Regular-Expressions-Assignment

March 7, 2024

0.0.1 **********

0.1 Regular Expression Assignment Solutions

0.2 Rajib Dutta

0.3 duttarajib78@gmail.com

0.4 Batch DS2402

0.4.1 **********

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:

[ ]:
```
import re
import pandas as pd
import numpy as np
```

[ ]:
```
def replaceSpaceCommaDot(text):
    pattern=r'[\s,\.]'
    p=re.compile(pattern=pattern)
    return p.sub(string=text, repl=':')

text="Python Exercises, PHP exercises."
replaceSpaceCommaDot(text)
```

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

Question 2- Create a dataframe using the dictionary below and remove everything (commas (,), !, XXXX, ;, etc.) from the columns except words.

Dictionary- {'SUMMARY' : ['hello, world!', 'XXXXX test', '123four, five:; six...']}

Expected output-

0 hello world

1 test

2 four five six

```
[ ]: def keepOnlyWords(text):
         pattern=r'([^a-zA-Z\s]+)|\b([Xx]{2}\w*)\b'
         p=re.compile(pattern=pattern)
         return p.sub(string=text, repl='')

     theDict={'SUMMARY' : ['hello, world!', 'XXXXX test', '123four, five:; six...',␣
      ↪', bubble,, xb']}
     df=pd.DataFrame(data=theDict)
     df.SUMMARY=df.SUMMARY.apply(keepOnlyWords)
     df.SUMMARY
```

```
[ ]: 0      hello world
     1             test
     2    four five six
     3        bubble xb
     Name: SUMMARY, dtype: object
```

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.

```
[ ]: def findWordsOfLengthFourOrMore(text):
         pattern=r'\b\w{4,}\b'
         p=re.compile(pattern=pattern)
         return p.findall(string=text)

     findWordsOfLengthFourOrMore('The woman loves her son 123_so00␣
      ↪muuuuuuuuuuuuuchhhh')
```

```
[ ]: ['woman', 'loves', '123_so00', 'muuuuuuuuuuuuuchhhh']
```

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.

```
[ ]: def findWordsOfLengthThreeToFive(text):
         pattern=re.compile(pattern=r'\b(\w{3,5})\b')
         return pattern.findall(text)
     text='The woman loves her son 123_so0o muuchhhh'
     findWordsOfLengthThreeToFive(text)
```

```
[ ]: ['The', 'woman', 'loves', 'her', 'son']
```

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

[ ]:
```
def removeParenthesis(texts):
    pattern1=re.compile(pattern=r'[\(\)]')
    intrmediate_texts=[pattern1.sub(string=word, repl='') for word in texts]
    pattern2=re.compile(pattern=r'\s+(\W+\w+)')
    clean_texts=[]
    for text in intrmediate_texts:
        clean_texts.append(pattern2.sub(string=text, repl=r'\1'))
    return clean_texts

texts=["example (.com)", "hr@fliprobo (.com)", "github (.com)", "Hello (Data␣
 ↪Science World)", "Data (Scientist)"]
removeParenthesis(texts)
```

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

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.

[ ]:
```
def removeParenthesisArea(text):
    pattern=r'\(.*\)'
    p=re.compile(pattern=pattern)
    return p.sub(string=text, repl='').strip()

with open('./sample_text.txt', 'w') as f:
```

```
    f.writelines(["example (.com)\n", "hr@fliprobo (.com)\n", "github (.
 ↪com)\n", "Hello (Data Science World)\n", "Data (Scientist)"])

with open('./sample_text.txt', 'r') as f:
    sample_text=f.readlines()

[removeParenthesisArea(line) for line in sample_text]
```

[ ]:
```
['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']

[ ]:
```
def splitByCapLetter(text):
    pattern=r'[A-Z]+'
    p=re.compile(pattern=pattern)
    init_letters=[text[it.start():it.end()] for it in p.finditer(text)]
    splitted_words=[word for word in p.split(text)]
    final_texts=[]
    for i, word in enumerate(init_letters):
        final_texts.append(word+splitted_words[i+1])
    if splitted_words[0]!='':
        final_texts.insert(0, splitted_words[0])
    return final_texts

text='anImportanceOOOfRegularExpressionsInPythonIsH'
print(splitByCapLetter(text))
text='IMportanceOfRegularExpressionsINnPythonL'
print(splitByCapLetter(text))
```

```
['an', 'Importance', 'OOOf', 'Regular', 'Expressions', 'In', 'Python', 'Is',
'H']
['IMportance', 'Of', 'Regular', 'Expressions', 'INn', 'Python', 'L']
```

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

Sample Text: "RegularExpression1IsAn2ImportantTopic3InPython"

Expected Output: RegularExpression 1IsAn 2ImportantTopic 3InPython

[ ]:
```
def putSpaceAfterNumber(text):
    pattern=re.compile(pattern=r'(\d+\D*)')
    return pattern.sub(string=text, repl=r' \1').strip()
```

```
text='0000RegularExpression100IsAn2ImportantTopic30InPython4000000'
print(putSpaceAfterNumber(text))
text='RegularExpression1IsAn2ImportantTopic3InPython'
print(putSpaceAfterNumber(text))
```

```
0000RegularExpression 100IsAn 2ImportantTopic 30InPython 4000000
RegularExpression 1IsAn 2ImportantTopic 3InPython
```

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

[ ]:
```
def putSpaceAfterCapLetterOrNumber(text):
    pattern=re.compile(pattern=r'(\d+|[A-Z]+)')
    return pattern.sub(string=text, repl=r' \1').strip()

text='0000RegularExpression100IsAn2ImportantTopic30IIInPython4000000'
print(putSpaceAfterCapLetterOrNumber(text))
text='RegularExpression1IsAn2ImportantTopic3InPython'
print(putSpaceAfterCapLetterOrNumber(text))
```

```
0000 Regular Expression 100 Is An 2 Important Topic 30 IIIn Python 4000000
Regular Expression 1 Is An 2 Important Topic 3 In Python
```

Question 10- Use the github link below to read the data and create a dataframe. After creating the dataframe extract the first 6 letters of each country and store in the dataframe under a new column called first_five_letters.

Github Link- https://raw.githubusercontent.com/dsrscientist/DSData/master/happiness_score_d

[ ]:
```
happiness=pd.read_csv("https://raw.githubusercontent.com/dsrscientist/DSData/
 ↪master/happiness_score_dataset.csv")
def extractFirstSix(text):
    pattern=r'^[A-Za-z].{5,5}'
    p=re.compile(pattern=pattern)
    if len(text) >= 6:
        return p.match(text).group()
    else:
        return text

happiness['first_five_letters']=happiness.Country.apply(extractFirstSix).values
happiness.head()
```

[ ]:

	Country	Region	Happiness Rank	Happiness Score	\
0	Switzerland	Western Europe	1	7.587	
1	Iceland	Western Europe	2	7.561	

```
2      Denmark  Western Europe                3            7.527
3       Norway  Western Europe                4            7.522
4       Canada   North America                5            7.427

   Standard Error  Economy (GDP per Capita)   Family  \
0         0.03411                   1.39651  1.34951
1         0.04884                   1.30232  1.40223
2         0.03328                   1.32548  1.36058
3         0.03880                   1.45900  1.33095
4         0.03553                   1.32629  1.32261

   Health (Life Expectancy)  Freedom  Trust (Government Corruption)  \
0                   0.94143  0.66557                        0.41978
1                   0.94784  0.62877                        0.14145
2                   0.87464  0.64938                        0.48357
3                   0.88521  0.66973                        0.36503
4                   0.90563  0.63297                        0.32957

   Generosity  Dystopia Residual first_five_letters
0     0.29678            2.51738             Switze
1     0.43630            2.70201             Icelan
2     0.34139            2.49204             Denmar
3     0.34699            2.46531             Norway
4     0.45811            2.45176             Canada
```

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

[ ]:
```
def matchLettersNumNUds(text):
    pattern=r'\w+'
    p=re.compile(pattern=pattern)
    return p.findall(text)
text='A boy_ got this %$+ on his 10_Notebooks costing $20'
matchLettersNumNUds(text)
```

[ ]:
```
['A', 'boy_', 'got', 'this', 'on', 'his', '10_Notebooks', 'costing', '20']
```

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

[ ]:
```
def matchStartingWithSpecificNum(text, start):
    pattern=rf'^{start}.*'
    p=re.compile(pattern=pattern)
    m=p.match(text)
    if m is not None:
        return m.group()

texts=['10_Notebooks costing $20', '20_Notebooks costing $30', '10 Pens costing␣
 ↪$15']
```

```
[matchStartingWithSpecificNum(text, 10) for text in texts if␣
 ↪matchStartingWithSpecificNum(text, 10) is not None]
```

[ ]: ['10_Notebooks costing $20', '10 Pens costing $15']

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

[ ]:
```
def removeLeadingZerosFromIP(ip):
    pattern=re.compile(pattern=r'\b[0]+(\d+)\b')
    return pattern.sub(string=ip,repl=r'\1')

ip='05.08.19.180'
print(removeLeadingZerosFromIP(ip))
ip='19.80.0019.00000180'
print(removeLeadingZerosFromIP(ip))
```

```
5.8.19.180
19.80.19.180
```

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.

[ ]:
```
def extractDates(text):
    pattern_month=r'\b(Jan(uary)?|Feb(ruary)?|Mar(ch)?|Apr(il)?|May|Jun(e)?
 ↪|Jul(y)?|Aug(ust)?|Sep(tember)?|Oct(ober)?|Nov(ember)?|Dec(ember)?)\b'
    pattern_day=r'\s+(\d{1,2})(st?|nd?|rd?|th?)'
    pattern_year=r'\s+(\d{4})'
    pattern=f'{pattern_month}{pattern_day}{pattern_year}'
    p=re.compile(pattern=pattern)
    dates=[]
    for iter in p.finditer(text):
        dates.append(iter.group())
    return dates

with open(file='./date_file.txt', mode='w') as f:
    f.write('''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 where Augusta was the␣
 ↪hero''')

with open(file='./date_file.txt', mode='r') as f:
```

```
    text=f.read()

extractDates(text)
```

[ ]: ['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'

[ ]:
```
def searchLiteral(text, *literals):
    if len(literals) > 0:
        matches=[]
        for literal in literals:
            pattern=r'\b'+literal+r'\b'
            p=re.compile(pattern=pattern)
            matches.append(p.search(text).group())
        return matches
    else:
        return 'No literal provided'

text='''The quick brown fox jumps over the lazy dog.
        Then the fox had bitten the dog.
        Being attached the dog hopped on a horse.
        The horse helped the dog to escape from the fox'''
print(searchLiteral(text, 'fox'))
print(searchLiteral(text, 'fox', 'dog'))
print(searchLiteral(text, 'fox', 'dog', 'horse'))
```

```
['fox']
['fox', 'dog']
['fox', 'dog', 'horse']
```

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'

[ ]:
```
def searchLiteral(text, *literals):
    if len(literals) > 0:
        literal_dict={}
        for literal in literals:
            pattern=r'\b'+literal+r'\b'
            p=re.compile(pattern=pattern)
            locations=[]
```

```
            for m in p.finditer(text):
                locations.append(m.span())
            literal_dict[m.group()]=locations
        return literal_dict
    else:
        return 'No literal provided'

text='''The quick brown fox jumps over the lazy dog.
        Then the fox had bitten the dog.
        Being attached the dog hopped on a horse.
        The horse helped the dog to escape from the fox'''
print(searchLiteral(text, 'fox'))
print(searchLiteral(text, 'fox', 'dog'))
print(searchLiteral(text, 'fox', 'dog', 'horse'))
```

```
{'fox': [(16, 19), (63, 66), (191, 194)]}
{'fox': [(16, 19), (63, 66), (191, 194)], 'dog': [(40, 43), (82, 85), (115,
118), (168, 171)]}
{'fox': [(16, 19), (63, 66), (191, 194)], 'dog': [(40, 43), (82, 85), (115,
118), (168, 171)], 'horse': [(131, 136), (151, 156)]}
```

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

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

Pattern : 'exercises'.

[ ]:
```
def getAllSubstrings(text, substring):
    pattern=rf'{substring}'
    p=re.compile(pattern=pattern)
    return p.findall(text)

text='Python exercises, PHP exercises, C# exercises'
getAllSubstrings(text, 'C#')
```

[ ]:
```
['C#']
```

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

[ ]:
```
def getAllSubstringsWithPositions(text, substring):
    pattern=rf'{substring}'
    p=re.compile(pattern=pattern)
    sub_string_pos={substring:[]}
    for m in p.finditer(text):
        sub_string_pos[substring].append(m.span())
    return sub_string_pos

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

```
print(getAllSubstringsWithPositions(text, 'exercises'))
print(getAllSubstringsWithPositions(text, 'C#'))
```

```
{'exercises': [(7, 16), (22, 31), (36, 45)]}
{'C#': [(33, 35)]}
```

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

[ ]:
```
def changeDateFormat(doc):
    pattern=r'\d{4,4}-\d{2,2}-\d{2,2}'
    p=re.compile(pattern=pattern, flags=re.IGNORECASE)
    matches=p.findall(doc)
    date_matches=[re.split(pattern='-', string=match) for match in matches]
    new_format_dates=[]
    for aDate in date_matches:
        new_format_dates.append(aDate[2]+'-'+aDate[1]+'-'+aDate[0])
    return new_format_dates

dt1 = "2026-01-02 is the format in yyyy-mm-dd with another date as 2024-11-12"
print(changeDateFormat(dt1))
```

```
['02-01-2026', '12-11-2024']
```

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']

[ ]:
```
def extractNumbersWithUptoTwoDecimal(text):
    pattern=r'\b\d+\.\d{1,2}\b'
    p=re.compile(pattern=pattern)
    return p.findall(text)

text='01.12 0132.123 2.31875 145.8 3.01 27.25 0.25'
extractNumbersWithUptoTwoDecimal(text)
```

[ ]:
```
['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.

[ ]:
```
def extractNumbersWithPosition(text):
    pattern=r'\d'
    p=re.compile(pattern=pattern)
    num_with_pos={}
    for m in p.finditer(text):
        if m.group() not in num_with_pos.keys():
```

```
            num_with_pos[m.group()]=[]
            num_with_pos[m.group()].append(m.start())
        else:
            num_with_pos[m.group()].append(m.start())
    return num_with_pos

text='I got 3794 videos and 459633 pictures on my phone'
extractNumbersWithPosition(text)
```

[ ]: {'3': [6, 26, 27], '7': [7], '9': [8, 24], '4': [9, 22], '5': [23], '6': [25]}

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

[ ]:
```
def extractLargestNumber(text):
    pattern=re.compile(pattern=r'\b\d+\b')
    numbers=np.array(pattern.findall(text))
    numbers=numbers.astype(np.int64)
    return numbers.max()

text='My marks in each semester are: 947, 896, 926, 524, 734, 950, 642'
extractLargestNumber(text)
```

[ ]: 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

[ ]:
```
def putSpaceBeforeWordsStartingWithCapitalLetter(text):
    pattern=re.compile(pattern=r'([A-Z][^A-Z]*)')
    return pattern.sub(string=text, repl=r' \1').strip()

text='RegularExpressionIsAnImportantTopicInPython'
putSpaceBeforeWordsStartingWithCapitalLetter(text)
```

[ ]: '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

[ ]:
```
def extractSequencesWithInitCap(text):
    pattern=re.compile(pattern=r'[A-Z][a-z]*')
    return pattern.findall(text)

text='RegularExpressionIsAnImportantTopicInPythonH'
extractSequencesWithInitCap(text)
```

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

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

[ ]:
```
def removeDuplicateWords(text):
    pattern=re.compile(pattern=r'\b(\w+)(\W+\1)+\b')
    return pattern.sub(string=text, repl=r'\1')

text='Hello hello world world world'
print(removeDuplicateWords(text))
text='Hello hello hello world world world'
print(removeDuplicateWords(text))
```

```
Hello hello world
Hello hello world
```

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

[ ]:
```
def acceptTextEndingWithAlphaNumeric(text):
    pattern=re.compile(pattern=r'\w$')
    if len(pattern.findall(text))>0:
        return True
    else:
        return False

while True:
    user_inp=input('Enter a text ending with alpha numeric charachter:')
    if acceptTextEndingWithAlphaNumeric(user_inp):
```

```
        print(f'Your input \'{user_inp}\' is accepted')
        break
    else:
        continue
```

```
Your input 'Hi there' is accepted
```

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’]

[ ]:
```
def extractHashTag(text):
    pattern=re.compile(pattern=r'#\w+')
    return pattern.findall(text)

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"""

extractHashTag(text)
```

[ ]:
```
['#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>T 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

[ ]:
```
def removeSymbols(text):
    pattern=re.compile(pattern=r'<.*>')
    return pattern.sub(string=text, repl='')

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'
removeSymbols(text)
```

[ ]:
```
'@Jags123456 Bharat band on 28??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.

[ ]:
```
def extractDatesFromFile(f):
    with open(file=f, mode='r') as txt:
        text=txt.read()
    pattern=re.compile(pattern=r'\b(0?[1-9]|1[0-9]|2[0-9]|3[0-1])-(0?
 ↪[1-9]|1[0-2])-(\d{4})\b')
    matches=[]
    for m in pattern.finditer(text):
        matches.append(m.group())
    return matches

with open(file='./text_file.txt', mode='w') as f:
    f.write('Ron was born on 12-09-1992 and he was admitted to school␣
 ↪15-12-1999.')
extractDatesFromFile('./text_file.txt')
```

[ ]:
```
['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.

[ ]:
```
def removeWordsOfLengthTwoToFour(text):
    pattern=re.compile(pattern=r'\b\w{2,4}\b')
    return pattern.sub(string=text, repl='').strip()

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.'
removeWordsOfLengthTwoToFour(text)
```

```
[ ]: 'following example creates  ArrayList  a capacity   elements. 4 elements   added
     ArrayList   ArrayList  trimmed accordingly.'
```