Question 1-Write a Python program to replace all occurrences of a space, comma, or dot with a colon.

Sample Text-'Python Exercises, PHP exercises.' Expected Output:Python:Exercises::PHP:exercises:

In [1]:
```
def replace_characters(text):
    replacements = [' ', ',', '.']

    for char in replacements:
        text = text.replace(char, ':')
    pi
```

```
Python:Exercises::PHP:exercises:
```

Question 2- Write a Python program to find all words starting with 'a' or 'e' in a given string.

In [2]:
```
import re

def find_words_starting_with_a_or_e(input_string):
    words = re.findall(r'\b[aeAE]\w+\b', input_string)
    return words

# Test the function
input_string = "Apple and elephant are animals, while eagle is a bird."
result = find_words_starting_with_a_or_e(input_string)
print(result)
```

```
['Apple', 'and', 'elephant', 'are', 'animals', 'eagle']
```

Question 3- Create a function inpython to find all words that are at least 4 characters long in a string.The use of the re.compile() method is mandatory.

In [3]:
```
import re

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

# Test the function
input_string = "This is a sample string with some long words like apple, banana
result = find_long_words(input_string)
print(result)
```

```
['This', 'sample', 'string', 'with', 'some', 'long', 'words', 'like', 'appl
e', 'banana', 'orange']
```

Question 4-Create a function in python to find all three, four, and five character words in a string. The use of the re.compile() method is mandatory.

In [4]:
```
import re

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

input_string = "This is a test string with some short and long words."
result = find_words_of_length(input_string)
print(result)
```

```
['This', 'test', 'with', 'some', 'short', 'and', 'long', 'words']
```

Question 5- Create a function in Python to remove the parenthesis in a list of strings.The use of the re.compile() method is mandatory.

Sample Text:["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello (Data Science World)", "Data (Scientist)"] Expected Output: example.com hr@fliprobo.com (mailto:hr@fliprobo.com) github.com Hello Data Science World Data Scientist

In [5]:
```
import re

def remove_parentheses(strings):
    result = []
    pattern = re.compile(r'\([^)]*\)')  # Regular expression to match anything

    for string in strings:
        modified_string = re.sub(pattern, '', string)
        result.append(modified_string)

    return result

sample_text = ["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello
output = remove_parentheses(sample_text)

for item in output:
    print(item)
```

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

Question 7-Write a regular expression in Python to split a string into uppercase letters.

Sample text: “ImportanceOfRegularExpressionsInPython” Expected Output:[‘Importance’, ‘Of’, ‘Regular’, ‘Expression’, ‘In’, ‘Python’]

In [22]:
```
import re

sample_text = "ImportanceOfRegularExpressionsInPython"
result = re.findall('[A-Z][a-z]*', sample_text)
print(result)
```

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

Question 8- Create a function in python to insert spaces between words starting with numbers.

Sample Text: “RegularExpression1IsAn2ImportantTopic3InPython" Expected Output: RegularExpression1IsAn 2ImportantTopic 3InPython

In [23]:
```
import re

def insert_spaces(text):
    modified_text = re.sub(r'(\d)([A-Za-z])', r'\1 \2', text)
    return modified_text

sample_text = "RegularExpression1IsAn2ImportantTopic3InPython"
result = insert_spaces(sample_text)
print(result)
```

```
RegularExpression1 IsAn2 ImportantTopic3 InPython
```

Question 9- Create a function in python to insert spaces between words starting with capital letters or with numbers.

Sample Text: “RegularExpression1IsAn2ImportantTopic3InPython" Expected Output:RegularExpression 1 IsAn 2 ImportantTopic 3 InPython

In [24]:
```
import re

def insert_spaces(text):
    result = re.sub(r'(?<=[a-z])([A-Z0-9])', r' \1', text)
    return result

sample_text = "RegularExpression1IsAn2ImportantTopic3InPython"
output = insert_spaces(sample_text)
print(output)
```

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

Question 10- Write a python program to extract email address from the text stored in the text file using Regular Expression.

Sample Text- Hello my name is Data Science and my email address is xyz@domain.com (mailto:xyz@domain.com) and alternate email address is xyz.abc@sdomain.domain.com (mailto:xyz.abc@sdomain.domain.com). Please contact us at hr@fliprobo.com (mailto:hr@fliprobo.com) for further information. Expected Output: ['xyz@domain.com', 'xyz.abc@sdomain.domain.com'] ['hr@fliprobo.com']

Note- Store given sample text in the text file and then extract email addresses.

In [25]:
```
import re

def extract_emails(text):
    pattern = r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'
    return re.findall(pattern, text)

def main():
    file_name = 'sample_text.txt'

    try:
        with open(file_name, 'r') as file:
            text = file.read()
            emails = extract_emails(text)

            primary_emails = []
            alternate_emails = []

            for email in emails:
                if 'alternate' in email:
                    alternate_emails.append(email)
                else:
                    primary_emails.append(email)

            print("Primary Email Addresses:", primary_emails)
            print("Alternate Email Addresses:", alternate_emails)

    except FileNotFoundError:
        print(f"File '{file_name}' not found.")

if __name__ == '__main__':
    main()
```

```
File 'sample_text.txt' not found.
```

Question 11- Write a Python program to match a string that contains only upper and lowercase letters, numbers, and underscores.

In [26]:
```
import re

def is_valid_string(s):
    pattern = r'^[a-zA-Z0-9_]+$'
    return re.match(pattern, s) is not None

# Test the function
test_strings = ["Hello_World123", "Python3", "invalid string!", "hello123#", "_
for s in test_strings:
    if is_valid_string(s):
        print(f"'{s}' is a valid string.")
    else:
        print(f"'{s}' is not a valid string.")
```

```
'Hello_World123' is a valid string.
'Python3' is a valid string.
'invalid string!' is not a valid string.
'hello123#' is not a valid string.
'_underscore_' is a valid string.
```

Question 12- Write a Python program where a string will start with a specific number.

In [10]:
```
def starts_with_number(string, number):
    # Convert the number to a string for comparison
    number_str = str(number)

    # Check if the string starts with the specified number
    if string.startswith(number_str):
        return True
    else:
        return False

# Test cases
string1 = "12345Hello"
number1 = 123
print(starts_with_number(string1, number1))  # Output: True

string2 = "56789World"
number2 = 123
print(starts_with_number(string2, number2))  # Output: False
```

```
True
False
```

Question 13- Write a Python program to remove leading zeros from an IP address

In [11]:
```
def remove_leading_zeros(ip_address):
    # Split the IP address into octets
    octets = ip_address.split('.')

    # Remove leading zeros from each octet
    cleaned_octets = [str(int(octet)) for octet in octets]

    # Join the cleaned octets to form the new IP address
    cleaned_ip_address = '.'.join(cleaned_octets)

    return cleaned_ip_address

# Test the function
ip_with_zeros = "192.168.001.001"
cleaned_ip = remove_leading_zeros(ip_with_zeros)
print("Original IP:", ip_with_zeros)
print("Cleaned IP:", cleaned_ip)
```

```
Original IP: 192.168.001.001
Cleaned IP: 192.168.1.1
```

Question 15- Write a Python program to search some literals strings in a string.

Sample text : 'The quick brown fox jumps over the lazy dog.' Searched words : 'fox', 'dog', 'horse'

In [13]:
```
def search_words(text, searched_words):
    found_words = []
    for word in searched_words:
        if word in text:
            found_words.append(word)
    return found_words

sample_text = 'The quick brown fox jumps over the lazy dog.'
searched_words = ['fox', 'dog', 'horse']

found_words = search_words(sample_text, searched_words)

if found_words:
    print("Found words:", found_words)
else:
    print("No words found.")
```

```
Found words: ['fox', 'dog']
```

Question 14- Write a regular expression in python to match a date string in the form of Month name followed by day number and year stored in a text file.

Sample text : 'On August 15th 1947 that India was declared independent from British colonialism, and the reins of control were handed over to the leaders of the Country'. Expected Output-August 15th 1947 Note- Store given sample text in the text file and then extract the date string asked format.

In [ ]:
```
import re

# Read the content of the text file
with open('sample.txt', 'r') as file:
    text = file.read()

# Define the regular expression pattern
pattern = r'\b(?:January|February|March|April|May|June|July|August|September|Oc

# Search for the pattern in the text
match = re.search(pattern, text)

# Extract and print the matched date string
if match:
    date_string = match.group(0)
    print("Extracted Date String:", date_string)
else:
    print("Date string not found in the text.")
```

Question 16- Write a Python program to search a literals string in a string and also find the location within the original string where the pattern occurs

Sample text : 'The quick brown fox jumps over the lazy dog.' Searched words : 'fox'

In [16]:
```
def find_pattern_locations(main_string, pattern):
    locations = []
    start = 0

    while start < len(main_string):
        index = main_string.find(pattern, start)
        if index == -1:
            break
        locations.append(index)
        start = index + 1

    return locations

sample_text = 'The quick brown fox jumps over the lazy dog.'
searched_word = 'fox'

pattern_locations = find_pattern_locations(sample_text, searched_word)

if pattern_locations:
    print(f"'{searched_word}' found at position(s): {pattern_locations}")
else:
    print(f"'{searched_word}' not found in the text.")
```

```
'fox' found at position(s): [16]
```

Question 17- Write a Python program to find the substrings within a string.

Sample text : 'Python exercises, PHP exercises, C# exercises' Pattern : 'exercises'.

In [17]:
```
import re

def find_substrings(text, pattern):
    matches = re.finditer(pattern, text)
    substrings = [match.group() for match in matches]
    return substrings

sample_text = 'Python exercises, PHP exercises, C# exercises'
search_pattern = 'exercises'

substrings = find_substrings(sample_text, search_pattern)
print("Substrings found:", substrings)
```

```
Substrings found: ['exercises', 'exercises', 'exercises']
```

Question 26- Write a python program using RegEx to accept string ending with alphanumeric character.

In [ ]:
```
import re

def validate_string(input_string):
    pattern = r'^.*[a-zA-Z0-9]$'
    if re.match(pattern, input_string):
        return True
    else:
        return False

# Get input from the user
user_input = input("Enter a string: ")

# Validate the input
if validate_string(user_input):
    print("Input string ends with an alphanumeric character.")
else:
    print("Input string does not end with an alphanumeric character.")
```

Question 20- Create a function in python to find all decimal numbers with a precision of 1 or 2 in a string.The use of the re.compile() method is mandatory.

In [ ]:
```
import re

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

# Example usage
input_string = "The price is $12.34 and the weight is 0.5 kg. The temperature i
decimal_numbers = find_decimal_numbers_with_precision(input_string)
print(decimal_numbers)
```

Question 21- Write a Python program to separate and print the numbers and their position of a given string.

In [ ]:
```
def separate_numbers_and_positions(input_string):
    numbers_and_positions = []

    for index, char in enumerate(input_string):
        if char.isdigit():
            numbers_and_positions.append((int(char), index))

    return numbers_and_positions

# Get input from the user
input_string = input("Enter a string: ")

# Call the function and print the result
result = separate_numbers_and_positions(input_string)
print("Numbers and their positions:")
for number, position in result:
    print(f"Number: {number}, Position: {position}")
```

Question 22- Write a regular expression in python program to extract maximum/largest numeric value from a string.

Sample Text: 'My marks in each semester are: 947, 896, 926, 524, 734, 950, 642' Expected Output: 950

In [ ]:
```
import re

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

sample_text = 'My marks in each semester are: 947, 896, 926, 524, 734, 950, 642
max_numeric_value = extract_max_numeric_value(sample_text)

if max_numeric_value is not None:
    print(f"Maximum numeric value: {max_numeric_value}")
else:
    print("No numeric values found in the input string.")
```

Question 23- Create a function in python to insert spaces between words starting with capital letters.

Sample Text: “RegularExpressionIsAnImportantTopicInPython" Expected Output:Regular

In [ ]:
```
import re

def insert_spaces(text):
    # Use regular expression to find words starting with capital letters
    pattern = r'(?<!^)(?=[A-Z])'
    words = re.split(pattern, text)

    # Join the words with spaces and capitalize the first letter
    result = ' '.join(words).capitalize()

    return result

# Sample Text
sample_text = "RegularExpressionIsAnImportantTopicInPython"

# Call the function and print the result
output = insert_spaces(sample_text)
print(output)
```

```
 regular expression is an important topic in python
```

Question 24-Python regex to find sequences of one upper case letter followed by lower case letters

In [ ]:
```
import re

text = "Abc Def Ghi jKl Mno Pqrs TuV wxYz"
pattern = r'[A-Z][a-z]+'

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

Question 25-Write a Python program to remove continuous duplicate words from Sentence using Regular Expression.

Sample Text:"Hello hello world world" Expected Output: Hello hello world

In [ ]:
```
import re

def remove_continuous_duplicates(sentence):
    # Use a regular expression to find continuous duplicate words
    pattern = r'\b(\w+)\s+\1\b'
    result = re.sub(pattern, r'\1', sentence, flags=re.IGNORECASE)
    return result

sample_text = "Hello hello world world"
output = remove_continuous_duplicates(sample_text)
print(output)
```

In [ ]:
```
Hello hello world
```

Question 27-Write a python program using RegEx to extract the hashtags.

Sample Text: """RT @kapil_kausik: #Doltiwal I mean #xyzabc is "hurt" by #Demonetization as the same has rendered USELESS <U+00A0><U+00BD><U+00B1><U+0089> "acquired funds" No wo""" Expected Output:['#Doltiwal', '#xyzabc', '#Demonetization']

In [ ]:
```
 import re

sample_text = """RT @kapil_kausik: #Doltiwal I mean #xyzabc is "hurt" by #Demor

# Define a regular expression pattern to match hashtags
pattern = r'#\w+'

# Use re.findall() to extract hashtags from the sample text
hashtags = re.findall(pattern, sample_text)

print("Expected Output:", hashtags)
```

In [ ]:
```
Expected Output: ['#Doltiwal', '#xyzabc', '#Demonetization']
```

Question 28- Write a python program using RegEx to remove <U+..> like symbols

Check the below sample text, there are strange symbols something of the sort <U+..> all over the place. You need to come up with a general Regex expression that will cover all such symbols. Sample Text: "@Jags123456 Bharat band on 28??<U+00A0><U+00BD><U+00B8><U+0082>Those who are protesting #demonetization are all different party leaders" Expected Output:@Jags123456 Bharat band on 28??Those who are protesting #demonetization are all different party leaders

In [ ]:
```
import re

def remove_unicode_symbols(text):
    pattern = r'<U\+[A-F0-9]{4}>'
    return re.sub(pattern, '', text)

sample_text = "@Jags123456 Bharat band on 28??<ed><U+00A0><U+00BD><ed><U+00B8><
output = remove_unicode_symbols(sample_text)
print(output)
```

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

Question 29- Write a python program to extract dates from the text stored in the text file.

Sample Text:Ron was born on 12-09-1992 and he was admitted to school 15-12-1999. Note- Store this sample text in the file and then extract dates.

In [ ]:
```
import re

def extract_dates_from_text(filename):
    with open(filename, 'r') as file:
        text = file.read()

    date_pattern = r'\b\d{2}-\d{2}-\d{4}\b'
    dates = re.findall(date_pattern, text)

    return dates

filename = 'sample_text.txt'
dates = extract_dates_from_text(filename)

print("Extracted dates:", dates)
```

Question 30-Create a function inpython to remove all words from a string of length between 2 and 4.

The use of the re.compile() method is mandatory. Sample Text:"The following example creates an ArrayList with a capacity of 50 elements. 4 elements are then added to the ArrayList and the ArrayList is trimmed accordingly." Expected Output:following example creates ArrayList a capacity elements. 4 elements added ArrayListArrayList trimmed accordingly.

In [ ]:
```
import re

def remove_words_between_length(text):
    pattern = re.compile'\b\w{2,4}\b'
    result = pattern.sub('', text)
    result = re.sub(r'\s+', ' ', result)  # Remove extra spaces
    return result.strip()

sample_text = "The following example creates an ArrayList with a capacity of 50
output = remove_words_between_length(sample_text)
print(output)
```

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