Jaccard Similarity Coefficient

โจทย์นี้มีข้อความที่คุณ Joseph Robinette Biden Jr. ทวีตไว้ไม่นานมานี้[จำนวนสามพันกว่าทวีต](#) โปรแกรมที่จะมาพัฒนากันจะอ่านข้อความค้นจากผู้ใช้ แล้วหาว่า ทวีตใดมีเนื้อหา "คล้าย" กับข้อความค้นที่สุดจำนวน n ทวีต

มีประเด็นที่ต้องคำนึงถึงในการเขียนโปรแกรมค้นข้อความในลักษณะนี้ ดังนี้

- คำหลายคำในข้อความไม่ค่อยน่าสนใจ เช่น a, an, the, in, on, ...
  เราเรียกคำเหล่านี้ว่า stop words ซึ่งจะถูกลบออกจากข้อความต่าง ๆ เพื่อเพิ่มความแม่นยำในการวัดความคล้าย และยังประหยัดเนื้อที่เก็บอีกด้วย
- คำอังกฤษแต่ละคำเขียนได้หลายรูปแบบ เช่น play, playing, played, ...
  คำต่าง ๆ ในข้อความจะถูกแปลงให้เป็น รากศัพท์ (stem) ของคำนั้น เพื่อช่วยเพิ่มความแม่นยำในการวัดความคล้าย
- การนิยามความคล้ายของสองข้อความใด ๆ
  โจทย์นี้ใช้ Jaccard similarity coefficient (https://en.wikipedia.org/wiki/Jaccard_index) ในการวัดความคล้าย
  กำหนดให้ S กับ T เป็นสองข้อความที่ต้องการวัดความคล้าย

$$J(S,T) = \frac{\text{จำนวนคำที่ปรากฏในทั้ง } S \text{ และ } T}{\text{จำนวนคำทั้งหมดใน } S \text{ และ } T \text{ รวมกัน (ไม่ซ้ำกัน)}}$$

ตัวอย่าง:

S_0 = " Which countries are open to vaccinated travellers?" นำมาแยกเป็นคำ ๆ และตัด stop words ออก จะได้

S_1 = ['countries', 'open', 'vaccinated', 'travellers'] นำแต่ละคำไปแปลงเป็นรากศัพท์ จะได้

S = ['countri', 'open', 'vaccin', 'travel']

และอีกข้อความ

T_0 = " Phuket's plan to welcome vaccinated travelers as soon as this fall" แยกเป็นคำ, ตัด stop words ออก จะได้

T_1 = ['phuket', 'plan', 'welcome', 'vaccinated', 'travelers', 'soon', 'fall'] นำแต่ละคำไปแปลงเป็นรากศัพท์ จะได้

T = ['phuket', 'plan', 'welcom', 'vaccin', 'travel', 'soon', 'fall']

พิจารณา S กับ T ได้

- คำที่ปรากฏทั้งใน S และ T คือ ['vaccin', 'travel']
- คำทั้งหมดใน S และ T รวมกัน คือ ['countri', 'vaccin', 'open', 'welcom', 'travel', 'phuket', 'soon', 'plan', 'fall']
- $J(S, T) = \frac{2}{9} = 0.22$

หมายเหตุ: รากศัพท์ที่แสดงอาจดูแปลก ๆ อันนี้ได้ผลมาจากฟังก์ชันที่เราจะใช้กันต่อไปที่ทำงานแบบง่าย ๆ

ตัวอย่างการใช้งานและการแสดงผล

ก่อนจะลงในรายละเอียด มาดูผลการสั่งทำงาน เมื่อเขียนโปรแกรมที่ทำงานถูกต้อง

```
>>> %Run jaccard.py
Query words   : COVID economic crisis
No. of results: 3

#1599 (0.22)
  180,000 people have died from this COVID
  crisis. And the heartbreaking truth is that it
  didn't have to happen this way.

#2907 (0.22)
  You can't deal with the economic crisis until
  you deal with the public health crisis. It's a
  false choice.

#1767 (0.2)
  What kind of president tries to defund Social
  Security during an economic crisis?
  https://t.co/FO47IjABTF
------------------------------------------------
Query words   : health care policy
No. of results: 5

#1295 (0.4)
  Health care is a right – not a privilege.

#2163 (0.4)
  Health care is a right for all – not a
  privilege for the few.

#623 (0.29)
  Vote like your health care is on the ballot –
  because it is.  https://t.co/eoxT07d7QB

#850 (0.29)
  Vote like your health care is on the ballot –
  because it is.   https://t.co/eoxT07uII9

#720 (0.22)
  It’s simple: Donald Trump thinks health care
  is a privilege. I think it is your right.
------------------------------------------------
Query words   : american president
No. of results: 5

#995 (0.5)
  I’ll be a president for all Americans. Not
  just the ones who vote for me.

#846 (0.4)
  I’m running as a proud Democrat. But I will be
  an American president.

#957 (0.4)
  Over 215,000 Americans have died from
  COVID-19, and what is President Trump doing?
  Nothing.

#2601 (0.4)
  This crisis isn't about you, Mr. President.
  It's about the American people.
  https://t.co/3a4ppJ964Y

#2668 (0.4)
  The health of the American people is not a
  joke, Mr. President.
------------------------------------------------
Query words   :
>>>
```

ต้องการค้นทวีตที่มีข้อความคล้ายคำเหล่านี้ จำนวน 3 ทวีต

ตรงนี้กด Enter เลย จะทำให้โปรแกรมเลิกการทำงาน

โปรแกรมต้นฉบับ

```
# Prog-06: Jaccard Similarity
# ??3??????21 Name ?
```

ใส่เลขประจำตัว ชื่อ นามสกุล

download code นี้ได้

ต้อง download ข้อมูลทดสอบนี้ biden.txt มาวางใน folder เดียวกับโปรแกรมก่อน run

โปรแกรมที่ส่ง ห้ามเปลี่ยนโค้ดส่วนที่เป็นสีแดงโดยเด็ดขาด เปลี่ยนได้เฉพาะส่วนที่มีพื้นหลังเป็นสีเขียวเท่านั้น

```
from nltk.stem import PorterStemmer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

STOP_WORDS = stopwords.words('english')
STEMMER = PorterStemmer()

def read_tweets():
    f = open('biden.txt', encoding='utf-8')
    tweets = [line.strip() for line in f.readlines()]
    f.close()
    return tweets

def normalize_text( text ):
    words = []
    for w in word_tokenize(text.lower()):
        if w.isalnum() and w not in STOP_WORDS:
            words.append(STEMMER.stem(w))
    return get_unique( words )

def main():
    tweets = read_tweets()
    norm_tweets = []
    for t in tweets:
        norm_tweets.append( normalize_text(t) )

    print_width = 48
    while True:
        query = input('Query words   : ')
        if query == '': break
        n = int(input('No. of results: '))
        norm_query = normalize_text(query)
        top_n = top_n_similarity(norm_tweets, norm_query, n)
        if len(top_n) == 0:
            print('No matches found.')
        else:
            for tid, jc_coef in top_n:
                show_tweet(tid, tweets[tid], jc_coef, print_width)
        print('-' * print_width)

#-----------------------------------------------------
def get_unique( words ):
    

def jaccard(words_1, words_2):
    

def top_n_similarity(norm_tweets, norm_query, n):
    

def show_tweet(tweet_id, tweet_content, jc_coef, print_width):
    

#-----------------------------------------------
main()
```

รูปแบบข้อมูล

```
tweets = [ str0, str1, ... ]

norm_tweets =
  [
    [ w00, w01, ... ],
    [ w10, w11, ... ],
    [ w20, w21, ... ],
    ...
  ]

query = 'a string'

norm_query = [ w0, w1, ... ]

top_n = [
          [23, 0.8],
          [14, 0.7],
          [300, 0.7],
          ...
        ]
```

ห้าม import อะไรเพิ่มเติม และห้ามใช้ที่เก็บข้อมูลจำพวก set หรือ dict หรือที่คล้ายคลึงโดยเด็ดขาด

คำอธิบายโปรแกรมต้นฉบับ

โปรแกรมต้นฉบับมีฟังก์ชันที่ใช้งานได้หลายฟังก์ชัน ดังนี้

ฟังก์ชัน	ตัวอย่างการใช้งาน
read_tweets() เรียกฟังก์ชันนี้แล้วจะได้ลิสต์ของข้อความทวีตที่ใช้ทดสอบการทำงานของโปรแกรม	**tweets = read_tweets()**
normalize_text(text) นำ text ไปแยกเป็นคำ, ตัด stop words, แปลงเป็นรากศัพท์ และลบคำซ้ำกันออก ได้ผลลัพธ์เป็นลิสต์	**t = 'Venus Williams and Serena Williams ' +** **'are two professional tennis players'** **nt = normalize_text(t)** ได้ **nt** เก็บ **['player', 'profession', 'serena',** **'tenni', 'two', 'venu', 'william']**

main()

มีขั้นตอนการทำงานดังนี้

- ใช้ **read_tweets()** อ่านข้อมูลทวีตทั้งหมดจากแฟ้มมาเก็บในลิสต์ชื่อว่า **tweets**
- นำแต่ละทวีตใน **tweets** ไปผ่าน **normalize_text** ได้เป็นลิสต์ของรากศัพท์ของทวีตนั้น แล้วไปเก็บในลิสต์ **norm_tweets**
- เข้าสู่วงวนที่ทำงานวนไปเรื่อย เพื่อจะรับข้อความค้น ไปค้นทวีต เมื่อใดที่ข้อความค้น มีความยาวเป็นศูนย์ ก็ออกจากวงวน
- ภายในวงวน หลังจากรับข้อความค้นเก็บใน **query** แล้ว ก็รับจำนวนเต็มอีกตัวเก็บใน **n**
  - นำ **query** ไปผ่าน **normalize_text** ได้เป็นลิสต์ของรากศัพท์ของ **query** เก็บใน **norm_query**
  - ส่ง **norm_tweets**, **norm_query** และ **n** ไปให้ **top_n_similarity** เพื่อค้นทวีตใน **norm_tweets** ที่คล้ายกับ **norm_query** ที่สุด **n** อันดับแรก ได้ผลกลับเก็บใน **top_n** เป็นลิสต์ที่เก็บ **[**เลขลำดับทวีต**,** ค่า **Jaccard]** ของทวีตที่คล้ายสุดที่หาได้ **n** ทวีต (หรือน้อยกว่า)
  - ถ้า **top_n** ไม่มีข้อมูล ก็แจ้งว่าไม่พบทวีตที่คล้ายเลย แต่ถ้ามีข้อมูล ก็ใช้ **show_tweet** แสดงข้อมูลแต่ละทวีตใน **top_n**

สิ่งที่ต้องทำ

ฟังก์ชันที่ต้องเขียนให้สมบูรณ์มีดังนี้

```
def get_unique( words ):

    # words เป็นลิสต์ที่เก็บสตริง

    # ต้องทำ: ตั้งค่าให้ตัวแปร unique_words ที่เก็บสตริงได้มาจาก words แต่ไม่มีตัวซ้ำ
              (คือตัวไหนมีซ้ำใน words จะมีตัวนั้นแค่ตัวเดียวใน unique_words
    # ตัวอย่าง: words เก็บ ['x', 'y', 'z', 'y', 'xyz', 'z']
    #          จะได้ unique_words เก็บ ['x', 'y', 'z' , 'xyz']
    #          ลำดับซ้ายขวาของสตริงใน unique_words อาจต่างไปจากที่ปรากฏใน words ก็ได้

    return unique_words
```

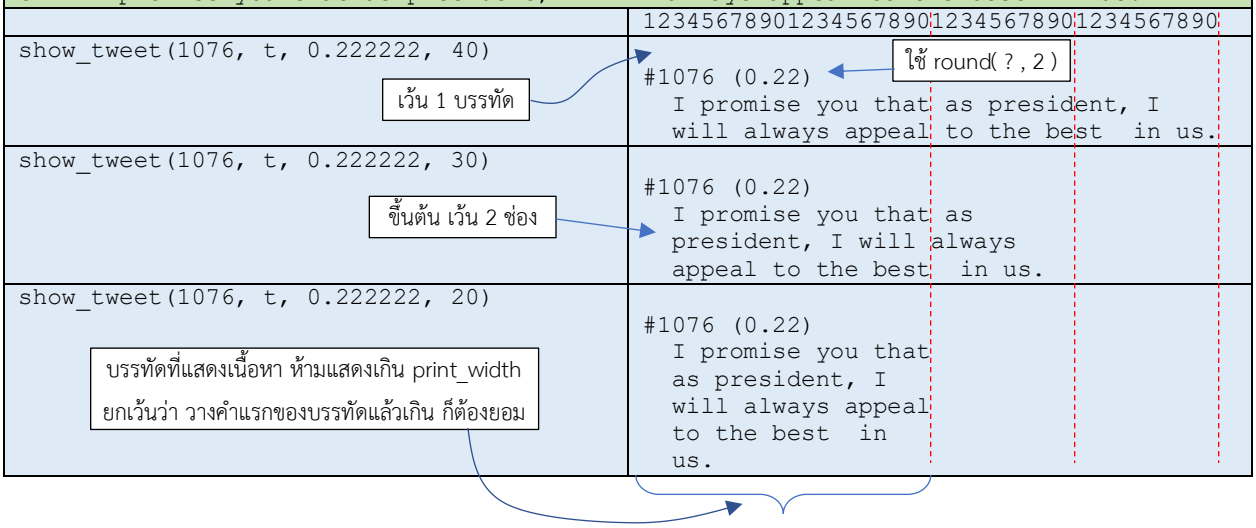
```
def jaccard(words_1, words_2):
    # words_1 และ words_2 เป็นลิสต์ของคำต่าง ๆ  (ไม่มีคำซ้ำใน words_1 และ ไม่มีคำซ้ำใน words_2)

    # ต้องทำ: ตั้งตัวแปร jaccard_coef ให้มีค่าเท่ากับ Jaccard similarity coefficient
    #         ที่คำนวณจากค่าใน words_1 และ words_2 ตามสูตรที่แสดงไว้ก่อนนี้
    # ตัวอย่าง: words_1 เก็บ ['x', 'y', 'z', 'xyz'] และ words_2 เก็บ ['y', 'x', 'w']
    #         จะได้ jaccard_coef มีค่าเท่ากับ 2/5 = 0.4

    return jaccard_coef
```

```
def top_n_similarity(norm_tweets, norm_query, n):
    # norm_tweets เป็นลิสต์ที่ภายในเก็บลิสต์ของคำต่าง ๆ  [ [w00,w01,...], [w10,w11,...], ... ]
    # norm_query  เป็นลิสต์ของคำต่าง ๆ
    # n           เป็นจำนวนเต็ม

    # ต้องทำ: ตั้งค่าให้ตัวแปร top_n ที่เก็บลิสต์ขนาดไม่เกิน n ช่อง แต่ละช่องเก็บลิสต์ย่อยขนาดสองช่อง
    #             [  [tweet_id, jaccard],  ...  ]
    #         tweet_id คือเลขอินเด็กซ์ของทวีตใน norm_tweets
    #         jaccard คือค่า Jaccard coefficient ของ norm_tweets[tweet_id] กับ norm_query
    #         โดยจะเลือกทวีตที่มีค่า Jaccard มากกว่า 0 และติดอันดับมากสุด n ตัวแรก ในกรณีที่มีค่า Jaccard
    #         เท่ากัน ให้เลือกอันที่มี tweet_id น้อยกว่าก่อน

    return top_n
```

```
def show_tweet(tweet_id, tweet_content, jc_coef, print_width)
    # tweet_id      เป็นจำนวนเต็มแทนเลขอินเด็กซ์ของทวีต
    # tweet_content เป็นสตริงเก็บข้อความของทวีตที่ต้องการแสดง
    # jc_coef       เป็นจำนวนจริงแทนค่า Jaccard coefficient
    # print_width   เป็นจำนวนเต็มแทนจำนวนตัวอักษรที่แสดงได้ในหนึ่งบรรทัด
    # ต้องทำ: นำข้อมูลทั้งหลายที่ได้รับมาแสดงทางจอภาพในรูปแบบที่แสดงในตัวอย่าง ฟังก์ชันนี้ไม่คืนผลอะไร
    #         การแยก tweet_content ออกเป็นคำ ๆ อาศัย split(' ') ได้เลย
    # ตัวอย่าง:
```

```
t = 'I promise you that as president, I will always appeal to the best  in us.'
```

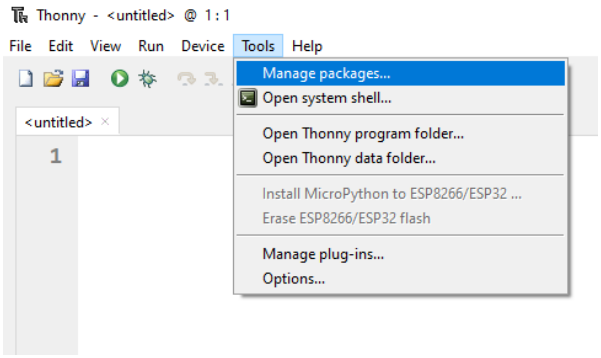
```
                                        12345678901234567890123456789012345678901234567890
show_tweet(1076, t, 0.222222, 40)       #1076 (0.22)
                                          I promise you that as president, I
                                          will always appeal to the best  in us.
show_tweet(1076, t, 0.222222, 30)       #1076 (0.22)
                                          I promise you that as
                                          president, I will always
                                          appeal to the best  in us.
show_tweet(1076, t, 0.222222, 20)       #1076 (0.22)
                                          I promise you that
                                          as president, I
                                          will always appeal
                                          to the best  in
                                          us.
```



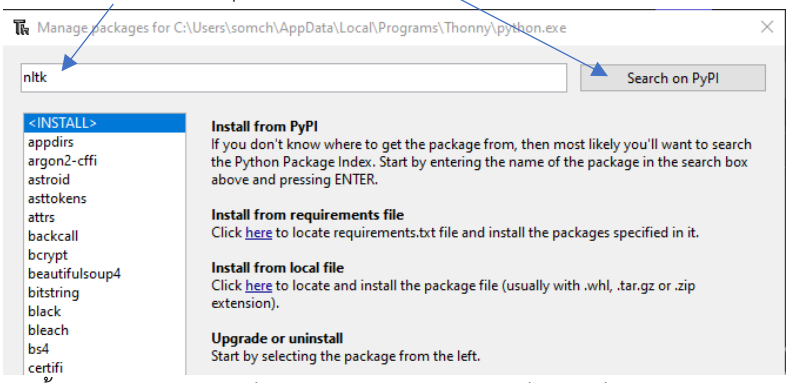
ต้องติดตั้ง NLTK ก่อน

โปรแกรมในโจทย์นี้ใช้ฟังก์ชันใน package ชื่อ NLTK (Natural Language Toolkit) ที่อำนวยความสะดวกในการแยกคำ หารากศัพท์ และมีรายการ stop words ให้ใช้ ดังนั้น ต้องติดตั้ง NLTK ก่อน ทำได้ดังนี้

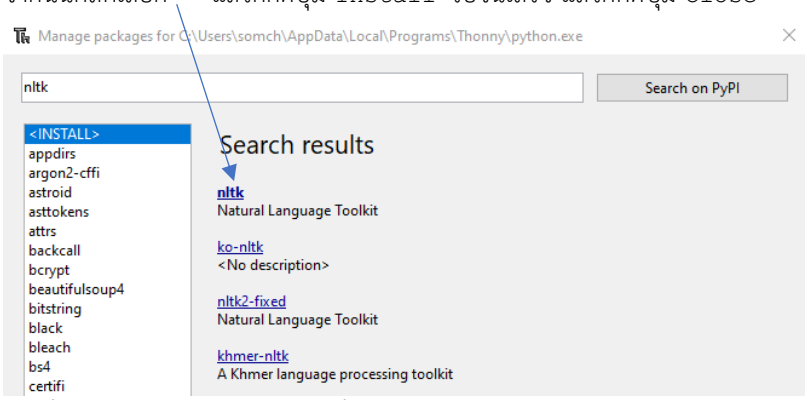
- ใน `Thonny` เลือกเมนู `Tools -> Manage packages`



- ใส่คำว่า **`nltk`** และกดปุ่ม **`Search on PyPI`**



- จากนั้นคลิกเลือก แล้วก็กดปุ่ม `Install` รอจนเสร็จ แล้วก็กดปุ่ม `Close`



- จากนั้นต้องพิมพ์สามคำสั่งดังต่อไปนี้ที่ `shell` ของ `Thonny` แล้วก็เริ่มเขียนโปรแกรมของการบ้านนี้ได้