12][0-9]|3[01]) – matches a day from 01 to 31.
* - – matches the hyphen separator.
* \d{4} – matches exactly 4 digits for the year.
* $ – end of the string.

**Matching a URL**

To match a simple URL like https://example.com:

^https?://[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$

**Explanation**:

* ^ – start of the string.
* https? – matches http or https (the s? means s is optional).
* :// – literal ://.
* [a-zA-Z0-9.-]+ – matches the domain name (letters, digits, hyphens, and dots).
* \. – literal dot.
* [a-zA-Z]{2,} – matches the top-level domain (at least two letters).
* $ – end of the string.

**Validating an Email. username@example.com**

^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$

**Explanation**:

* ^[a-zA-Z0-9.\_%+-]+: Start of string, then one or more alphanumeric characters, dots, underscores, and some other special symbols.
* @: The @ symbol is required.
* [a-zA-Z0-9.-]+: Domain part with letters, numbers, and dots.
* \.[a-zA-Z]{2,}$: A dot followed by 2 or more letters (top-level domain like .com, .org).