PYTHON FOR COMPUTATIONAL PROBLEM SOLVING

Introduction to re Module

UE25CS151A

Department of Computer Science and Engineering
Prof. Kundhavai K R, CSE Department

re Module

- The re module in Python provides support for Regular Expressions (regex). Regex is a powerful text-processing tool used to search, match, and manipulate patterns in strings. Helps automate tasks like validation, extraction, text cleaning, and pattern matching. Need not use pip install, as it is a built-in (standard library) module in Python.

- A **pattern** defines what you want to search in text
- Examples:
  - `r"\d+"` → any sequence of digits
    `r"\b[Pp]\w+"` → words starting with P or p
    `r"\w{3}"` → exactly 3 characters

Why not use string methods?

- `.find()` searches exact text
- RegEx can match **flexible patterns**

Key Metacharacters

Meta	Meaning	Example
.	Any character (except newline)	h.t → **hat**, **hot**
\d	Digit (0–9)	\d\d → **12**
\w	Word character (a-z, A-Z, 0-9, _)	\w\w → **hi**, **A5**
\s	Whitespace	hello\sworld → **hello world**
+	One or more	\d+ → **123**
*	Zero or more	ab*c → **ac**, **abc**
[]	Character set	[aeiou] → any vowel
^	Start of string	^Hello
$	End of string	world$

Main Functions in re

re.search() → finds **first match**

re.findall() → finds **all matches** in a list

re.search() Example

```
import re
text = "My car brand is Audi."
pattern = r"c\w\w"  # 3-letter word starting with 'c'
match = re.search(pattern, text)
if match:
    print(match.group(), match.start(), match.end())
```

Output:

```
car 3 6
```

- `.group()` → actual text
  `.start()` → starting index
  `.end()` → ending index

re.findall() Examples

```
text = "My numbers: 123, 456, 789."
print(re.findall(r"\d", text))      # single digits → ['1','2',...]
print(re.findall(r"\d+", text))     # full numbers → ['123','456','789']
print(re.findall(r"\w+", "Python is fun!"))  # words → ['Python','is','fun']
```

re.sub() Example

Replace spaces with underscores

```
import re
text = "Welcome to Python class"
new_text = re.sub(r"\s", "_", text)
print(new_text)
```

Output:

```
Welcome_to_Python_class
```

re.sub(pattern, replacement, text) → **substitute** function

Word Boundaries & Quantifiers

- \b → word boundary
- {n} → exactly n characters
- Examples:

```
text = "The fox ran"
print(re.findall(r"\b\w{3}\b", text))  # ['The','fox','ran']
print(re.findall(r"\w+ing", "learning coding swimming"))  # ['learning','coding','swimming']
```

Splitting and Replacing

Splitting

```
text = "apple, banana orange,grapes"
print(re.split(r"[\s,]+", text))  # split by spaces or commas
```

Output: ['apple','banana','orange','grapes']

Replacing

```
text = "Password: 12345"
print(re.sub(r"\d","*",text))  # replace digits → Password: *****
```

Matching Patterns

Words starting with p or P:

```
text = "Python programming is popular"
pattern = r"\b[Pp]\w+"
print(re.findall(pattern, text))  # ['Python','programming','popular']
```

Text starting with "Hello":

```
text = "Hello world"
pattern = r"^Hello"
if re.search(pattern, text): print("Starts with Hello")
```

Emails & Phone Numbers

- **Find emails:**

```
text = "info@pes.edu, support@pes.com"
print(re.findall(r"\S+@\S+", text))
# ['info@pes.edu','support@pes.com']
```

- **Find 10-digit numbers:**

```
text = "Call 9876543210 or 9123456789"
print(re.findall(r"\d{10}", text))
```

Emails & Phone Numbers

- **Find 10-digit numbers starting with 9:**

```
text = "9000000000, 8123456789"
print(re.findall(r"\b9\d{9}\b", text))  # ['9000000000']
```

THANK YOU

Department of Computer Science and Engineering

Prof. Kundhavai K R, CSE Department

Ack: Teaching Assistant:
Adithya Jeyaramsankar– PES2UG22CS029