الاسم / اسلام السيد رمزى الغرباوى

سيكشن / 1

CHAPTER 4

MCQ:

1. re.match()

2. One or more digits

3. (ing)$

4. ['o', 'o', 'a', 'i']

5. ^[a-zA-Z_]\w$*

6. Negation (not these characters)

7. ['Python', 'is', 'easy']

8. re.sub()

True or False:

1. False

2. True

3. False

4. True

5. True

6. True

7. False

8. False

Short Answers:

1. Differentiate between re.match() and re.search().

- re.match() checks the pattern only at the beginning of the string.
- re.search() checks anywhere in the string and returns the first match

2. **Explain the difference between +, *, and ? quantifiers.**

- + → one or more
- * → zero or more
- ? → zero or one

3. **Purpose of named groups in regex + example**

Named groups allow you to label parts of a regex so they can be accessed by name.

Example:

pattern = r"(?P\d{4})-(?P\d{2})-(?P\d{2})"

You can extract using:
match.group("year")

4. Regex to match a date (YYYY-MM-DD)

^\d{4}-\d{2}-\d{2}$

5. What does this pattern validate?

^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$

6. Why use re.split(r"\s+", text) instead of str.split()?

- str.split() splits only on single spaces
- re.split(r"\s+") splits on any amount of whitespace (tabs, multiple spaces,
  newlines)

7. Difference between raw string (r"pattern") and normal string?

Raw strings don’t escape backslashes, so regex is cleaner.

Example:
Normal → "\\d+"
Raw → r"\d+"

8. **How can re.sub() be used for text normalization?**

Example: Remove multiple spaces:

re.sub(r"\s+", " ", text)
This replaces any number of spaces with one space.

#Problem 1

```
import re

emails=["user@example.com","bad-email", "test@domain.org"]

pattern=r"^[A-Za-z0-9._]+@[A-Za-z0-9.-]+\.(com|org|edu)$"

for email in emails:
```

```
    if re.match(pattern, email):
        print(f"{email} → Valid")
    else:
        print(f"{email} → Invalid")
```

#Problem 2

```
import re
text = "I love #Python and #AI."
hashtags = re.findall(r"#\w+", text)
print(hashtags)
```

#Problem 3

```
import re
pattern = r"^(\(\d{3}\)|\d{3})[-.]\d{3}[-.]\d{4}$"
tests=["(123)-456-7890","123-456-7890",
"123.456.7890", "5551234"]
for t in tests:
   print(t, "→", bool(re.match(pattern, t)))
```

#Problem 4

```
import re
from collections import Counter
text = "Python, Python! AI is great; python AI."
words = re.findall(r"[A-Za-z]+", text)
counts = Counter(word.capitalize() for word in words)
print(dict(counts))
```

#Problem 5

```
import re
text = "This is is a test test"
duplicates = re.findall(r"\b(\w+)\s+\1\b", text)
matches = [f"{w} {w}" for w in duplicates]
print(matches)
```

#Problem 6

```
import re
text = "The events are on 2023-05-12 and 2024-01-01."
dates = re.findall(r"\d{4}-\d{2}-\d{2}", text)
print(dates)
```

#Problem 7

```
import re
text = "Card: 1234-5678-9012-3456"
masked = re.sub(r"\d(?=\d{4})", "*", text)
print(masked)
```

#Problem 8

```
import re
text = "I know Python, Java, and C++ but not Ruby."
languages = re.findall(r"\b[A-Za-z]+\+*\b", text)
```

```
print(languages)
```