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]:
```python
def replace_characters(input_text):
    replacement_chars = [' ', ',', '.']
    output_text = input_text

    for char in replacement_chars:
        output_text = output_text.replace(char, ':')

    return output_text

sample_text = 'Python Exercises, PHP exercises.'
result = replace_characters(sample_text)
print(result)
```

```
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]:
```python
import re

def find_letter_starting_with_a_or_e(input_text):
    letter = re.findall(r'\b[aeAE]\w+\b', input_text)
    return letter

sample_text = "Apples are awesome and elephants eat apples."
outcome = find_letter_starting_with_a_or_e(sample_text)
print(outcome)
```

```
['Apples', 'are', 'awesome', 'and', 'elephants', 'eat', 'apples']
```

Question 3- Create a function in python 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]:
```python
import re

def find_long_letters(input_text):
    pattern = re.compile(r'\b\w{4,}\b')
    long_letters = pattern.findall(input_text)
    return long_letters

sample_text = "This is a sample text with various words of different lengths."
output = find_long_letters(sample_text)
print(output)
```

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

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]:
```python
import re

def find_words_of_length(text, min_length, max_length):
    pattern = re.compile(r'\b\w{' + str(min_length) + ',' + str(max_length) + r'}\b')
    words = pattern.findall(text)
    return words

input_text = "Hello, this is a sample text with words of various lengths like cat, dog, apple, table, and chair."
three_four_five_letter_words = find_words_of_length(input_text, 3, 5)
print(three_four_five_letter_words)
```

```
['Hello', 'this', 'text', 'with', 'words', 'like', 'cat', 'dog', 'apple', 'table', 'and', 'chair']
```

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 github.com Hello Data Science World Data Scientist

In [5]:
```python
import re

def remove_parentheses(strings):
    pattern = re.compile(r'\((.*?)\)')
    output = []

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

    return output

sample_text = ["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello (Data Science World)", "Data (Scientist)"]
outcome = remove_parentheses(sample_text)

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

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

Question 6- Write a python program to remove the parenthesis area from the text stored in the text file using Regular Expression. Sample Text: ["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello (Data Science World)", "Data (Scientist)"] Expected Output: ["example", "hr@fliprobo", "github", "Hello", "Data"] Note- Store given sample text in the text file and then to remove the parenthesis area from the text.

In [6]:
```python
import re

def remove_parentheses(text):
    pattern = re.compile(r'\s*\([^)]*\)')
    fied_text = re.sub(pattern, '', text)
    return fied_text.strip()

input_file_path = "input.txt"
output_file_path = "output.txt"

with open(input_file_path, 'r') as input_file:
    lines = input_file.readlines()

fied_lines = [remove_parentheses(line) for line in lines]

with open(output_file_path, 'w') as output_file:
    for line in fied_lines:
        output_file.write(line + "\n")

print("Modified text has been saved to 'output.txt'.")
```

```
Modified text has been saved to 'output.txt'.
```

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 [7]:
```python
import re

sample_text = "ImportanceOfRegularExpressionsInPython"
split_words = re.findall(r'[A-Z][a-z]*', sample_text)

print(split_words)
```

```
['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: RegularExpression 1IsAn 2ImportantTopic 3InPython

In [8]:
```python
import re

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

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

```
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 [9]:
```python
import re

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

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

```
R egularE xpression1 IsA n2 ImportantT opic3 InP ython
```

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 and alternate email address is xyz.abc@sdomain.domain.com. Please contact us at 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 [13]:
```python
import re

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

input_file_path = "INP.txt.txt"

with open(input_file_path, 'r') as input_file:
    text = input_file.read()

found_addresses = extract_emails(text)

for email in found_addresses:
    print(email)
```

```
xyz@domain.com
xyz.abc@sdomain.domain.com
hr@fliprobo.com
```

In [ ]:
```
Question 11: Match a string that contains only upper and lowercase letters, numbers, and underscores.
```

In [16]:
```python
import re

def match_string(input_string):
    regex_pattern = re.compile(r'^[A-Za-z0-9_]+$')
    if regex_pattern.match(input_string):
        matching_result = "Match found"
    else:
        matching_result = "No match"
    return matching_result

t_string = "Hello_World123"
output_message = match_string(t_string)
print(output_message)
```

```
Match found
```

Question 12: Match a string that starts with a specific number.

In [17]:
```python
import re

def match_start_with_number(input_string, start_number):
    custom_pattern = re.compile(r'^' + re.escape(start_number))
    if custom_pattern.match(input_string):
        matching_output = "Match found"
    else:
        matching_output = "No match"
    return matching_output

input_text = "42Python"
target_number = "42"
final_result = match_start_with_number(input_text, target_number)
print(final_result)
```

```
Match found
```

In [ ]:
```
Question 13: Remove leading zeros from an IP address.
```

In [18]:
```python
def remove_leading_zeros(ip_address):
    address_parts = ip_address.split('.')
    no_zeros_parts = [str(int(part)) for part in address_parts]
    return '.'.join(no_zeros_parts)

input_ip_with_zeros = "192.168.001.001"
ip_without_zeros = remove_leading_zeros(input_ip_with_zeros)
print(ip_without_zeros)
```

```
192.168.1.1
```

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 [21]:
```python
import re

def extract_date(text):
    pattern = re.compile(r'\b(?:January|February|March|April|May|June|July|August|September|October|November|December)\s+\d{1,2}(?:st|nd|rd|tl
    date_match = pattern.search(text)
    if date_match:
        return date_match.group()
    else:
        return "Date not found"

input_file_path = "10_txt.txt"

with open(input_file_path, 'r') as input_file:
    text = input_file.read()

extracted_date = extract_date(text)
print(extracted_date)
```

```
August 15th 1947
```

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 [22]:
```python
def search_words(text, words):
    matching_words = [word for word in words if word in text]
    return matching_words

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

matching_words = search_words(input_text, target_words)
for word in matching_words:
    print(f"'{word}' found in the text.")
```

```
'fox' found in the text.
'dog' 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

In [1]:
```python
def search_string(pattern, text):
    locations = []
    start = 0

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

    return locations

text_to_search = 'The quick brown fox jumps over the lazy dog.'
pattern_to_find = 'fox'

locations = search_string(pattern_to_find, text_to_search)

if locations:
    print(f"The word '{pattern_to_find}' was found at positions: {locations}")
else:
    print(f"The word '{pattern_to_find}' was not found in the text.")
```

```
The word 'fox' was found at positions: [16]
```

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

In [2]:
```python
import re

def find_substrings(custom_pattern, custom_text):
    substrings = re.findall(custom_pattern, custom_text)
    return substrings

custom_text = 'Python exercises, PHP exercises, C# exercises'
custom_pattern = 'exercises'

substrings = find_substrings(custom_pattern, custom_text)

if substrings:
    print(f"Substrings '{custom_pattern}' found: {substrings}")
else:
    print(f"No substrings '{custom_pattern}' found in the text.")
```

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

Question 18- Write a Python program to find the occurrence and position of the substrings within a string.

In [3]:
```python
import re

def find_occurrences(custom_pattern, custom_text):
    occurrences = []
    start = 0

    while start < len(custom_text):
        index = custom_text.find(custom_pattern, start)
        if index == -1:
            break
        occurrences.append((custom_pattern, index))
        start = index + 1

    return occurrences

custom_text = 'Python exercises, PHP exercises, C# exercises'
custom_pattern = 'exercises'

occurrences = find_occurrences(custom_pattern, custom_text)

if occurrences:
    print("Occurrences of substrings:")
    for pattern, index in occurrences:
        print(f"'{pattern}' found at position {index}")
else:
    print(f"No occurrences of substrings '{custom_pattern}' found in the text.")
```

```
Occurrences of substrings:
'exercises' found at position 7
'exercises' found at position 22
'exercises' found at position 36
```

Question 19- Write a Python program to convert a date of yyyy-mm-dd format to dd-mm-yyyy format.

In [4]:
```python
def convert_date_format(date):
    parts = date.split('-')
    if len(parts) != 3:
        return "Invalid date format"
    return f"{parts[2]}-{parts[1]}-{parts[0]}"

date_in_yyyy_mm_dd = "2023-08-26"
date_in_dd_mm_yyyy = convert_date_format(date_in_yyyy_mm_dd)
print(f"Original date: {date_in_yyyy_mm_dd}")
print(f"Converted date: {date_in_dd_mm_yyyy}")
```

```
Original date: 2023-08-26
Converted date: 26-08-2023
```

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. Sample Text: "01.12 0132.123 2.31875 145.8 3.01 27.25 0.25" Expected Output: ['01.12', '145.8', '3.01', '27.25', '0.25']

In [9]:
```python
import re

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

sample_text = "01.12 0132.123 2.31875 145.8 3.01 27.25 0.25"
decimal_numbers = find_decimal_numbers(sample_text)
print("Decimal numbers with precision of 1 or 2:", decimal_numbers)
```

```
Decimal numbers with precision of 1 or 2: ['01.12', '145.8', '3.01', '27.25', '0.25']
```

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

In [12]:
```python
import re

def separate_and_print_numbers(text):
    custom_pattern = re.compile(r'\d+')
    numbers_with_positions = [(match.group(), match.start()) for match in custom_pattern.finditer(text)]
    return numbers_with_positions

custom_string = "abc123xyz456pqr789"
numbers_with_positions = separate_and_print_numbers(custom_string)

print("Numbers and their positions:")
for number, position in numbers_with_positions:
    print(f"Number: {number}, Position: {position}")
```

```
Numbers and their positions:
Number: 123, Position: 3
Number: 456, Position: 9
Number: 789, Position: 15
```

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 [13]:
```python
import re

sample_text = 'My marks in each semester are: 947, 896, 926, 524, 734, 950, 642'
numeric_values = re.findall(r'\d+', sample_text)
max_numeric_value = max(map(int, numeric_values))
print("Maximum numeric value:", max_numeric_value)
```

```
Maximum numeric value: 950
```

Question 23- Create a function in python to insert spaces between words starting with capital letters. Sample Text: “RegularExpressionIsAnImportantTopicInPython" Expected Output: Regular Expression Is An Important Topic In Python

In [14]:
```python
import re

def insert_spaces(custom_text):
    spaced_text = re.sub(r'(?<=.)([A-Z])', r' \1', custom_text)
    return spaced_text

custom_input = "RegularExpressionIsAnImportantTopicInPython"
custom_result = insert_spaces(custom_input)
print("Inserted spaces:", custom_result)
```

```
Inserted spaces: Regular Expression Is An Important Topic In Python
```

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

In [15]:
```python
import re

custom_text = "Aa Bb Cd Ee Ef Gg Hh"
uppercase_lowercase_sequences = re.findall(r'[A-Z][a-z]+', custom_text)
print("Uppercase followed by lowercase sequences:", uppercase_lowercase_sequences)
```

```
Uppercase followed by lowercase sequences: ['Aa', 'Bb', 'Cd', 'Ee', 'Ef', 'Gg', 'Hh']
```

Question 25- Write a Python program to remove continuous duplicate words from Sentence using Regular Expression. Sample Text: "Hello hello world world"

In [19]:
```python
import re

def remove_continuous_duplicates(text):
    text = re.sub(r'\b(\w+)( \1)+\b', r'\1', text)
    return text

sample_text = "Hello hello world world"
text = remove_continuous_duplicates(sample_text)
print(" text:", text)
```

```
 text: Hello hello world
```

In [ ]:
```
Question 26-  Write a python program using RegEx to accept string ending with alphanumeric character.
```

In [21]:
```python
import re

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


different_test_strings = ["HelloWorld1", "Regex1234", "Testing123", "456Test", "!Symbol$", "123456"]

for test_string in different_test_strings:
    if is_valid_string(test_string):
        print(f"'{test_string}' is valid.")
    else:
        print(f"'{test_string}' is not valid.")
```

```
'HelloWorld1' is valid.
'Regex1234' is valid.
'Testing123' is valid.
'456Test' is valid.
'!Symbol$' is not valid.
'123456' is valid.
```

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 [22]:
```python
import re

sample_text = """RT @kapil_kausik: #Doltiwal I mean #xyzabc is "hurt" by #Demonetization as the same has rendered USELESS <ed><U+00A0><U+00BD>

hashtags = re.findall(r'#\w+', sample_text)

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

```
Extracted hashtags: ['#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 [23]:
```python
import re

sample_text = "@Jags123456 Bharat band on 28??<ed><U+00A0><U+00BD><ed><U+00B8><U+0082>Those who  are protesting #demonetization  are all diffe

new_text = re.sub(r'<U\+[A-Fa-f0-9]+>', '', sample_text)

print("New text:", new_text)
```

```
New text: @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 [1]:
```python
import re

with open('date.txt', 'r') as file:
    sample_text = file.read()

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

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

```
Extracted dates: ['12-09-1992', '15-12-1999']
```

Question 30- Create a function in python 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 ArrayList ArrayList trimmed accordingly.

In [2]:
```python
import re

def remove_words_between_length(input_text):
    custom_pattern = re.compile(r'\b\w{2,4}\b')
    new_modified_text = re.sub(custom_pattern, '', input_text)
    new_modified_text = re.sub(r'\s+', ' ', new_modified_text).strip()
    return new_modified_text

sample_text = "The following example creates an ArrayList with a capacity of 50 elements. 4 elements are then added to the ArrayList and the /
new_output = remove_words_between_length(sample_text)
print(new_output)
```

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

In [ ]: