一、執行環境：terminal

二、程式語言：python3

三、執行方式：

terminal 執行 python3 b06705039_hw4.py

使用套件 nltk, string, os, math, numpy

四、作業處理邏輯：

(有使用heap, 參考改編網路上的實作heap, 最主要為了加速用)

global function:

最主要給MaxHeap用的function

- _swap：把陣列中的兩個值交換
- _sift_up：給一個陣列跟index, 從index開始檢查parent（(index-1)/2）, 如果parent的similarity比自己大或是已經沒有parent，就結束，不然就與parent換位置，繼續叫_sift_up。
- _sift_down：給一個陣列跟index, 從index開始看檢查child（index*2+1 & index*2+2），如果有兩個child就挑出similarity比較大的child，只有一個child就直接拿來比對，看有沒有比自己的similarity小，如果比較小就結束，比較大就與child換位置，繼續叫_sift_down。

main 所使用的global function

- classifyK (要分的群數K , 陣列A）：A為一個pair裡面是兩個index, 代表他們的similarity在被選出時擁有最大的similarity。對A裡每個pair, 在 文件數-K 步前，去看是否存在我已經分好群的陣列中, 如果在一個群裡找到其中一個index, 就把兩個index都加入此群，如果在兩個群中找到index，把後面的群加入前面的群，刪除後面的群，如果沒有在任何一個群中找到這兩個index, 就創一個新的群放這兩個index。

  文件數-K 步後，其中一個index出現在某群中，另一個index就自己創一群，如果兩個index都沒有出現在任何群中，就創兩個群分別塞這兩個index。

  回傳分好群的陣列 (每個index都存一個set)
- toTerm (filename): tokenize, lowercase, stemming, stopword, 對特別小文字做處理（要移掉網址）
- writeFile (array, filename): 把陣列中的每個set轉乘list, list.sort()排序，寫入文件中，寫完一個index中的set後，空一行繼續印下一個index中的set。

implement class heap:

- initial : 設一個像 private member的變數 _heap （陣列）, 可以在initial時直接加入值 ex, x = MaxHeap(similarity, index) 或 x = MaxHeap()。
- push：會push進一個 [similarity, index]的pair, 把他 append到private member _heap的最後一個位置，使用 _sift_up(self._heap, 最後一個index), 來動到適合的位置。
- pop：先用 _swap 把最後一個值跟第一個值交換，把最後一個node刪掉，最後把移動到top的值用 _sift_down移動到它的位置。

- getMax：回傳第一個值（一個pair）, [Max similarity, index]
- len：回傳此_heap, 這個 array的長度
- delete：先搜尋 _heap裡每個 node的 index找到我們想刪的 index, 找到後把這個 node跟最後一個node交換, 把交換過來的node往上先移到parent已經不比這個node小，再往下移動，移到此node該在的位置。

main:

step1：用toTerm讀入所有文檔,
對每個文件儲存term & frequency,
和存dictionary

step2：掃全部的文件, 把dictionary裡的字去做檢查存成 idf,
存一個tf陣列把每個文件中相同term的frequency存起來。

step3：對全部文件，全部的term frequency和自己本身的存成dictionary維度的陣列，裡面的值為normalize的 tf

step4：設一個陣列 I, 表示index是否代表一個cluster,
對每個文件也設一個heap, 算出與其他文件的similarity, 存similarity & index進heap。

step5：要找出每一層最大的similarity pair, 存成陣列A,
執行文 件數-1 步, 每一步去看每個文件對應的heap, 找出最大的heap top值，
把文件index i與對應到的最大similarity的文件index j存進 A ,
對每個文件的heap進行更新，刪除index 為 i, j 的兩個node,
對每個文件的heap存入similarity（使用complete link），
(有使用過single link但效果不太好，會有一個超級大群，其他可能都只擁有一個index)
更新儲存文件對所有cluster 的similarity的陣列。

step6：使用classifyK, 得到分成8, 13, 20群的陣列,
利用writeFile把這三個陣列分別存成三個txt檔。