標註執行環境：Visual Studio Code

程式語言：Python 3.11

執行方式：

1. 準備文件：將待處理的1095個txt檔案放入名為data的資料夾。
2. 安裝 NLTK：打開命令行輸A入pip install nltk (用於import porter stemming)。
3. 執行程式：在命令行中輸入python pa4.py。
4. 查看輸出：cluster結果會被保存到名為 20.txt, 13.txt, 8.txt 的文件中。

作業處理邏輯說明：

1. 讀取訓練和測試資料：
    - 使用 os.listdir() 函數讀取 data 資料夾中的所有文本文件。
    - 根據文件名中的數字 ID，將每個文件的內容讀取並處理。
2. Tokenization（對每個訓練文件進行 tokenization）：
    - 將文件內容轉為小寫字母。
    - 刪除stop words。（停用詞從 stopwords.txt 文件中讀取）
    - 清理標點符號，保留字母和數字字符。
    - 使用 Porter Stemming 進行詞根還原。
3. 詞彙字典建立：
    - 將處理過的所有詞彙存儲在一個set中。
4. 計算 TF和 DF：
    - TF：對每個tokenized_doc中的token計算其在該文doc的出現次數，並將結果存儲在 tf_matrix 中。
    - DF：計算dictionary中每個term在多少個doc中出現過，並將結果存儲在 df_all 中。
5. 計算 IDF：
    - 根據 df_all 和總doc數量計算每個term的 IDF 值。
    - 公式：IDF(t)=log (N / df(t))
6. 計算 TF-IDF：
    - 使用計算出的 TF 和 IDF 值，對每個文檔中的每個詞計算其 TF-IDF 值，並將結果存儲在 tf_idf_matrix 中。
7. 計算Cosine Similarity：
    - 使用Cosine Similarity計算兩個doc之間的相似度。
    - 公式：Cosine Similarity(A,B)=A·B / ‖A‖‖B‖
8. 初始化相似度矩陣：
    - 每對文檔 i 和 j，計算相似度，並儲存在矩陣 C[i][j] 中。
    - 用I變數紀錄doc是否被合併。
    - A list紀錄合併組合。
9. 進行cluster：
    - 選擇相似度最大的兩個cluster進行合併

    - 合併後要更新similarity值。使用complete-link clustering方法，找出兩cluster距離最遠的兩個doc(similarity最低)作為similarity值。
10. 輸出結果：
    - 每當文檔數量達到指定的聚類數（20、13 或 8），聚類結果將會被保存到對應的文件中。每個文件包含每個簇的文檔 ID。