Question 1- Write a Python program to check that a string contains only a certain set of characters (in this case a-z, A-Z and 0-9).

In [23]:
```
import re
```

In [4]:
```
def character(string):
    pattern = r'^[a-zA-Z0-9]+$'
    if re.match(pattern, string):
        print("Statement is True")
    else:
        print("Statement is False")

string1 = "Hello123"
character(string1)

string2 = "Hello_123"
character(string2)
```

```
Statement is True
Statement is False
```

Question 2- Create a function in python that matches a string that has an a followed by zero or more b's

In [12]:
```
def match(string):
    pattern = r'^ab*$'
    if re.match(pattern, string):
        return True
    else:
        return False

strings = ["a", "ab", "a00bb", "abbb", "af", "abccc"]
for string in strings:
    if match(string):
        print(f"{string} matches the pattern.")
    else:
        print(f"{string} does not match the pattern.")
```

```
a matches the pattern.
ab matches the pattern.
a00bb does not match the pattern.
abbb matches the pattern.
af does not match the pattern.
abccc does not match the pattern.
```

Question 3- Create a function in python that matches a string that has an a followed by one or more b's

In [13]:
```
def match(string):
    pattern = r'^ab+$'
    if re.match(pattern, string):
        return True
    else:
        return False

strings = ["a", "ab", "abb", "abbc", "ac", "abccc"]
for string in strings:
    if match(string):
        print(f"{string} matches the pattern.")
    else:
        print(f"{string} does not match the pattern.")
```

```
a does not match the pattern.
ab matches the pattern.
```

```
abb matches the pattern.
abbc does not match the pattern.
ac does not match the pattern.
abccc does not match the pattern.
```

Question 4- Create a function in Python and use RegEx that matches a string that has an a followed by zero or one 'b'.

In [14]:
```python
def match(string):
    pattern = r'^ab?$'
    if re.match(pattern, string):
        return True
    else:
        return False

strings = ["a", "ab", "abb", "abbc", "ac", "abccc"]
for string in strings:
    if match(string):
        print(f"{string} matches the pattern.")
    else:
        print(f"{string} does not match the pattern.")
```

```
a matches the pattern.
ab matches the pattern.
abb does not match the pattern.
abbc does not match the pattern.
ac does not match the pattern.
abccc does not match the pattern.
```

Question 5- Write a Python program that matches a string that has an a followed by three 'b'.

In [16]:
```python
def match(string):
    pattern = r'^abbb$'
    if re.match(pattern, string):
        return True
    else:
        return False

strings = ["a", "ab", "abbb", "abbc", "accc", "abccc"]
for string in strings:
    if match(string):
        print(f"{string} matches the pattern.")
    else:
        print(f"{string} does not match the pattern.")
```

```
a does not match the pattern.
ab does not match the pattern.
abbb matches the pattern.
abbc does not match the pattern.
accc does not match the pattern.
abccc does not match the pattern.
```

Question 6- Write a regular expression in Python to split a string into uppercase letters.

In [24]:
```python
def split(string):
    pattern = r'(?=[A-Z])'
    split_list = re.split(pattern, input_string)
    return split_list

string = "ImportanceOfRegularExpressionsInPython"
Splits=split(string)
print(Splits)
```

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

Question 7- Write a Python program that matches a string that has an a followed by two to three 'b'.

In [11]:
```
def match_pattern(string):
    pattern = r'a{1}b{2,3}'
    match = re.search(pattern, string)
    if match:
        return True
    else:
        return False

test_string = 'abb'
if match_pattern(test_string):
    print("Match found!")
else:
    print("No match found.")
```

```
Match found!
```

Question 8- Write a Python program to find sequences of lowercase letters joined with a underscore.

In [15]:
```
def find_sequences(test):
    pattern = r'[a-z]+_[a-z]+'
    sequences = re.findall(pattern, test)
    return sequences

test = 'my_dog_name_is_lisa_girl'
sequences = find_sequences(test)
print(sequences)
```

```
['my_dog', 'name_is', 'lisa_girl']
```

Question 9- Write a Python program that matches a string that has an 'a' followed by anything, ending in 'b'

In [18]:
```
def match_string(string):
    pattern = r'a.*b$'
    match = re.search(pattern, string)
    if match:
        return True
    else:
        return False


strings = ["acb", "azb", "a1234b", "abcd", "a1234"]
for string in strings:
    if match_string(string):
        print(f"{string} matches the pattern.")
    else:
        print(f"{string} does not match the pattern.")
```

```
acb matches the pattern.
azb matches the pattern.
a1234b matches the pattern.
abcd does not match the pattern.
a1234 does not match the pattern.
```

Question 10- Write a Python program that matches a word at the beginning of a string.

In [3]:
```
def match_word(string, word):
    pattern = r'\b' + re.escape(word)
    match = re.search(pattern, string)
    return match is not None


string = "Data is everything."
```

```
word = "Data"

if match_word(string, word):
    print("The word '{}' is at the beginning of the string.".format(word))
else:
    print("The word '{}' is not at the beginning of the string.".format(word))
```

```
The word 'Data' is at the beginning of the string.
```

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

In [5]:
```
def match_characters(string):
    pattern = r'^[a-zA-Z0-9_]+$'
    match = re.match(pattern, string)
    return match is not None

strings = ["Data123", "data_analyising", "Excel!!", "BI_123", "OpenAI", "123456", "_"]

for string in strings:
    if match_characters(string):
        print("The string '{}' contains only allowed characters.".format(string))
    else:
        print("The string '{}' contains disallowed characters.".format(string))
```

```
The string 'Data123' contains only allowed characters.
The string 'data_analyising' contains only allowed characters.
The string 'Excel!!' contains disallowed characters.
The string 'BI_123' contains only allowed characters.
The string 'OpenAI' contains only allowed characters.
The string '123456' contains only allowed characters.
The string '_' contains only allowed characters.
```

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

In [13]:
```
def match_start_with_number(string, number):
    pattern = r'^' + re.escape(str(number))
    match = re.match(pattern, string)
    return match is not None

string_a = "12Data"
string_b = "35world"
number_a = 12
number_b = 35

if match_start_with_number(string_a, number_a):
    print("The string '{}' starts with the number {}.".format(string_a, number_a))
else:
    print("The string '{}' does not start with the number {}.".format(string_a, number_a

if match_start_with_number(string_b, number_b):
    print("The string '{}' starts with the number {}.".format(string_b, number_b))
else:
    print("The string '{}' does not start with the number {}.".format(string_b, number_b
```

```
The string '12Data' starts with the number 12.
The string '35world' starts with the number 35.
```

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

In [14]:
```
def remove_leading_zeros(ip_address):
    pattern = r'\b0+(\d+)'
    updated_address = re.sub(pattern, r'\1', ip_address)
    return updated_address
```

```
ip_address = "192.168.031.001"
updated_address = remove_leading_zeros(ip_address)
print("Original IP address: {}".format(ip_address))
print("Updated IP address: {}".format(updated_address))
```

```
Original IP address: 192.168.031.001
Updated IP address: 192.168.31.1
```

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.

In [34]:
```
def match_date_string(text):
    pattern = r'([A-Za-z]+)\s+(\d{1,2}),\s+(\d{4})'
    matches = re.findall(pattern, text)
    return matches

# Example usage
file_path = "test.txt"

with open(file_path, "r") as file:
    file_content = file.read()

date_strings = match_date_string(file_content)
for date in date_strings:
    print("Found date: {} {}, {}".format(date[0], date[1], date[2]))
```

```
Found date: August 15, 1947
```

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

In [35]:
```
import re

def search_literals(string, literals):
    found_literals = []
    for literal in literals:
        pattern = re.escape(literal)
        match = re.search(pattern, string)
        if match:
            found_literals.append(literal)
    return found_literals

text = 'The quick brown fox jumps over the lazy dog.'
search_list = ['fox']

found_literals = search_literals(text, search_list)
print("Found literals:", found_literals)
```

```
Found literals: ['fox']
```

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 [36]:
```
def search_literals(string, literals):
    found_literals = []
    for literal in literals:
        pattern = re.escape(literal)
        matches = re.finditer(pattern, string)
        for match in matches:
            found_literals.append((literal, match.start()))
    return found_literals


text = 'The quick brown fox jumps over the lazy dog.'
search_list = ['fox']
```

```
found_literals = search_literals(text, search_list)
print("Found literals:")
for literal, location in found_literals:
    print("Literal: '{}' at location: {}".format(literal, location))
```

```
Found literals:
Literal: 'fox' at location: 16
```

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

In [37]:
```
def find_substrings(string, pattern):
    matches = re.findall(pattern, string)
    return matches

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

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

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

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

In [38]:
```
def find_substrings(string, pattern):
    matches = re.finditer(pattern, string)
    substrings = []
    for match in matches:
        substring = match.group()
        start = match.start()
        end = match.end()
        substrings.append((substring, start, end))
    return substrings

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

substrings = find_substrings(text, search_pattern)
print("Substrings found:")
for substring, start, end in substrings:
    print("Substring: '{}', Position: {}-{}".format(substring, start, end))
```

```
Substrings found:
Substring: 'exercises', Position: 7-16
Substring: 'exercises', Position: 22-31
Substring: 'exercises', Position: 36-45
```

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

In [39]:
```
def convert_date(date):
    pattern = r'(\d{4})-(\d{2})-(\d{2})'
    match = re.match(pattern, date)
    if match:
        year = match.group(1)
        month = match.group(2)
        day = match.group(3)
        converted_date = '{}-{}-{}'.format(day, month, year)
        return converted_date
    else:
        return 'Invalid date format'

date = '1993-09-16'
converted_date = convert_date(date)
print("Converted date:", converted_date)
```

```
Converted date: 16-09-1993
```

Question 20- Write a Python program to find all words starting with 'a' or 'e' in a given string.

In [43]:
```python
def find_words_starting_with_a_or_e(string):
    pattern = r'\b[aAeE]\w+'
    matches = re.findall(pattern, string)
    return matches

text = "Ant and an elephant entered the room."

words = find_words_starting_with_a_or_e(text)
print("Words starting with 'a' or 'e':", words)
```

```
Words starting with 'a' or 'e': ['Ant', 'and', 'an', 'elephant', 'entered']
```

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

In [44]:
```python
def separate_numbers_with_positions(string):
    pattern = r'\d+'
    matches = re.finditer(pattern, string)
    numbers = []
    for match in matches:
        number = match.group()
        start = match.start()
        end = match.end()
        numbers.append((number, start, end))
    return numbers

text = "I have 1 car, 2 bike, and 1 dog."

numbers = separate_numbers_with_positions(text)
print("Numbers and their positions:")
for number, start, end in numbers:
    print("Number: {}, Position: {}-{}".format(number, start, end))
```

```
Numbers and their positions:
Number: 1, Position: 7-8
Number: 2, Position: 14-15
Number: 1, Position: 26-27
```

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

In [45]:
```python
def extract_max_numeric_value(string):
    pattern = r'\d+'
    matches = re.findall(pattern, string)
    if matches:
        max_value = max(map(int, matches))
        return max_value
    else:
        return None

text = "The maximum value is 99, followed by 42 and 78."

max_value = extract_max_numeric_value(text)
if max_value is not None:
    print("Maximum numeric value:", max_value)
else:
    print("No numeric value found in the string.")
```

```
Maximum numeric value: 99
```

Question 23- Write a Regex in Python to put spaces between words starting with capital letters

In [46]:
```python
def add_spaces_between_capital_words(string):
    pattern = r'([A-Z][a-z]+)'
    modified_string = re.sub(pattern, r' \1', string)
    return modified_string

text = "IHaveSomethingToSay"

modified_text = add_spaces_between_capital_words(text)
print("Modified text:", modified_text)
```

```
Modified text: I Have Something To Say
```

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

In [48]:
```
def find_uppercase_sequences(string):
    pattern = r'[A-Z][a-z]+'
    matches = re.findall(pattern, string)
    return matches

text = "The Dog Bark By Seeing a Cat."

sequences = find_uppercase_sequences(text)
print("Uppercase sequences followed by lowercase letters:")
for sequence in sequences:
    print(sequence)
```

```
Uppercase sequences followed by lowercase letters:
The
Dog
Bark
By
Seeing
Cat
```

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

In [49]:
```
def remove_duplicate_words(sentence):
    pattern = r'\b(\w+)\b\s+(?=.*\b\1\b)'
    modified_sentence = re.sub(pattern, '', sentence)
    return modified_sentence

sentence = "The Dog Dog Bark By Seeing a Cat Dog Bark."

modified_sentence = remove_duplicate_words(sentence)
print("Modified sentence:", modified_sentence)
```

```
Modified sentence: The By Seeing a Cat Dog Bark.
```

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

In [50]:
```
def is_string_ending_with_alphanumeric(string):
    pattern = r'^.*[a-zA-Z0-9]$'
    match = re.match(pattern, string)
    return match is not None

strings = ["Data123", "data_analyising", "Excel!!", "BI_123", "OpenAI", "123456"]

for string in strings:
    if is_string_ending_with_alphanumeric(string):
        print("String '{}' ends with an alphanumeric character.".format(string))
    else:
        print("String '{}' does not end with an alphanumeric character.".format(string))
```

```
String 'Data123' ends with an alphanumeric character.
String 'data_analyising' ends with an alphanumeric character.
String 'Excel!!' does not end with an alphanumeric character.
String 'BI_123' ends with an alphanumeric character.
String 'OpenAI' ends with an alphanumeric character.
String '123456' ends with an alphanumeric character.
```

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

In [51]:
```
def extract_hashtags(string):
    pattern = r'\#\w+'
    hashtags = re.findall(pattern, string)
    return hashtags

text = """RT @kapil_kausik: #Doltiwal I mean #xyzabc is "hurt" by #Demonetization as the

hashtags = extract_hashtags(text)
print("Extracted hashtags:", hashtags)
```

```
Extracted hashtags: ['#Doltiwal', '#xyzabc', '#Demonetization']
```

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

In [52]:
```
def remove_symbols(string):
    pattern = r'<U\+[A-Fa-f0-9]+>'
    modified_string = re.sub(pattern, '', string)
    return modified_string

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

modified_text = remove_symbols(text)
print("Modified text:", modified_text)
```

```
Modified text: @Jags123456 Bharat band on 28??<ed><ed>Those who  are protesting #demonet
ization  are all different party leaders
```

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

In [59]:
```
def extract_dates_from_file(file_path):
    pattern = r'\b\d{1,2}/\d{1,2}/\d{4}\b'
    dates = []

    with open(file_path, 'r') as file:
        file_content = file.read()
        matches = re.findall(pattern, file_content)
        dates.extend(matches)

    return dates

file_path = "sample.txt"

with open(file_path, "r") as file:
    file_content = file.read()

extracted_dates = extract_dates_from_file(file_path)
print("Extracted dates:")
for date in extracted_dates:
    print(date)
```

```
Extracted dates:
12/09/1992
15/2/1999
```

Question 30- Write a Python program to replace all occurrences of a space, comma, or dot with a colon.

In [60]:
```
def replace_space_comma_dot_with_colon(text):
    pattern = r'[ ,.]'
    modified_text = re.sub(pattern, ':', text)
    return modified_text

sample_text = 'Python Exercises, PHP exercises.'

modified_text = replace_space_comma_dot_with_colon(sample_text)
print("Modified text:", modified_text)
```

```
Modified text: Python:Exercises::PHP:exercises:
```