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 [7]:
```python
import re



def test(check_str):
    pattern = r'^[a-zA-Z0-9]+$'
    if re.match(pattern, check_str):
        print('accepted:', check_str)
    else:
        print('not accepted:', check_str)

test(check_str='abcdefgh')
test(check_str='ABCDEFGH0123')
test(check_str='ABCDE123567!')
```

```
accepted: abcdefgh
accepted: ABCDEFGH0123
not accepted: ABCDE123567!
```

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

In [9]:
```python
def match_pattern(string):
    pattern = r'^ab*$'

    if re.match(pattern, string):
        print("String matches the pattern")
    else:
        print("String does not match the pattern")

match_pattern("ab")
match_pattern("abc")
match_pattern("abb")
match_pattern("a")
```

```
String matches the pattern
String does not match the pattern
String matches the pattern
String matches the pattern
```

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

In [16]:
```python
import re

def match_pattern(string):
    pattern = r'^ab+$'

    if re.match(pattern, string):
        print(f"String '{string}' matches the pattern")
    else:
        print(f"String '{string}' does not match the pattern")

match_pattern("ab")
match_pattern("abc")
match_pattern("abb")
match_pattern("a")
```

```
String 'ab' matches the pattern
String 'abc' does not match the pattern
String 'abb' matches the pattern
String 'a' does not match the pattern
```

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

In [19]:
```python
import re
def match_pattern(string):
        pattern= r'^ab?$'
        if re.match(pattern,string):
            print(f"String '{string}'matches the pattern")
        else:
                print(f"String'{string}'does not matches the pattern")
match_pattern("ab")
match_pattern("abb")
match_pattern("abc")
match_pattern("a")
```

```
String 'ab'matches the pattern
String'abb'does not matches the pattern
String'abc'does not matches the pattern
String 'a'matches the pattern
```

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

In [22]:
```python
import re
def match_pattern(string):
    pattern = r'^ab{3}$'
    if  re.match(pattern,string):
        print(f"String '{string}'matches the pattern")
    else:
        print(f"String '{string}'does not matches the pattern")
match_pattern("abc")
match_pattern("abbbb")
match_pattern("ab")
match_pattern("abb")
match_pattern("abbb")
```

```
String 'abc'does not matches the pattern
String 'abbbb'does not matches the pattern
String 'ab'does not matches the pattern
String 'abb'does not matches the pattern
String 'abbb'matches the pattern
```

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

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

In [23]:
```python
import re

text="mportanceOfRegularExpressionsInPython"

result= re.split(r'(?=[A-Z])',text)
print(result)
```

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

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

In [27]:
```python
import re
def match_pattern(string):
    pattern = r'^ab{3,}/$'
    if re.match(pattern,string):
        print(f"String '{string}'matches the pattern")
    else:
        print(f"String '{string}'does not matches the pattern")
match_pattern("bb")
match_pattern("bbb")
match_pattern("b")
```

```
String 'bb'does not matches the pattern
String 'bbb'does not matches the pattern
String 'b'does not matches the pattern
```

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

In [28]:
```python
import re

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


text = "this_is_a_test_string_with_sequences_like_this_one"
result = find_sequences(text)
print(result)
```

```
['this_is_a_test_string_with_sequences_like_this_one']
```

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

In [39]:
```python
import re

def match_pattern(string):
    pattern = r'^a.*b$'

    if re.match(pattern, string):
        print(f"String '{string}' matches the pattern.")
    else:
        print(f"String '{string}' does not match the pattern.")


match_pattern("abc")
match_pattern("adeb")
match_pattern("ab")
match_pattern("axxxxxb")
```

```
String 'abc' does not match the pattern.
String 'adeb' matches the pattern.
String 'ab' matches the pattern.
String 'axxxxxb' matches the pattern.
```

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

In [30]:
```python
import re

def match_word_at_beginning(string, word):
    pattern = fr'^{word}\b'

    if re.match(pattern, string):
        print(f"The word '{word}' is at the beginning of the string.")
    else:
        print(f"The word '{word}' is not at the beginning of the string.")

# Test cases
match_word_at_beginning("Hello world", "Hello")
match_word_at_beginning("Python is great", "is")
match_word_at_beginning("Programming is fun", "fun")
match_word_at_beginning("Hello, how are you?", "how")
```

```
The word 'Hello' is at the beginning of the string.
The word 'is' is not at the beginning of the string.
The word 'fun' is not at the beginning of the string.
The word 'how' is not at the beginning of the string.
```

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

In [31]:
```python
import re

def match_pattern(string):
    pattern = r'^[A-Za-z0-9_]+$'

    if re.match(pattern, string):
        print(f"String '{string}' matches the pattern.")
    else:
        print(f"String '{string}' does not match the pattern.")

# Test cases
match_pattern("PK_1133")
match_pattern("Abc_de_velliers_456")
match_pattern("Invalid@String")
match_pattern("Upercaseletters")
```

```
String 'PK_1133' matches the pattern.
String 'Abc_de_velliers_456' matches the pattern.
String 'Invalid@String' does not match the pattern.
String 'Upercaseletters' matches the pattern.
```

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

In [38]:
```python
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}.")


starts_with_number("3344abc", 123)
starts_with_number("456xyz", 123)
starts_with_number("789def", 789)
starts_with_number("abc123", 989)
```

```
The string '3344abc' does not start with the number 123.
The string '456xyz' does not start with the number 123.
The string '789def' starts with the number 789.
The string 'abc123' does not start with the number 989.
```

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

In [40]:
```python
def remove_leading_zeros(ip_address):
    components = ip_address.split('.')
    sanitized_components = [str(int(component)) for component in components]
    sanitized_ip_address = '.'.join(sanitized_components)
    return sanitized_ip_address

ip_address = "192.168.001.001"
sanitized_ip = remove_leading_zeros(ip_address)
print(sanitized_ip)
```

```
192.168.1.1
```

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 [37]:
```python
import re

pattern = r"\b[A-Za-z]+\s\d+,\s\d{4}\b"


text = "Today is July 15, 2023. Tomorrow will be July 16, 2023."
matches = re.findall(pattern, text)
print(matches)
```

```
['July 15, 2023', 'July 16, 2023']
```

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

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

In [41]:
```python
def search_literals(text, searched_words):
    found_words = []
    for word in searched_words:
        if word in text:
            found_words.append(word)
    return found_words

text = 'The quick brown fox jumps over the lazy dog.'
searched_words = ['cow', 'dog', 'cat']
result = search_literals(text, searched_words)
print(result)
```

```
['dog']
```

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 [42]:
```python
import re

def search_string_locations(text, pattern):
    locations = []
    regex_pattern = re.compile(re.escape(pattern))
    matches = regex_pattern.finditer(text)
    for match in matches:
        start = match.start()
        end = match.end()
        locations.append((start, end))
    return locations


text = 'The quick brown fox jumps over the lazy dog.'
pattern = 'fox'
result = search_string_locations(text, pattern)
print(result)
```

```
[(16, 19)]
```

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

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

In [43]:
```python
def find_substrings_3(text, pattern):
    substrings = [substring for substring in text.split(pattern)]
    return substrings

text = 'Python exercises, SQL exercises, Pandas exercises'
pattern = 'exercises'
result = find_substrings_3(text, pattern)
print(result)
```

```
['Python ', ', SQL ', ', Pandas ', '']
```

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

In [45]:
```python
import re

def find_occurrences_2(text, substring):
    occurrences = [(match.group(), match.start()) for match in re.finditer(substring, text)]
    return occurrences

text = 'Data scienceis great and fun  is easy to learn'
substring = 'Data Science'
result = find_occurrences_2(text, substring)
print(result)
```

```
[]
```

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

In [47]:
```python
def convert_date_5(date):
    formatted_date = date.replace("-", "/").replace("/", "-", 2)
    return formatted_date

date = "2023-07-16"
result = convert_date_5(date)
print(result)
```

```
2023-07-16
```

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

In [48]:
```python
import re

def find_words_2(text):
    words = re.findall(r'\b[ae]\w+', text)
    return words

text = 'johncena the rock batista edge randy orton the og of wwe'
result = find_words_2(text)
print(result)
```

```
['edge']
```

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

In [5]:
```python
def separate_number2(text):
    result = [(char, i) for i, char in enumerate(text) if char.isdigit()]
    return result

text = 'fed 123 def 456 ehi 789'
result = separate_numbers_2(text)
for number, position in result:
    print(f"Number: {number}, Position: {position}")
```

```
Number: 1, Position: 4
Number: 2, Position: 5
Number: 3, Position: 6
Number: 4, Position: 12
Number: 5, Position: 13
Number: 6, Position: 14
Number: 7, Position: 20
Number: 8, Position: 21
Number: 9, Position: 22
```

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

In [8]:
```python
import re

def extract_maximum_numeric(txt):
    pattern = r'\d+'
    numbers=re.findall(pattern,txt)
    max_number = max(map(int,numbers))
    return max_number
txt= 'abc 123 def 567 ghi 789 jkl 890 '
max_value = extract_maximum_numeric(txt)
print(f"Maximum numeric value :  { max_value}")
```

```
Maximum numeric value :  890
```

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

In [ ]:
```python
import re

def words_starting_with_capital(string):

    pattern = r'(?<=\w)([A-Z])'
    updated_text = re.sub(pattern, r' \1', text)
    return updated_text
text = "ThisIsARegularExpression"
InternshipAtFlipRoboTECh"
result = add_spaces(text)
print(result)
'
```

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

In [ ]:
```python
import re

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

text = "The dog is running late and slow"
result = find_sequences(text)
print(result)
```

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

In [ ]:
```python
import re

def remove_duplicates_3(sentence):
    pattern = r'\b(?P<word>\w+)\b\s+(?=\b(?P=word)\b)'
    result = re.sub(pattern, '', sentence)
    return result

sentence = "This is data science ."
result = remove_duplicates_3(sentence)
print(result)
```

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

In [ ]:
```python
import re

def string_ending_alphanumeric_5(text):
    pattern = r'\w$'
    result = re.match(pattern, text)
    return result is not None
```

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

Sample Text: 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""" Output: ['#Doltiwal', '#xyzabc', '#Demonetization']

In [ ]:
```python
import re

def extract_hashtags(text):
    pattern = r'#[A-Za-z0-9_]+'
    hashtags = re.findall(pattern, text)
    return hashtags

text = 'pK @Rohan_Prasenjeet: #BAlti I mean #xyzabc is "gone" by #Torrando as the same has happened multiple time <ed><U+00A0><U+00BD><ed><U+00B1><U+0089> "acquired funds" No wo'
hashtags = extract_hashtags(text)
print(hashtags)
```

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" Output: @Jags123456 Bharat band on 28??Those who are protesting #demonetization are all different party leaders

In [ ]:
```python
import re

def remove_symbols_2(text):
    pattern = r'<U\+\w{4}>'
    result = re.sub(pattern, '', text)
    return result

text = "@PK123456 liq band on 28??<ed><U+00A0><U+00BD><ed><U+00B8><U+0082>Those who are wanting #liquorfree are all different humans"
result = remove_symbols_2(text)
print(result)
```

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. Store this sample text in the file and then extract dates.

In [ ]:
```python
import re

def extract_dates_2(file_path):
    with open(file_path, 'r') as file:
        text = file.read()
        pattern = r'\d{2}-\d{2}-\d{4}'
        match = re.search(pattern, text)
        dates = [match.group()] if match else []
        return dates

file_path = 'sample_text.txt'
dates = extract_dates_2(file_path)
print(dates)
```

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

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

In [ ]:
```python
def replace_symbols_7(text):
    words = text.split()
    result = ':'.join(words)
    return result
```