Task 3 (1.10.2023)

October 9, 2023

[1]:
```
#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:
```

[2]:
```
import re
Sample_Text = 'Python Exercises, PHP exercises.'
pattern = "[,. ]"
result= re.sub(pattern,":",Sample_Text)
result
```

[2]:
```
'Python:Exercises::PHP:exercises:'
```

[3]:
```
#Question 2-  Write a Python program to find all words starting with 'a' or 'e'␣
 ↪in a given string.
```

[4]:
```
string='We all fear change and yet change is the very essence of life. How does␣
 ↪a stream cross the mighty, desolate desert? Can it allow itself to change␣
 ↪its very form to survive the journey? The Tale of the Sands helps the reader␣
 ↪absorb the lesson that change can transform us, if only we have the courage␣
 ↪to embrace it. The only characters in this book are a beautiful, flowing␣
 ↪stream, a mighty, dry desert, the wind and the voice of faith that lives␣
 ↪within us all. As the stream confronts its many fears of being subsumed by␣
 ↪the wind, the voice reminds us that change allows us to grow, move and␣
 ↪prosper.'

result=re.findall(r'\b[a|e]\w*\b', string)
print(result)
```

```
['all', 'and', 'essence', 'a', 'allow', 'absorb', 'embrace', 'are', 'a', 'a',
'and', 'all', 'allows', 'and']
```

[5]:
```
#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.
```

[6]:
```
string='We all fear change and yet change is the very essence of life. How does␣
 ↪a stream cross the mighty, desolate desert? Can it allow itself to change␣
 ↪its very form to survive the journey? The Tale of the Sands helps the reader␣
 ↪absorb the lesson that change can transform us, if only we have the courage␣
 ↪to embrace it. The only characters in this book are a beautiful, flowing␣
 ↪stream, a mighty, dry desert, the wind and the voice of faith that lives␣
 ↪within us all. As the stream confronts its many fears of being subsumed by␣
 ↪the wind, the voice reminds us that change allows us to grow, move and␣
 ↪prosper.'
compile=re.compile(r'\b\w{4,}\b')
result=compile.findall(string)
print(result)
```

```
['fear', 'change', 'change', 'very', 'essence', 'life', 'does', 'stream',
'cross', 'mighty', 'desolate', 'desert', 'allow', 'itself', 'change', 'very',
'form', 'survive', 'journey', 'Tale', 'Sands', 'helps', 'reader', 'absorb',
'lesson', 'that', 'change', 'transform', 'only', 'have', 'courage', 'embrace',
'only', 'characters', 'this', 'book', 'beautiful', 'flowing', 'stream',
'mighty', 'desert', 'wind', 'voice', 'faith', 'that', 'lives', 'within',
'stream', 'confronts', 'many', 'fears', 'being', 'subsumed', 'wind', 'voice',
'reminds', 'that', 'change', 'allows', 'grow', 'move', 'prosper']
```

[7]:
```
#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.
```

[8]:
```
string='We all fear change and yet change is the very essence of life. How does␣
 ↪a stream cross the mighty, desolate desert? Can it allow itself to change␣
 ↪its very form to survive the journey? The Tale of the Sands helps the reader␣
 ↪absorb the lesson that change can transform us, if only we have the courage␣
 ↪to embrace it. The only characters in this book are a beautiful, flowing␣
 ↪stream, a mighty, dry desert, the wind and the voice of faith that lives␣
 ↪within us all. As the stream confronts its many fears of being subsumed by␣
 ↪the wind, the voice reminds us that change allows us to grow, move and␣
 ↪prosper.'
compile=re.compile(r'\b\w{3,5}\b')
result=compile.findall(string)
print(result)
```

```
['all', 'fear', 'and', 'yet', 'the', 'very', 'life', 'How', 'does', 'cross',
'the', 'Can', 'allow', 'its', 'very', 'form', 'the', 'The', 'Tale', 'the',
'Sands', 'helps', 'the', 'the', 'that', 'can', 'only', 'have', 'the', 'The',
'only', 'this', 'book', 'are', 'dry', 'the', 'wind', 'and', 'the', 'voice',
'faith', 'that', 'lives', 'all', 'the', 'its', 'many', 'fears', 'being', 'the',
'wind', 'the', 'voice', 'that', 'grow', 'move', 'and']
```

[9]:
```
#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

[10]:
```
sample_text=["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello␣
 ↪(Data Science World)", "Data (Scientist)"]
target=[]

for m in sample_text:
    pattern=re.compile(r'\(|\)')
    result=pattern.sub("",m)
    target.append(result)

results=[]

for m in target:
    pattern = re.compile(r'\s+\.')
    result=pattern.sub('.',m)
    results.append(result)

print(results)
```

```
['example.com', 'hr@fliprobo.com', 'github.com', 'Hello Data Science World',
'Data Scientist']
```

[11]:
```
#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”]

[12]:
```
sample_text: ["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello␣
 ↪(Data Science World)", "Data (Scientist)"]
results=[]

for m in sample_text:
    pattern = re.compile(r'\s*\([^)]*\)')
    result=pattern.sub('',m)
    results.append(result)

print(results)
```

```
['example', 'hr@fliprobo', 'github', 'Hello', 'Data']
```

[13]:
```
#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’]

[14]:
```
sample_text="ImportanceOfRegularExpressionsInPython"

output = re.split(r'(?<=[a-z])(?=[A-Z])',sample_text)

output
```

[14]:
```
['Importance', 'Of', 'Regular', 'Expressions', 'In', 'Python']
```

[15]:
```
#Question 8- Create a function in python to insert spaces between words␣
↪starting with numbers.
```

Sample Text: “RegularExpression1IsAn2ImportantTopic3InPython”

Expected Output: RegularExpression 1IsAn 2ImportantTopic 3InPython

[16]:
```
sample_text="RegularExpression1IsAn2ImportantTopic3InPython"

output = re.sub(r'(?<=[a-z])(?=[0-9])'," ",sample_text)

output
```

[16]:
```
'RegularExpression 1IsAn 2ImportantTopic 3InPython'
```

[17]:
```
#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

[18]:
```
sample_text="RegularExpression1IsAn2ImportantTopic3InPython"

output = re.sub(r'(?<=\D)(?=\d)|(?<=\d)(?=\D)'," ",sample_text)

output
```

[18]:
```
'RegularExpression 1 IsAn 2 ImportantTopic 3 InPython'
```

[19]:
```
#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’]

[20]:
```
#Note- Store given sample text in the text file and then extract email␣
↪addresses.
```

[21]:
```
file=open("Question10.txt","r")
results=[]

for i in file:

    pattern = re.compile(r'\b[A-Za-z0-9\.\_\%\+\-]+[@][a-z.]+\.[A-Z|a-z]{2,}\b')
    matches = re.findall(pattern,i)
    results.append(matches)

print(results)
```

```
[['xyz@domain.com', 'xyz.abc@sdomain.domain.com'], ['hr@fliprobo.com']]
```

[22]:
```
#Question 11- Write a Python program to match a string that contains only upper␣
↪and lowercase letters, numbers, and underscores
```

[23]:
```
string = ['Abava556/-', 'Amelia62*%#', 'Abigail44_', 'Raj$#%^', 'Dd*&46_',␣
↪'S_am5_']

results=[]

for i in string:
    pattern = r'^[a-zA-Z0-9_]*$'
    result = re.search(pattern,i)
    results.append(result)

results
```

[23]:
```
[None,
 None,
 <re.Match object; span=(0, 10), match='Abigail44_'>,
 None,
 None,
 <re.Match object; span=(0, 6), match='S_am5_'>]
```

[24]:
```
#Question 12- Write a Python program where a string will start with a specific␣
↪number
```

[25]:
```
strings = ['9EmilyDavis', '17TheodoreDinh', '21LunaSanders',␣
↪'34PenelopeJordan', '46AustinVo', '59JoshuaGupta', '65RubyBarnes',␣
↪'79LukeMartin', '80EastonBailey', '92MadelineWalker', '102SavannahAli']

def match(string):
```

```
    pattern = r"^9"    # Replace 9 with the desired number
    return (re.match(pattern, string))

output = [string for string in strings if match(string)]
print(output)
```

```
['9EmilyDavis', '92MadelineWalker']
```

[26]:
```
strings = ['9EmilyDavis', '17TheodoreDinh', '21LunaSanders',␣
 ↪'34PenelopeJordan', '46AustinVo', '59JoshuaGupta', '65RubyBarnes',␣
 ↪'79LukeMartin', '80EastonBailey', '92MadelineWalker', '102SavannahAli']

results=[]

for p in strings:
    pattern = r'^9[\w]+'  # Replace 9 with the desired number
    result  = re.match(pattern,p)
    results.append(result)

results
```

[26]:
```
[<re.Match object; span=(0, 11), match='9EmilyDavis'>,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 <re.Match object; span=(0, 16), match='92MadelineWalker'>,
 None]
```

[27]:
```
#Question 13- Write a Python program to remove leading zeros from an IP address
```

[28]:
```
ip_address = '198.0020.010.0210'

def remove_all_zeros(ip_address):
    pattern = r'[0]*'
    return re.sub(pattern,"", ip_address)
print("remove all zeros:", remove_all_zeros(ip_address))

#remove leading zeros

result = re.sub(r'\.[0]*', '.', ip_address)
print("remove leading zeros:",result)
```

```
remove all zeros: 198.2.1.21
```

```
remove leading zeros: 198.20.10.210
```

[29]:
```
#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

[30]:
```
text = open("Question-14.txt","r")
results=[]

for i in text:
    pattern = r"\b\w+\s\d{1,2}\w{0,}\s\d{4}\b"
    match   = re.search(pattern,i)
    results.append(match)

if match:
    print(match.group())
```

```
August 15th 1947
```

[31]:
```
#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'

[32]:
```
sample_text = 'The quick brown fox jumps over the lazy dog'

patterns = ['fox', 'dog', 'horse']

for i in patterns:

    if re.search(i,sample_text):
        print('"%s" in "%s" =' % (i, sample_text),"found")
    else:
        print('"%s" in "%s" =' % (i, sample_text),"Not found")
```

```
"fox" in "The quick brown fox jumps over the lazy dog" = found
"dog" in "The quick brown fox jumps over the lazy dog" = found
"horse" in "The quick brown fox jumps over the lazy dog" = Not found
```

[33]:
```
#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'

[34]:
```
sample_text = 'The quick brown fox jumps over the lazy dog.'
pattern = 'fox'

match = re.search(pattern,sample_text)

s = match.start()
e = match.end()

print('Found "%s" in "%s" from %d to %d ' % (match.re.pattern, match.string, s,␣
 ↪e))
```

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

[35]:
```
#Question 17- Write a Python program to find the substrings within a string.
```

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

Pattern : 'exercises'.

[36]:
```
sample_text = 'Python exercises, PHP exercises, C# exercises'
pattern = 'exercises'
result =re.findall(pattern,sample_text)

result
```

[36]:
```
['exercises', 'exercises', 'exercises']
```

[37]:
```
#Question 18- Write a Python program to find the occurrence and position of the␣
 ↪substrings within a string.
```

[38]:
```
text = 'Python exercises, PHP exercises, C# exercises'
pattern = 'exercises'

for match in re.finditer(pattern, text):
    s = match.start()
    e = match.end()

    print('Found "%s" at %d:%d' % (text[s:e], s, e))
```

```
Found "exercises" at 7:16
Found "exercises" at 22:31
Found "exercises" at 36:45
```

[39]:
```
#Question 19- Write a Python program to convert a date of yyyy-mm-dd format to␣
 ↪dd-mm-yyyy format.
```

[40]:
```
date= "2024-10-05"
```

```
def new_formate(date):
    return re.sub(r'(\d{4})-(\d{1,2})-(\d{1,2})',r'\3-\2-\1', date)

new_date = new_formate(date)

print("original format(yyyy-mm-dd):",date)
print("new format(dd-mm-yyyy):",new_date)
```

```
original format(yyyy-mm-dd): 2024-10-05
new format(dd-mm-yyyy): 05-10-2024
```

[41]:
```
#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']

[42]:
```
string= '01.12 0132.123 2.31875 145.8 3.01 27.25 0.25'

def find_all_decimal_numbers(string):
    pattern=re.compile(r'\b\d+\.\d{1,2}\b')
    match=re.findall(pattern,string)
    return match

result= find_all_decimal_numbers(string)

print(result)
```

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

[43]:
```
#Question 21- Write a Python program to separate and print the numbers and␣
 ↪their position of a given string.
```

[44]:
```
sample_text = 'On August 15th 1947 that India was declared independent from␣
 ↪British colonialism'
pattern = r'\d+'

matches = re.finditer(pattern,sample_text)

for m in matches:
    print('Number {} found at position {}'.format(m.group(), m.span()))
```

```
Number 15 found at position (10, 12)
Number 1947 found at position (15, 19)
```

[45]:
```
#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

[46]:
```
Sample_text='My marks in each semester are: 947, 896, 926, 524, 734, 950, 642'

pattern = re.findall(r'\d+',Sample_text)

number = map(int, pattern)

print("Maximum value:",max(number))
```

```
Maximum value: 950
```

[47]:
```
#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

[48]:
```
sample_text="RegularExpressionIsAnImportantTopicInPython"

output = re.sub(r'(?<=[a-z])(?=[A-Z])'," ",sample_text)

output
```

[48]:
```
'Regular Expression Is An Important Topic In Python'
```

[49]:
```
string = 'RegularExpressionIsAnImportantTopicInPython'

def insert_spaces(string):
    words = re.split(r'(?=[A-Z])', string)
    return ' '.join(words)

result = insert_spaces(string)

print(result)
```

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

[50]:
```
#Question 24- Python regex to find sequences of one upper case letter followed␣
 ↪by lower case letters
```

[51]:
```
sample_text="RegularExpressionIsAnImportantTopicInPython"

output = re.findall(r'[A-Z][a-z]+',sample_text)

output
```

[51]:
```
['Regular', 'Expression', 'Is', 'An', 'Important', 'Topic', 'In', 'Python']
```

[52]:
```
#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

[53]:
```
sentence = "Hello hello world world"

def remove_duplicates(sentence):
    pattern = r'\b(\w+)(\s+\1\b)+'
    result  = re.sub(pattern, r'\1',sentence)
    return result

output = remove_duplicates(sentence)
print(output)
```

```
Hello hello world
```

[54]:
```
#Question 26-  Write a python program using RegEx to accept string ending with␣
↪alphanumeric character.
```

[55]:
```
string =input("Enter An Alphanumeric String:")

def check_string(string):
    match = re.search(r'\w$', string)
    return match

if check_string(string):
  print("String ends with an alphanumeric character:TRUE")
else:
  print("String does not end with an alphanumeric character:FALSE")
```

```
Enter An Alphanumeric String:HELLO123
String ends with an alphanumeric character:TRUE
```

[56]:
```
#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’]

[57]:
```
sample_text="""RT @kapil_kausik: #Doltiwal I mean #xyzabc is "hurt" by␣
↪#Demonetization as the same has rendered USELESS␣
↪<ed><U+00A0><U+00BD><ed><U+00B1><U+0089> "acquired funds" No wo"""
```

```
def extract_hashtags(sample_text):

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

results=extract_hashtags(sample_text)

results
```

[57]:
```
['#Doltiwal', '#xyzabc', '#Demonetization']
```

[58]:
```
#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

[59]:
```
sample_text="@Jags123456 Bharat band on 28??
 ↪<ed><U+00A0><U+00BD><ed><U+00B8><U+0082>Those who  are protesting␣
 ↪#demonetization  are all different party leaders"

def extract_strange_symbol(sample_text):

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

results=extract_strange_symbol(sample_text)

results
```

[59]:
```
'@Jags123456 Bharat band on 28??<ed><ed>Those who  are protesting
#demonetization  are all different party leaders'
```

[60]:
```
#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.

[61]:
```
text_file=open("Question-29.txt","r")
```

```
results=[]

for i in text_file:
    pattern = r"\b\d+\-\d+\-\d+\b"
    match   = re.findall(pattern,i)
    results.append(match)

results
```

[61]:
```
[['12-09-1992', '15-12-1999']]
```

[62]:
```
#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.

[63]:
```
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."

def remove_to_string(sample_text):
    pattern=re.compile(r'\b\w{2,4}\b')
    match=re.sub(pattern,"",sample_text)
    return match

result= remove_to_string(sample_text)

print (re.sub((r'\s{2,}')," ",result))
```

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

[ ]:
```

```