**PYTHON: UNIT-2/ Labex2**

**CODE:**

import re

# Sample text

text = """

abc

defabc

abc123

def

a.b

a\nb

word

sword

password

wordy

a\_word

\_word\_

Word

word

@example

contact@domain.org

@mention

nonmention

mixed@mention

Here are some email addresses:

user1@example.com, contact@domain.org, and a username without an @ sign.

Another line: @mention, nonmention, and mixed@mention.

abb ab abbb aaa aaaa aaaaa aa a

bbb

aaab

aab

a b

(abc

ABC

AbC

1abc

xyz

pqr

"""

# 1. Using (?m)^abc: Extract all lines that start with 'abc'

lines\_starting\_with\_abc = re.findall(r'(?m)^abc', text)

print("Lines starting with 'abc':", lines\_starting\_with\_abc)

# 2. Using (?s)a.b: Extract all occurrences of 'a' followed by any single character (including newline), followed by 'b'

occurrences\_a\_b = re.findall(r'(?s)a.b', text)

print("Occurrences of 'a' followed by any single character and 'b':", occurrences\_a\_b)

# 3. Using \bword\b: Extract all occurrences of the whole word 'word'

whole\_word\_occurrences = re.findall(r'\bword\b', text)

print("Occurrences of the whole word 'word':", whole\_word\_occurrences)

# 4. Using \Bword\B: Extract all occurrences of 'word' when it is part of a larger word

part\_of\_larger\_word = re.findall(r'\Bword\B', text)

print("Occurrences of 'word' as part of a larger word:", part\_of\_larger\_word)

# 5. Using (?<=@)\w+: Extract all sequences of word characters that are immediately preceded by an @

sequences\_after\_at = re.findall(r'(?<=@)\w+', text)

print("Sequences of word characters immediately preceded by '@':", sequences\_after\_at)

# 6. Using (?<!@)\w+: Extract all sequences of word characters that are not immediately preceded by an @

sequences\_not\_after\_at = re.findall(r'(?<!@)\w+', text)

print("Sequences of word characters not immediately preceded by '@':", sequences\_not\_after\_at)

# 7. Using a?b: Extract all matches of the pattern a?b

matches\_a\_b = re.findall(r'a?b', text)

print("Matches of the pattern 'a?b':", matches\_a\_b)

# 8. Using a{3}: Extract all matches of the pattern a{3}

matches\_a3 = re.findall(r'a{3}', text)

print("Matches of the pattern 'a{3}':", matches\_a3)

# 9. Using a{2,4}: Extract all matches of the pattern a{2,4}

matches\_a2\_4 = re.findall(r'a{2,4}', text)

print("Matches of the pattern 'a{2,4}':", matches\_a2\_4)

# 10. Using [abc]: Extract all single characters that are either 'a', 'b', or 'c'

characters\_abc = re.findall(r'[abc]', text)

print("Single characters 'a', 'b', or 'c':", characters\_abc)

# 11. Using [^abc]: Extract all single characters that are not 'a', 'b', or 'c'

characters\_not\_abc = re.findall(r'[^abc]', text)

print("Single characters not 'a', 'b', or 'c':", characters\_not\_abc)

# 12. Using \(abc: Extract all occurrences of the literal string (abc

literal\_string = re.findall(r'\(abc', text)

print("Occurrences of the literal string '(abc':", literal\_string)

# 13. Using (?i)abc: Extract all occurrences of the string 'abc' in a case-insensitive manner

case\_insensitive\_abc = re.findall(r'(?i)abc', text)

print("Occurrences of 'abc' in a case-insensitive manner:", case\_insensitive\_abc)

**OUTPUT:**

