Chap.-3 R.E.(Regular Expression)

- ## Metacharacters
  - [] - Set of characters
  - \ - Signals a special sequence (Escape special characters)
  - .(dot) - Any character
  - ^ - Starts with (but in [...] - inverse)
  - $ - Ends with - it's only check at 0 index
  - '*' - Zero or More occurance
  - '+' - One or More occurance
  - ? - Zero or One occurance
  - {} - Exact occurance
  - ! - Either or
  - \d - only digit(0-9)
  - \D - Except Digit Everything
  - \s - whitespace
  - \w - AlphaNumeric [a-z, A-Z, 0-9]
  - \W - (Not a Word Character): Matches any character that is not a letter, digit, or underscore. This includes punctuation, spaces, and other symbols.
  - \b - begining or end of a word
  - \B - NOT or end of a word

- ## re.findall(pattern, string)

In [1]:

```python
import re
text = 'This is the Python Programming class of 2025'
reg = r't\w*'
ans = re.findall(reg,text)
print(ans) # ['the', 'thon']
```

```
['the', 'thon']
```

Type *Markdown* and LaTeX: α^2

In [2]:

```python
import re
text = 'This is the Python Programming class of 2025'
reg = r'\bt\w*'
ans = re.findall(reg,text)
print(ans)
```

```
['the']
```

In [4]:

```python
import re
text = 'This is the Python Programming class of 2025'
reg = r'\b[t|T]\w*'
ans = re.findall(reg,text)
print(ans)
```

```
['This', 'the']
```

In [5]:

```python
import re
text = 'This is the Python Programming class of 2025'
reg = r'\bt\w*'
ans = re.findall(reg,text,re.I)
print(ans)
```

```
['This', 'the']
```

- ### re.I - Ignore Case(Capital or small).
- ## re.search(reg,text)

In [6]:

```python
import re
text = 'This is the Python Programming class of 2025'
reg = r'\bt\w*'
ans = re.search(reg,text,re.I)
print(ans)
```

```
<re.Match object; span=(0, 4), match='This'>
```

In [7]:

```
import re
text = 'This is the Python Programming class of 2025'
reg = r'\bt\w*'
ans = re.search(reg,text,re.I)
print(ans.group())
```

```
This
```

In [8]:

```
import re
text = 'This is the Python Programming class of 2025'
reg = r'\bt\w*'
ans = re.search(reg,text,re.I)
print(ans.span())
```

```
(0, 4)
```

In [11]:

```
import re
text = 'This is the Python Programming class of 2025'
reg = r'\bt\w*'
ans = re.search(reg,text,re.I)
print(ans.start())
```

```
0
```

In [12]:

```
import re
text='This is the python Programming class of 2025'
reg=r'\bz\w*'
ans=re.search(reg,text,re.I)
print(ans)
# if not in string then return none and other method dont work on none so it gives error
```

```
None
```

In [16]:

```
import re
text = 'This is the Pyth-on Prog*ramming class of 2025'
reg = r'\W'
ans = re.split(reg,text)
print(ans)
```

```
['This', 'is', 'the', 'Pyth', 'on', 'Prog', 'ramming', 'class', 'of', '2025']
```

In [17]:

```
import re
text = 'This is the Pyth-on Prog*ramming class of 2025'
reg = r'\W'
ans = re.split(reg,text,2) # max 2 split
print(ans) # ['This', 'is', 'the Pyth-on Prog*ramming class of 2025']

# if 2 split then 3 peice of String
```

```
['This', 'is', 'the Pyth-on Prog*ramming class of 2025']
```

In [18]:

```
import re
text = 'This is the Pyth-on Prog*ramming class of 2025'
reg = r'\w'
ans = re.split(reg,text)
print(ans)
```

```
['', '', '', '', ' ', '', ' ', '', '', ' ', '', '', '', '-', '', ' ', '', '', '', '*', '', '', '', '', '', '', ' ', '',
'', '', '', ' ', '', ' ', '', '', '', '']
```

- **re.sub(reg,'$',text) - for replace**

In [20]:

```
import re
text = 'This is the Pyth-on Prog*ramming class of 2025'
reg = r'\s'
ans = re.sub(reg,'$',text)
print(ans)
```

```
This$is$the$Pyth-on$Prog*ramming$class$of$2025
```

In [23]:

```
# replace digits with P
import re
text = 'This is the Pyth-on Prog*ramming class of 2025'
reg = r'\d'
ans = re.sub(reg,'P',text)
print(ans)
```

```
This is the Pyth-on Prog*ramming class of PPPP
```

In [24]:

```python
import re
text = 'This is the Pyth-on Prog*ramming class of 2025'
reg = r'\d+'
ans = re.sub(reg,'P',text)
print(ans)
```

```
This is the Pyth-on Prog*ramming class of P
```

In [33]:

```python
import re
text = ' This is the Pyth-on Prog*ramming class of 2025'
reg = r'^\s'
ans = re.sub(reg,'$',text)
print(ans)
# starting with
```

```
$This is the Pyth-on Prog*ramming class of 2025
```

In [41]:

```python
import re
text = ' This is the Pyth-on Prog*ramming class of 2025 '
reg = r'^\s|\s$'
ans = re.sub(reg,'$',text)
print(ans)
```

```
$This is the Pyth-on Prog*ramming class of 2025$
```

In [40]:

```python
import re
text = ' This is the Pyth-on Prog*ramming class of 2025    '
reg = r'^\s|\s$'
ans = re.sub(reg,'$',text)
print(ans)
```

```
$This is the Pyth-on Prog*ramming class of 2025   $
```

In [39]:

```python
import re
text = ' This is the Pyth-on Prog*ramming class of 2025    '
reg = r'^\s|\s*$'
ans = re.sub(reg,'$',text)
print(ans)
```

```
$This is the Pyth-on Prog*ramming class of 2025$$
```

In [42]:

```python
import re
text = ' This is the Pyth-on Prog*ramming class of 2025    '
reg = r'^\s|\s+$'
ans = re.sub(reg,'$',text)
print(ans)
```

```
$This is the Pyth-on Prog*ramming class of 2025$
```

In [57]:

```python
import re
text = 'This is the Pyth-on Prog*ramming class of 2025'
reg = r'^\bP\w*'
ans = re.findall(reg,text)
print(ans)
```

```
[]
```

In [58]:

```python
import re
text = 'PThis is the Pyth-on Prog*ramming class of 2025'
reg = r'^\bP\w*'
ans = re.findall(reg,text)
print(ans)
```

```
['PThis']
```

In [59]:

```python
import re
text = 'PThis is the Pyth-on Prog*ramming class of 2025'
reg = r'^P*'
ans = re.findall(reg,text)
print(ans)
```

```
['P']
```

In [72]:

```
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
print(text)
```

```
Alice's Adventures in Wonderland

                ALICE'S ADVENTURES IN WONDERLAND

                          Lewis Carroll

               THE MILLENNIUM FULCRUM EDITION 3.0




                            CHAPTER I

                      Down the Rabbit-Hole


  Alice was beginning to get very tired of sitting by her sister
on the bank, and of having nothing to do:  once or twice she had
peeped into the book her sister was reading, but it had no
```

In [71]:

```
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
# print(text)
reg = r'\b[aeiou]\w*'
ans = re.findall(reg,text)
print(ans)
```

```
['in', 'of', 'on', 'and', 'of', 'once', 'or', 'into', 'it', 'or', 'in', 'it', 'and', 'is', 'use', 'of', 'a', 'or', 'i
n', 'own', 'as', 'as', 'and', 'of', 'a', 'of', 'up', 'and', 'a', 'eyes', 'in', 'it', 'out', 'of', 'itself', 'it', 'ove
r', 'afterwards', 'it', 'occurred', 'ought', 'at', 'at', 'it', 'all', 'actually', 'and', 'at', 'it', 'and', 'on', 'it',
'across', 'a', 'either', 'a', 'or', 'a', 'out', 'of', 'it', 'and', 'across', 'after', 'it', 'and', 'in', 'it', 'a', 'un
der', 'another', 'after', 'it', 'once', 'in', 'out', 'again', 'on', 'a', 'and', 'a', 'about', 'a', 'or', 'of', 'as', 'a
bout', 'and', 'and', 'out', 'it', 'anything', 'at', 'of', 'and', 'and', 'and', 'and', 'upon', 'a', 'one', 'of', 'as',
'it', 'it', 'empty', 'of', 'it', 'into', 'one', 'of', 'as', 'it', 'after', 'a', 'as', 'of', 'all', 'at', 'anything', 'a
bout', 'it', 'even', 'if', 'off', 'of', 'an', 'end', 'aloud', 'of', 'earth', 'of', 'in', 'in', 'and', 'a', 'opportunit
y', 'off', 'as', 'one', 'it', 'it', 'over', 'about', 'or', 'idea', 'or', 'either', 'again', 'if', 'earth', 'it', 'out',
'among', 'one', 'as', 'it', 'at', 'all', 'ask', 'of', 'is', 'am', 'is', 'or', 'and', 'as', 'as', 'air', 'it', 'an', 'ig
norant', 'asking', 'it', 'ask', 'it', 'up', 'else', 'again', 'of', 'at', 'are', 'in', 'air', 'afraid', 'a', 'and', 'a',
'eat', 'and', 'on', 'in', 'a', 'of', 'eat', 'eat', 'and', 'eat', 'as', 'answer', 'either', 'it', 'it', 'off', 'and', 'i
n', 'and', 'earnestly', 'ever', 'eat', 'a', 'upon', 'a', 'of', 'and', 'and', 'over', 'a', 'and', 'up', 'on', 'in', 'a',
'up', 'it', 'all', 'overhead', 'another', 'and', 'in', 'it', 'a', 'away', 'and', 'in', 'it', 'as', 'it', 'a', 'ears',
'and', 'it', 'it', 'in', 'a', 'up', 'a', 'of', 'all', 'all', 'and', 'all', 'one', 'and', 'up', 'other', 'every', 'eve
r', 'out', 'again', 'upon', 'a', 'all', 'of', 'on', 'it', 'except', 'a', 'and', 'it', 'one', 'of', 'of', 'alas', 'eithe
r', 'or', 'at', 'any', 'it', 'open', 'any', 'of', 'on', 'upon', 'a', 'and', 'it', 'a', 'about', 'inches', 'in', 'and',
'it']
```

In [77]:

```
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
# print(text)
reg = r'\b[^aeiou ]\w*'
ans = re.findall(reg,text)
print(ans)

# ^ - inverse starts with
```

```
['Alice', "'s", 'Adventures', 'Wonderland', '\n', 'ALICE', "'S", 'ADVENTURES', 'IN', 'WONDERLAND', '\n', 'Lewis', 'Carr
oll', '\n', 'THE', 'MILLENNIUM', 'FULCRUM', 'EDITION', '3', '.0', '\n', 'CHAPTER', 'I', '\n', 'Down', 'the', 'Rabbit',
'-Hole', '\n', 'Alice', 'was', 'beginning', 'to', 'get', 'very', 'tired', 'sitting', 'by', 'her', 'sister', '\non', 'th
e', 'bank', ',', 'having', 'nothing', 'to', 'do', ':', 'twice', 'she', 'had', '\npeeped', 'the', 'book', 'her', 'siste
r', 'was', 'reading', ',', 'but', 'had', 'no', '\npictures', 'conversations', ',', 'what', 'the', 'book', ',', 'though
t', 'Alice', 'without', 'pictures', 'conversation', '?', 'So', 'she', 'was', 'considering', 'her', 'mind', 'well', 'sh
e', 'could', ',', 'for', 'the', 'hot', 'day', 'made', 'her', 'feel', 'very', 'sleepy', 'stupid', ')', 'whether', '\nth
e', 'pleasure', 'making', 'daisy', '-chain', 'would', 'be', 'worth', 'the', 'trouble', '\nof', 'getting', 'picking', 't
he', 'daisies', ',', 'when', 'suddenly', 'White', '\nRabbit', 'with', 'pink', 'ran', 'close', 'by', 'her', '.', 'Ther
e', 'was', 'nothing', 'so', 'VERY', 'remarkable', 'that', ';', 'nor', 'did', 'Alice', '\nthink', 'so', 'VERY', 'much',
'the', 'way', 'to', 'hear', 'the', 'Rabbit', 'say', 'to', '\nitself', ',', 'Oh', 'dear', '!', 'Oh', 'dear', '!', 'I',
'shall', 'be', 'late', '!', 'when', 'she', 'thought', '\nit', ',', 'to', 'her', 'that', 'she', 'to', 'have', '\nwondere
d', 'this', ',', 'but', 'the', 'time', 'seemed', 'quite', 'natural', ')', 'but', 'when', 'the', 'Rabbit', 'TOOK', 'A',
'WATCH', 'OUT', 'OF', 'ITS', 'WAISTCOAT', '-', 'POCKET', ',', 'looked', ',', 'then', 'hurried', ',', 'Alice', 'starte
d', 'to', '\nher', 'feet', ',', 'for', 'flashed', 'her', 'mind', 'that', 'she', 'had', 'never', '\nbefore', 'seen', 'ra
bbit', 'with', 'waistcoat', '-pocket', ',', 'watch', 'to', '\ntake', ',', 'burning', 'with', 'curiosity', ',', 'she',
'ran', 'the', '\nfield', ',', 'fortunately', 'was', 'just', 'time', 'to', 'see', 'pop', '\ndown', 'large', 'rabbit', '-
hole', 'the', 'hedge', '.', 'In', 'moment', 'down', 'went', 'Alice', ',', 'never', '\nconsidering', 'how', 'the', 'worl
d', 'she', 'was', 'to', 'get', '.', 'The', 'rabbit', '-hole', 'went', 'straight', 'like', 'tunnel', 'for', 'some', 'wa
y', ',', 'then', 'dipped', 'suddenly', 'down', ',', 'so', 'suddenly', 'that', 'Alice', 'had', 'not', '\nmoment', 'to',
'think', 'stopping', 'herself', 'before', 'she', 'found', 'herself', '\nfalling', 'down', 'very', 'deep', 'well', '.',
'Either', 'the', 'well', 'was', 'very', 'deep', ',', 'she', 'fell', 'very', 'slowly', ',', 'for', 'she', '\nhad', 'plen
ty', 'time', 'she', 'went', 'down', 'to', 'look', 'her', 'to', '\nwonder', 'what', 'was', 'going', 'to', 'happen', 'nex
t', '.', 'First', ',', 'she', 'tried', 'to', 'look', '\ndown', 'make', 'what', 'she', 'was', 'coming', 'to', ',', 'bu
t', 'was', 'too', 'dark', 'to', '\nsee', ';', 'then', 'she', 'looked', 'the', 'sides', 'the', 'well', ',', '\nnoticed',
'that', 'they', 'were', 'filled', 'with', 'cupboards', 'book', '-shelves', ';', 'here', 'there', 'she', 'saw', 'maps',
'pictures', 'hung', 'pegs', '.', 'She', '\ntook', 'down', 'jar', 'from', 'the', 'shelves', 'she', 'passed', ';', 'was',
'\nlabelled', 'ORANGE', 'MARMALADE', "'", 'but', 'to', 'her', 'great', 'disappointment', '\nwas', ':', 'she', 'did', 'n
ot', 'like', 'to', 'drop', 'the', 'jar', 'for', 'fear', 'killing', '\nsomebody', ',', 'so', 'managed', 'to', 'put', 'th
e', 'cupboards', 'she', '\nfell', 'past', '.', 'Well', '!', 'thought', 'Alice', 'to', 'herself', ',', 'such', 'fall',
'this', ',', 'I', '\nshall', 'think', 'nothing', 'tumbling', 'down', 'stairs', '!', 'How', 'brave', 'they', "'ll", '\na
ll', 'think', 'me', 'home', '!', 'Why', ',', 'I', 'wouldn', "'t", 'say', ',', 'I', 'fell', 'the', 'top', 'the', 'hous
e', '!', 'Which', 'was', 'very', 'likely', '\ntrue', '.', 'Down', ',', 'down', ',', 'down', '.', 'Would', 'the', 'fal
l', 'NEVER', 'come', 'to', '!', 'I', '\nwonder', 'how', 'many', 'miles', 'I', "'ve", 'fallen', 'by', 'this', 'time',
'?', 'she', 'said', '.', 'I', 'must', 'be', 'getting', 'somewhere', 'near', 'the', 'centre', 'the', '.', 'Let', '\nme',
'see', ':', 'that', 'would', 'be', 'four', 'thousand', 'miles', 'down', ',', 'I', 'think', '-', 'for', ',', 'you', 'se
e', ',', 'Alice', 'had', 'learnt', 'several', 'things', 'this', 'sort', 'her', '\nlessons', 'the', 'schoolroom', ',',
'though', 'this', 'was', 'not', 'VERY', 'good', '\nopportunity', 'for', 'showing', 'her', 'knowledge', ',', 'there', 'w
as', 'no', 'to', '\nlisten', 'to', 'her', ',', 'still', 'was', 'good', 'practice', 'to', 'say', ')', 'yes', ',', 'tha
t', "'s", 'the', 'right', 'distance', '-', 'but', 'then', 'I', 'wonder', 'what', 'Latitude', '\nor', 'Longitude', 'I',
"'ve", 'got', 'to', '?', 'Alice', 'had', 'no', 'what', 'Latitude', 'was', ',', 'Longitude', ',', 'but', 'thought', 'the
y', 'were', 'nice', 'grand', 'words', 'to', '\nsay', '.', 'Presently', 'she', 'began', '.', 'I', 'wonder', 'I', 'shal
l', 'fall', 'right', '\nTHROUGH', 'the', '!', 'How', 'funny', "'ll", 'seem', 'to', 'come', 'the', '\npeople', 'that',
'walk', 'with', 'their', 'heads', 'downward', '!', 'The', 'Antipathies', ',', 'I', '\nthink', '-', 'she', 'was', 'rathe
r', 'glad', 'there', 'WAS', 'no', 'listening', ',', 'this', '\ntime', ',', 'didn', "'t", 'sound', 'the', 'right', 'wor
d', ')', 'but', 'I', 'shall', '\nhave', 'to', 'them', 'what', 'the', 'name', 'the', 'country', ',', 'you', 'know', '.',
'Please', ',', 'Ma', "'am", ',', 'this', 'New', 'Zealand', 'Australia', '?', 'she', 'tried', '\nto', 'curtsey', 'she',
'spoke', '-', 'fancy', 'CURTSEYING', 'you', "'re", 'falling', '\nthrough', 'the', '!', 'Do', 'you', 'think', 'you', 'co
uld', 'manage', '?', 'And', 'what', '\nan', 'little', 'girl', 'she', "'ll", 'think', 'me', 'for', '!', 'No', ',', "'l
l", '\nnever', 'do', 'to', ':', 'perhaps', 'I', 'shall', 'see', 'written', 'somewhere', '.', 'Down', ',', 'down', ',',
'down', '.', 'There', 'was', 'nothing', 'to', 'do', ',', 'so', 'Alice', 'soon', '\nbegan', 'talking', '.', 'Dinah', "'l
l", 'miss', 'me', 'very', 'much', 'to', '-night', ',', 'I', '\nshould', 'think', '!', 'Dinah', 'was', 'the', 'cat',
'.', 'I', 'hope', 'they', "'ll", 'remember', '\nher', 'saucer', 'milk', 'tea', '-time', '.', 'Dinah', 'my', 'dear',
'!', 'I', 'wish', 'you', 'were', '\ndown', 'here', 'with', 'me', '!', 'There', 'no', 'mice', 'the', ',', 'I', "'m",
',', 'but', '\nyou', 'might', 'catch', 'bat', ',', 'that', "'s", 'very', 'like', 'mouse', ',', 'you', 'know', '.', 'Bu
t', 'do', 'cats', 'bats', ',', 'I', 'wonder', '?', 'And', 'here', 'Alice', 'began', 'to', 'get', '\nrather', 'sleepy',
',', 'went', 'saying', 'to', 'herself', ',', 'dreamy', 'sort', '\nway', ',', 'Do', 'cats', 'bats', '?', 'Do', 'cats',
'bats', '?', 'sometimes', ',', 'Do', '\nbats', 'cats', '?', 'for', ',', 'you', 'see', ',', 'she', 'couldn', "'t", '\nqu
estion', ',', 'didn', "'t", 'much', 'matter', 'which', 'way', 'she', 'put', '.', 'She', 'felt', '\nthat', 'she', 'was',
'dozing', ',', 'had', 'just', 'begun', 'to', 'dream', 'that', 'she', '\nwas', 'walking', 'hand', 'hand', 'with', 'Dina
h', ',', 'saying', 'to', 'her', 'very', '\nearnestly', ',', 'Now', ',', 'Dinah', ',', 'tell', 'me', 'the', 'truth',
':', 'did', 'you', '\nbat', '?', 'when', 'suddenly', ',', 'thump', '!', 'thump', '!', 'down', 'she', 'came', 'heap',
'\nsticks', 'dry', 'leaves', ',', 'the', 'fall', 'was', '.', 'Alice', 'was', 'not', 'bit', 'hurt', ',', 'she', 'jumpe
d', 'to', 'her', 'feet', '\nmoment', ':', 'she', 'looked', ',', 'but', 'was', 'dark', ';', 'before', 'her', '\nwas', 'l
ong', 'passage', ',', 'the', 'White', 'Rabbit', 'was', 'still', '\nsight', ',', 'hurrying', 'down', '.', 'There', 'wa
s', 'not', 'moment', 'to', 'be', 'lost', ':', 'went', 'Alice', 'like', 'the', 'wind', ',', 'was', 'just', 'time', 'to',
'hear', '\nsay', ',', 'turned', 'corner', ',', 'Oh', 'my', 'whiskers', ',', 'how', 'late', '\nit', "'s", 'getting',
'!', 'She', 'was', 'close', 'behind', 'when', 'she', 'turned', 'the', '\ncorner', ',', 'but', 'the', 'Rabbit', 'was',
'no', 'longer', 'to', 'be', 'seen', ':', 'she', 'found', '\nherself', 'long', ',', 'low', 'hall', ',', 'which', 'was',
'lit', 'by', 'row', 'lamps', '\nhanging', 'from', 'the', 'roof', '.', 'There', 'were', 'doors', 'round', 'the', 'hall',
',', 'but', 'they', 'were', 'locked', ';', 'when', 'Alice', 'had', 'been', 'the', 'way', 'down', 'side', 'the', '\nothe
r', ',', 'trying', 'door', ',', 'she', 'walked', 'sadly', 'down', 'the', 'middle', ',', 'wondering', 'how', 'she', 'wa
s', 'to', 'get', '.', 'Suddenly', 'she', 'came', 'little', 'three', '-legged', 'table', ',', 'made', '\nsolid', 'glas
s', ';', 'there', 'was', 'nothing', 'tiny', 'golden', 'key', ',', 'Alice', "'s", 'first', 'thought', 'was', 'that', 'mi
ght', 'belong', 'to', 'the', '\ndoors', 'the', 'hall', ';', 'but', ',', '!', 'the', 'locks', 'were', 'too', 'large',
',', '\nthe', 'key', 'was', 'too', 'small', ',', 'but', 'rate', 'would', 'not', '\nthem', '.', 'However', ',', 'the',
'second', 'time', 'round', ',', 'she', 'came', 'low', '\ncurtain', 'she', 'had', 'not', 'noticed', 'before', ',', 'behi
nd', 'was', 'little', '\ndoor', 'fifteen', 'high', ':', 'she', 'tried', 'the', 'little', 'golden', 'key', '\nin', 'th
e', 'lock', ',', 'to', 'her', 'great', 'delight', 'fitted', '!', 'THE', 'END']
```

In [78]:

```python
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
# print(text)
reg = r'\b[aeiou]\w{2}'
ans = re.findall(reg,text)
print(ans)
```

```
['and', 'onc', 'int', 'and', 'use', 'own', 'and', 'and', 'eye', 'out', 'its', 'ove', 'aft', 'occ', 'oug', 'all', 'act',
'and', 'and', 'acr', 'eit', 'out', 'and', 'acr', 'aft', 'and', 'und', 'ano', 'aft', 'onc', 'out', 'aga', 'and', 'abo',
'abo', 'and', 'and', 'out', 'any', 'and', 'and', 'and', 'and', 'upo', 'one', 'emp', 'int', 'one', 'aft', 'all', 'any',
'abo', 'eve', 'off', 'end', 'alo', 'ear', 'and', 'opp', 'off', 'one', 'ove', 'abo', 'ide', 'eit', 'aga', 'ear', 'out',
'amo', 'one', 'all', 'ask', 'and', 'air', 'ign', 'ask', 'ask', 'els', 'aga', 'are', 'air', 'afr', 'and', 'eat', 'and',
'eat', 'eat', 'and', 'eat', 'ans', 'eit', 'off', 'and', 'and', 'ear', 'eve', 'eat', 'upo', 'and', 'and', 'ove', 'and',
'all', 'ove', 'ano', 'and', 'awa', 'and', 'ear', 'and', 'all', 'all', 'and', 'all', 'one', 'and', 'oth', 'eve', 'eve',
'out', 'aga', 'upo', 'all', 'exc', 'and', 'one', 'ala', 'eit', 'any', 'ope', 'any', 'upo', 'and', 'abo', 'inc', 'and']
```

In [79]:

```python
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
# print(text)
reg = r'\b[aeiou]\w{2}\b'
ans = re.findall(reg,text)
print(ans)
```

```
['and', 'and', 'use', 'own', 'and', 'and', 'out', 'all', 'and', 'and', 'out', 'and', 'and', 'out', 'and', 'and', 'and',
'out', 'and', 'and', 'and', 'and', 'one', 'one', 'all', 'off', 'end', 'and', 'off', 'one', 'out', 'one', 'all', 'ask',
'and', 'air', 'ask', 'are', 'air', 'and', 'eat', 'and', 'eat', 'eat', 'and', 'eat', 'off', 'and', 'and', 'eat', 'and',
'and', 'and', 'all', 'and', 'and', 'and', 'all', 'all', 'and', 'all', 'one', 'and', 'out', 'all', 'and', 'one', 'any',
'any', 'and', 'and']
```

In [2]:

```python
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
reg = r'\b[aeiou]\w{2,5}\b'
ans = re.findall(reg,text)
print(ans)
# min Length 3 to 5
```

```
['and', 'once', 'into', 'and', 'use', 'own', 'and', 'and', 'eyes', 'out', 'itself', 'over', 'ought', 'all', 'and', 'an
d', 'across', 'either', 'out', 'and', 'across', 'after', 'and', 'under', 'after', 'once', 'out', 'again', 'and', 'abou
t', 'about', 'and', 'and', 'out', 'and', 'and', 'and', 'and', 'upon', 'one', 'empty', 'into', 'one', 'after', 'all', 'a
bout', 'even', 'off', 'end', 'aloud', 'earth', 'and', 'off', 'one', 'over', 'about', 'idea', 'either', 'again', 'eart
h', 'out', 'among', 'one', 'all', 'ask', 'and', 'air', 'asking', 'ask', 'else', 'again', 'are', 'air', 'afraid', 'and',
'eat', 'and', 'eat', 'eat', 'and', 'eat', 'answer', 'either', 'off', 'and', 'and', 'ever', 'eat', 'upon', 'and', 'and',
'over', 'and', 'all', 'and', 'away', 'and', 'ears', 'and', 'all', 'all', 'and', 'all', 'one', 'and', 'other', 'every',
'ever', 'out', 'again', 'upon', 'all', 'except', 'and', 'one', 'alas', 'either', 'any', 'open', 'any', 'upon', 'and',
'about', 'inches', 'and']
```

In [3]:

```python
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
reg = r'\b[aeiou]\w{3,}\b'
ans = re.findall(reg,text)
print(ans)
# Length min 3 or more
```

```
['once', 'into', 'eyes', 'itself', 'over', 'afterwards', 'occurred', 'ought', 'actually', 'across', 'either', 'across',
'after', 'under', 'another', 'after', 'once', 'again', 'about', 'about', 'anything', 'upon', 'empty', 'into', 'after',
'anything', 'about', 'even', 'aloud', 'earth', 'opportunity', 'over', 'about', 'idea', 'either', 'again', 'earth', 'amo
ng', 'ignorant', 'asking', 'else', 'again', 'afraid', 'answer', 'either', 'earnestly', 'ever', 'upon', 'over', 'overhea
d', 'another', 'away', 'ears', 'other', 'every', 'ever', 'again', 'upon', 'except', 'alas', 'either', 'open', 'upon',
'about', 'inches']
```

In [5]:

```
import re
f = open('alice_in_wonderland.txt', encoding='utf-8')
text = f.read()
reg = r'\b[aeiou]\w{,4}\b'
ans = re.findall(reg,text)
print(ans)
# Length max 4
```

```
['in', 'of', 'on', 'and', 'of', 'once', 'or', 'into', 'it', 'or', 'in', 'it', 'and', 'is', 'use', 'of', 'a', 'or', 'i
n', 'own', 'as', 'as', 'and', 'of', 'a', 'of', 'up', 'and', 'a', 'eyes', 'in', 'it', 'out', 'of', 'it', 'over', 'it',
'ought', 'at', 'at', 'it', 'all', 'and', 'at', 'it', 'and', 'on', 'it', 'a', 'a', 'or', 'a', 'out', 'of', 'it', 'and',
'after', 'it', 'and', 'in', 'it', 'a', 'under', 'after', 'it', 'once', 'in', 'out', 'again', 'on', 'a', 'and', 'a', 'ab
out', 'a', 'or', 'of', 'as', 'about', 'and', 'and', 'out', 'it', 'at', 'of', 'and', 'and', 'and', 'and', 'upon', 'a',
'one', 'of', 'as', 'it', 'it', 'empty', 'of', 'it', 'into', 'one', 'of', 'as', 'it', 'after', 'a', 'as', 'of', 'all',
'at', 'about', 'it', 'even', 'if', 'off', 'of', 'an', 'end', 'aloud', 'of', 'earth', 'of', 'in', 'in', 'and', 'a', 'of
f', 'as', 'one', 'it', 'it', 'over', 'about', 'or', 'idea', 'or', 'again', 'if', 'earth', 'it', 'out', 'among', 'one',
'as', 'it', 'at', 'all', 'ask', 'of', 'is', 'am', 'is', 'or', 'and', 'as', 'as', 'air', 'it', 'an', 'it', 'ask', 'it',
'up', 'else', 'again', 'of', 'at', 'are', 'in', 'air', 'a', 'and', 'a', 'eat', 'and', 'on', 'in', 'a', 'of', 'eat', 'ea
t', 'and', 'eat', 'as', 'it', 'it', 'off', 'and', 'in', 'and', 'ever', 'eat', 'a', 'upon', 'a', 'of', 'and', 'and', 'ov
er', 'a', 'and', 'up', 'on', 'in', 'a', 'up', 'it', 'all', 'and', 'in', 'it', 'a', 'away', 'and', 'in', 'it', 'as', 'i
t', 'a', 'ears', 'and', 'it', 'it', 'in', 'a', 'up', 'a', 'of', 'all', 'all', 'and', 'all', 'one', 'and', 'up', 'othe
r', 'every', 'ever', 'out', 'again', 'upon', 'a', 'all', 'of', 'on', 'it', 'a', 'and', 'it', 'one', 'of', 'of', 'alas',
'or', 'at', 'any', 'it', 'open', 'any', 'of', 'on', 'upon', 'a', 'and', 'it', 'a', 'about', 'in', 'and', 'it']
```

In [8]:

```
import re
text = 'Python is $Programming'
reg = r'\W'
ans = re.findall(reg, text)
print(ans)
```

```
[' ', ' ', '$']
```

In [10]:

```
import re
text = 'Python is $Programming'
reg = r'$'
ans = re.findall(reg, text)
print(ans)
```

```
['']
```

In [11]:

```
import re
text = 'Python is $Programming'
reg = r'\$'
ans = re.findall(reg, text)
print(ans)
```

```
['$']
```

In [22]:

```
import re
text = 'Python** is*** Pro'
reg = r'\w+\W?'
ans = re.findall(reg, text)
print(ans)
```

```
['Python*', 'is*', 'Pro']
```

In [24]:

```
import re
text = 'Python** is*** Pro'
reg = r'\W?'
ans = re.findall(reg, text)
print(ans)
```

```
['', '', '', '', '', '', '*', '*', ' ', '', '', '*', '*', '*', ' ', '', '', '', '']
```

In [25]:

```
import re
text = 'Python** is*** Pro'
reg = r'\w?'
ans = re.findall(reg, text)
print(ans)
```

```
['P', 'y', 't', 'h', 'o', 'n', '', '', '', 'i', 's', '', '', '', '', 'P', 'r', 'o', '']
```

In [ ]:

```

```

In [ ]:

```

```