25/9/25- sys, glob, and re

1. The sys Module

The sys module provides access to **system-specific parameters and functions**.

1.1 Python Version & Platform

```
import sys

print("Python Version:", sys.version)
print("Platform:", sys.platform)
```

Output (example):

```
Python Version: 3.12.0 (main, Oct 2024, ...)
[GCC 11.2.0]
Platform: win32
```

🔎 **Theory**

- sys.version → Returns current Python version details.
- sys.platform → Returns the operating system name (win32, linux, darwin for Mac).

1.2 Command Line Arguments

```
import sys
print("Arguments:", sys.argv)
```

Output (if run as python script.py hello):

```
Arguments: ['script.py', 'hello']
```

🔎 **Theory**

- sys.argv → List of arguments passed to the script from the command line.
- Useful for automation and batch processing.

1.3 Exiting a Program

```
import sys
print("Before exit")
sys.exit()
print("This will not print")
```

Output:

```
Before exit
```

🔎 **Theory**

- sys.exit() → Immediately stops the program execution.

2. The glob Module

The glob module is used to **search for files and directories** using wildcards (*, ?).

2.1 List Python Files

```
import glob
print(glob.glob("*.py"))
```

Output (example):

```
['main.py', 'test.py', 'script.py']
```

🔎 **Theory**

- glob.glob("*.py") → Finds all files in the current folder that end with .py.

2.2 List CSV Files in a Folder

```
import glob
print(glob.glob("data/*.csv"))
```

Output (example):

```
['data/file1.csv', 'data/file2.csv']
```

🔎 **Theory**

- Matches files inside data folder ending with .csv.

2.3 Recursive Search

```
import glob
print(glob.glob("**/*.txt", recursive=True))
```

Output (example):

```
['notes.txt', 'docs/readme.txt']
```

🔎 **Theory**

- recursive=True → Searches inside all subfolders as well.

3. The re Module (Regular Expressions)

The re module is used for **pattern matching and text searching**.

3.1 search() – Find a Pattern

```
import re
text = "I love Python"
result = re.search("Python", text)
print("Found at:", result.start())
```

Output:

```
Found at: 7
```

🔎 **Theory**

- re.search(pattern, text) → Finds first match of pattern in text.

3.2 findall() – Find All Matches

```
import re
text = "apple, banana, apple, mango"
print(re.findall("apple", text))
```

Output:

['apple', 'apple']

🔎 **Theory**

- Returns **all matches** of the pattern as a list.

3.3 sub() – Replace Pattern

```
import re
text = "I like Java"
print(re.sub("Java", "Python", text))
```

Output:

I like Python

🔎 **Theory**

- re.sub(old, new, text) → Replaces all occurrences of a pattern.

3.4 split() – Split Text by Pattern

```
import re
text = "one,two;three four"
print(re.split("[,; ]", text))
```

Output:

['one', 'two', 'three', 'four']

🔎 **Theory**

- Splits text wherever it finds a comma, semicolon, or space.