```python
import re
```

Copy code

🔹 Common Regex Functions in Python

Function	Usage	Example
`re.match(pattern, string)`	Matches only **at the beginning** of the string	`re.match(r"\d+", "123abc")` → `"123"`
`re.search(pattern, string)`	Finds the **first occurrence**	`re.search(r"\d+", "abc123xyz")` → `"123"`
`re.findall(pattern, string)`	Returns **all matches** in a list	`re.findall(r"\d+", "a1 b22 c333")` → `["1","22","333"]`
`re.finditer(pattern, string)`	Returns **iterable of Match objects**	`for m in re.finditer(r"\d+", "a1 b22"): print(m.group())`
`re.sub(pattern, repl, string)`	Replace matches	`re.sub(r"\d+", "#", "a1 b22")` → `"a# b#"`
`re.split(pattern, string)`	Split string by regex	`re.split(r"\s+", "one two three")` → `["one","two","three"]`
`re.compile(pattern, flags=0)`	Precompile regex for reuse	`pat = re.compile(r"\d+")` then `pat.findall("123abc")`

🔹 Regex Metacharacters

Pattern	Meaning	Example
`.`	Any character (except newline `\n`)	`re.findall(r"a.c", "abc a-c aXc")` → `["abc","a-c","aXc"]`
`^`	Start of string	`re.findall(r"^Hi", "Hi there\nHi again")` → `["Hi"]`
`$`	End of string	`re.findall(r"end$", "the end")` → `["end"]`
`*`	0 or more	`re.findall(r"\d*", "123 abc")` → `["123","",""]`
`+`	1 or more	`re.findall(r"\d+", "123 abc")` → `["123"]`
`?`	0 or 1 (optional)	`re.findall(r"colou?r", "color colour")` → `["color","colour"]`
`{n}`	Exactly n	`re.findall(r"\d{3}", "12345")` → `["123"]`
`{n,}`	At least n	`re.findall(r"\d{2,}", "1 12 123")` → `["12","123"]`
`{n,m}`	Between n and m	`re.findall(r"\d{2,4}", "1 12 123 12345")` → `["12","123","1234"]`
`[]`	Character set	`re.findall(r"[aeiou]", "hello")` → `["e","o"]`
`[^]`	Negated set	`re.findall(r"[^0-9]", "a1b2")` → `["a","b"]`

`	`	OR
`()`	Group / capture	`re.search(r"(abc)+", "abcabc")` → `"abcabc"`

🔹 Predefined Character Classes

Shortcut	Meaning	Example
`\d`	Digit (0–9)	`re.findall(r"\d", "a1b2")` → `["1","2"]`
`\D`	Non-digit	`re.findall(r"\D", "a1")` → `["a"]`
`\w`	Word char (letters, digits, _)	`re.findall(r"\w", "a_1!")` → `["a","_","1"]`
`\W`	Non-word char	`re.findall(r"\W", "a_1!")` → `["!"]`
`\s`	Whitespace (space, tab, newline)	`re.findall(r"\s", "a b\tc")` → `[" ","\t"]`
`\S`	Non-whitespace	`re.findall(r"\S", "a b")` → `["a","b"]`
`\b`	Word boundary	`re.findall(r"\bcat\b", "cat scatter")` → `["cat"]`
`\B`	Not a word boundary	`re.findall(r"\Bcat\B", "scattered")` → `["cat"]`

Flags (Modifiers)

Flag	Usage	Example
`re.IGNORECASE` or `re.I`	Case-insensitive	`re.findall(r"abc", "ABC", re.I)` → `["ABC"]`
`re.MULTILINE` or `re.M`	`^` and `$` match each line	`re.findall(r"^Hi", "Hi\nHi", re.M)` → `["Hi","Hi"]`
`re.DOTALL` or `re.S`	`.` matches newline too	`re.findall(r"a.+c", "a\nc", re.S)` → `["a\nc"]`
`re.VERBOSE` or `re.X`	Ignore whitespace & comments in pattern	`re.compile(r"""\d+ # match digits""", re.X)`

Practical Examples

python Copy code

```python
# 1. Extract all emails
text = "Contact me at test@mail.com or admin@xyz.org"
emails = re.findall(r"[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}", text)
print(emails)  # ['test@mail.com', 'admin@xyz.org']

# 2. Validate phone number (Indian format)
phone = "9876543210"
if re.fullmatch(r"[6-9]\d{9}", phone):
    print("Valid phone")
```

```python
# 1. Extract all emails
text = "Contact me at test@mail.com or admin@xyz.org"
emails = re.findall(r"[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}", text)
print(emails)  # ['test@mail.com', 'admin@xyz.org']

# 2. Validate phone number (Indian format)
phone = "9876543210"
if re.fullmatch(r"[6-9]\d{9}", phone):
    print("Valid phone")
else:
    print("Invalid")

# 3. Replace multiple spaces with one
line = "This   has   too    many spaces"
print(re.sub(r"\s+", " ", line))  # "This has too many spaces"

# 4. Extract numbers
s = "Order 12 items, each 99 rupees"
print(re.findall(r"\d+", s))  # ['12', '99']

# 5. Split by comma or semicolon
data = "apple,orange;banana,grape"
print(re.split(r"[;,]", data))  # ['apple', 'orange', 'banana', 'grape']
```

1. Remove extra spaces, tabs, and newlines

```python
text = "This   has   too    many   spaces\nand tabs\t\t."
clean = re.sub(r"\s+", " ", text).strip()
print(clean)
# Output: "This has too many spaces and tabs ."
```

---

2. Remove special characters (keep only letters & numbers)

```python
text = "Hello!!! This #is @messy$$ data123."
clean = re.sub(r"[^a-zA-Z0-9\s]", "", text)
print(clean)
# Output: "Hello This is messy data123"
```

---

3. Keep only numbers (extract digits)

```python
text = "Order ID: 12345, Price: 999 INR"
numbers = re.findall(r"\d+", text)
print(numbers)
# Output: ['12345', '999']
```

4. Remove numbers (keep only text)

python Copy code

```python
text = "User123 bought 45 items"
clean = re.sub(r"\d+", "", text)
print(clean)
# Output: "User bought  items"
```

---

5. Remove HTML/XML tags

python Copy code

```python
text = "<p>Hello <b>World</b></p>"
clean = re.sub(r"<.*?>", "", text)
print(clean)
# Output: "Hello World"
```

---

6. Normalize case (lowercase all text)

python Copy code

```python
text = "Clean THIS TeXT"
clean = text.lower()
print(clean)
# Output: "clean this text"
```

7. Remove punctuation

```python
text = "Hello, world! How's it going?"
clean = re.sub(r"[^\w\s]", "", text)
print(clean)
# Output: "Hello world Hows it going"
```

8. Validate & clean emails

```python
text = "Contact: test@mail.com, wrong@mail, hello@xyz.org"
emails = re.findall(r"[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}", text)
print(emails)
# Output: ['test@mail.com', 'hello@xyz.org']
```

9. Clean phone numbers (keep only digits)

```python
phone = "+91-98765 43210"
clean = re.sub(r"\D", "", phone)  # remove non-digits
print(clean)
# Output: "919876543210"
```

10. Remove stopwords (basic cleaning)

```python
text = "This is a sample sentence with stopwords"
stopwords = r"\b(is|a|with|the|and|of|to)\b"
clean = re.sub(stopwords, "", text, flags=re.I)
clean = re.sub(r"\s+", " ", clean).strip()
print(clean)
# Output: "This sample sentence stopwords"
```