1.python programme to replace all occurrence of a space,comma,or dot with a colon

In [3]:
```
sample_text='Python Exercises,PHP exercises.'
chars_to_replace=[' ','.',',']

def replace_chars_with_colon(text):
    for char in chars_to_replace:
        text=text.replace(char,':')
    return text

result=replace_chars_with_colon(sample_text)

print('expected output:',result)
```
```
expected output: Python:Exercises:PHP:exercises:
```

2

In [19]:
```
import pandas as pd
import re

Data={'SUMMARY':['hello,world!','**** test','123four, five, six...']}
df=pd.DataFrame(Data)

def cleaned_Data(Data):
    cleaned_Data=re.sub(r'[^\w\s]','',Data)
    return cleaned_Data

df ['SUMMARY']=df['SUMMARY'].apply(cleaned_Data)
print(df)
```
```
            SUMMARY
0        helloworld
1              test
2  123four five six
```

3

In [14]:
```
import re
input_text = "This is a sample text with some words of different lengths."

def find_long_words(input_string):
    pattern = re.compile(r'\b\w{4,}\b')
    result = pattern.findall(input_string)
    return result
```

```
result = find_long_words(input_text)
print(result)
```

```
['This', 'sample', 'text', 'with', 'some', 'words', 'different', 'lengths']
```

In [15]:
```
#QUESTION 4
import re

def find_words(text):
    pattern = re.compile(r'\b\w{3,5}\b')
    words = re.findall(pattern, text)
    return words

sample_text = "Ram and Meena are going for a trip."
words = find_words(sample_text)
print(words)
```

```
['Ram', 'and', 'Meena', 'are', 'going', 'for', 'trip']
```

In [ ]:
```
#question 5
```

In [16]:
```
def remove_parentheses(strings):
    pattern = re.compile(r'\([^)]*\)')
    cleaned_strings = [re.sub(pattern, '', string)for string in strings]
    return cleaned_strings

sample_strings = ["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello (Dat
cleaned_strings = remove_parentheses(sample_strings)
for string in cleaned_strings:
    print(string)
```

```
example
hr@fliprobo
github
Hello
Data
```

In [4]:
```
#question 6
import re
sample_text=["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello (Data Sci
"Data (Scientist)"]
text = file.read()

text = re.sub(r'\([^)]*\)', '', text)
output = [word.strip() for word in text.split(',')]

print(output)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[4], line 5
      2 import re
      3 sample_text=["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello
(Data Science World)",
      4 "Data (Scientist)"]
----> 5 text = file.read()
      7 text = re.sub(r'\([^)]*\)', '', text)
      8 output = [word.strip() for word in text.split(',')]

NameError: name 'file' is not defined
```

In [24]:
```
#question 7
```

In [6]:
```
sample_text="ImportanceOfRegularExpressionInPython"
pattern=re.compile(r'[A-Z][a-z]*')
result=pattern.findall(sample_text)
print(result)
```

```
['Importance', 'Of', 'Regular', 'Expression', 'In', 'Python']
```

In [5]:
```
#question 8
```

In [7]:
```
import re
text="RegularExpression1IsAn2ImportantTopic3InPython"
def insert_spaces(text):
    pattern=re.compile(r'(?<=[a-zA-Z])(?=\d)')
    new_text=re.sub(pattern,' ',text)
    return  new_text
result=insert_spaces(text)
print(result)
```

```
RegularExpression 1IsAn 2ImportantTopic 3InPython
```

In [8]:
```
#question 9
```

In [5]:
```
import re
sample_text = "RegularExpression1IsAn2ImportantTopic3InPython"

def insert_spaces(text):
    pattern = r'(?<=[a-z])(?=[A-Z0-9])|(?<=[0-9])(?=[A-Za-z])'
    modified_text = re.sub(pattern, ' ', text)
    return modified_text


output = insert_spaces(sample_text)
print(output)
```

```
Regular Expression 1 Is An 2 Important Topic 3 In Python
```

In [3]:
```
#question 11
```

In [4]:
```
import re

def match_string(string):
    pattern = r'^[a-zA-Z0-9_]+$'
    if re.match(pattern, string):
        print("The string matches the pattern.")
    else:
        print("The string does not match the pattern.")

# Test the program
string1 = "Hello_World123"
match_string(string1)  # Output: The string matches the pattern.

string2 = "Hello-World"
match_string(string2)
```

```
The string matches the pattern.
The string does not match the pattern.
```

In [10]:
```
# question 12
string1 = "123abc"
starts_with_number(string1,123)

string2 = "abc123"
starts_with_number(string2,123)

def starts_with_number(string, number):
    if string.startswith(str(number)):
        print(f"The string '{string}' starts with the number {number}.")
    else:
        print(f"The string '{string}' does not start with the number {number}.")
```

```
The string '123abc' starts with the number 123.
The string 'abc123' does not start with the number 123.
```

In [12]:
```
#question 13

import re
ip_address = "315.09.083.215"

def remove_leading_zeros(ip_address):
    pattern = r'\b0+(\d)'
    modified_ip = re.sub(pattern, r'\1', ip_address)
    return modified_ip

output = remove_leading_zeros(ip_address)
print(output)
```

```
315.9.83.215
```

In [26]:
```
#Question 14-


import re


text = "On August 15th 1947 that India was declared independent from British coloniali

pattern = r"\b([A-Z][a-z]+) \d{1,2}(?:st|nd|rd|th)? \d{4}\b"

matches = re.findall(pattern, text)
print(matches)
```

```
['August']
```

In [14]:
```
#question 15
sample_text = "The quick brown fox jumps over the lazy dog."
searched_words = ['fox', 'dog', 'horse']
found_words = []
def search_strings(text, searched_words):

    for word in searched_words:
        if word in text:
            found_words.append(word)
    return found_words

found_words = search_strings(sample_text, searched_words)
print(found_words)
```

```
['fox', 'dog']
```

In [16]:
```
#question 16
import re
sample_text = 'The quick brown fox jumps over the lazy dog.'
searched_word = 'fox'
search_and_locate_string(sample_text, searched_word)

def search_and_locate_string(text, pattern):
    match = re.search(pattern, text)
    if match:
        start = match.start()
        end = match.end()
        print(f'Found "{pattern}" in "{text}" from {start} to {end}')
    else:
        print(f'"{pattern}" not found in "{text}"')
```

```
Found "fox" in "The quick brown fox jumps over the lazy dog." from 16 to 19
```

In [18]:
```
#question 17
sample_text='Python exercises,PHP exercises,C# exercises'
pattern='exercises'

def find_substrings(text,pattern):
    substrings=re.findall(pattern,text)
    return substrings

substrings=find_substrings(sample_text,pattern)
print(substrings)
```

```
['exercises', 'exercises', 'exercises']
```

In [23]:
```
#question 18
import re
sample_text='Python exercises,PHP exercises,C# exercises'

def find_occurrences(text, pattern):
    occurrences = []
    for match in re.finditer(pattern, text):
        start = match.start()
        end = match.end()
        occurrences.append((text[start:end], start, end))
    return occurrences


pattern = 'exercises'

occurrences = find_occurrences(sample_text, pattern)
for occurrence in occurrences:
    substring, start, end = occurrence
    print(f'Found "{substring}" at {start}:{end}')
```

```
Found "exercises" at 7:16
Found "exercises" at 21:30
Found "exercises" at 34:43
```

In [24]:
```
# question 19
date='2024-01-31'

def convert_date_format(date):
    parts=date.split('-')
    new_date=f"{parts[2]}-{parts[1]}-{parts[0]}"
    return new_date

new_date=convert_date_format(date)
print(new_date)
```

```
31-01-2024
```

In [27]:
```
#question 20
import re
sample_text = "01.12 0132.123 2.31875 145.8 3.01 27.25 0.25"

def find_decimal_numbers(text):
    pattern = re.compile(r'\b\d+\.\d{1,2}\b')
    decimal_numbers = pattern.findall(text)
    return decimal_numbers



decimal_numbers = find_decimal_numbers(sample_text)
print(decimal_numbers)
```

```
['01.12', '145.8', '3.01', '27.25', '0.25']
```

In [28]:
```
#question 21
import re
sample_text = "01.12 0132.123 2.31875 145.8 3.01 27.25 0.25"

def find_numbers_with_positions(text):
    pattern = re.compile(r'\b\d+\b')
    numbers_with_positions = []
    for match in pattern.finditer(text):
        number = match.group()
        position = match.start()
        numbers_with_positions.append((number, position))
    return numbers_with_positions


numbers_with_positions = find_numbers_with_positions(sample_text)
for number, position in numbers_with_positions:
    print(f'Number: {number}, Position: {position}')
```

```
Number: 01, Position: 0
Number: 12, Position: 3
Number: 0132, Position: 6
Number: 123, Position: 11
Number: 2, Position: 15
Number: 31875, Position: 17
Number: 145, Position: 23
Number: 8, Position: 27
Number: 3, Position: 29
Number: 01, Position: 31
Number: 27, Position: 34
Number: 25, Position: 37
Number: 0, Position: 40
Number: 25, Position: 42
```

In [30]:
```
#question 22
import re

def extract_maximum_numeric_value(string):
    numbers = re.findall(r'\d+', string)
    if numbers:
        max_number = max(map(int, numbers))
        return max_number
    else:
        return None

# Test the program
sample_string = "200kl861hgfc234ye"
max_number = extract_maximum_numeric_value(sample_string)
print(f"The maximum numeric value in the string is: {max_number}")
```

```
The maximum numeric value in the string is: 861
```

In [34]:
```
#question 23

import re
sample_text = "RegularExpressionIsAnImportantTopicInPython"

def insert_spaces(string):
    modified_string = re.sub(r'(?=[A-Z])', ' ', string)
    return modified_string


modified_text = insert_spaces(sample_text)
print(modified_text)
```

```
 Regular Expression Is An Important Topic In Python
```

In [35]:
```
# question 24
import re
sample_string = "RegularExpressionIsAnImportantTopicInPython"

def find_sequences(string):
    sequences = re.findall(r'[A-Z][a-z]+', string)
    return sequences


sequences = find_sequences(sample_string)
print(sequences)
```

```
['Regular', 'Expression', 'Is', 'An', 'Important', 'Topic', 'In', 'Python']
```

In [36]:
```
#question 25
import re

def remove_duplicate_words(sentence):
    modified_sentence = re.sub(r'\b(\w+)( \1\b)+', r'\1', sentence)
    return modified_sentence

# Test the program
sample_text = "Hello hello world world"
modified_text = remove_duplicate_words(sample_text)
print(modified_text)
```

```
Hello hello world
```

In [ ]:

In [ ]:
```
#question 26
```

In [11]:
```
import re

def check_string(string):
    pattern = r'^.*[a-zA-Z0-9]$'
    if re.match(pattern, string):
        print("Accepted")
    else:
        print("Not Accepted")

check_string("manvirai326")
check_string("manvirai@")
```

```
Accepted
Not Accepted
```

In [13]:
```
#question 27

import re

def extract_hashtags(text):
    hashtags = re.findall(r'#\w+', text)
    return hashtags

# Test the program
text = 'RT @kapil_kausik: #Doltiwal I mean #xyzabc is "hurt" by #Demonetization as the
hashtags = extract_hashtags(text)
print(hashtags)
```

```
['#Doltiwal', '#xyzabc', '#Demonetization']
```

In [20]:
```
#question 28
import re


sample_text = "@Jags123456 Bharat band on 28??<ed><U+00A0><U+00BD><ed><U+00B8><U+0082>

modified_text = re.sub(r'<U\+\w+>', '', sample_text)


print(modified_text)
```

```
@Jags123456 Bharat band on 28??<ed><ed>Those who are protesting #demonetization are a
ll different party leaders
```

In [29]:
```
#question29
import re
```

In [24]:
```
#question 30

import re

def remove_words(text):
    pattern = re.compile(r'\b\w{2,4}\b')
    modified_text = pattern.sub('', text)
    return modified_text


text = "The following example creates an ArrayList with a capacity of 50 elements. 4 e
modified_text = remove_words(text)
print(modified_text)
```

```
 following example creates  ArrayList  a capacity   elements. 4 elements   added   Ar
rayList   ArrayList  trimmed accordingly.
```

In [ ]: