

1. 執行環境：Eclipse 2019-06

2. 程式語言：Java (version: 1.8.0_101)

3. 執行方式：（* 有提供助教整個 pa1 java project）

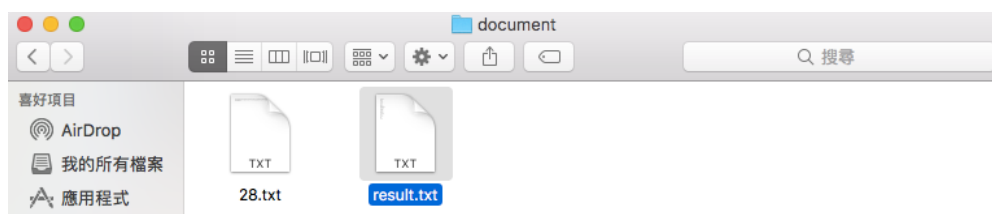
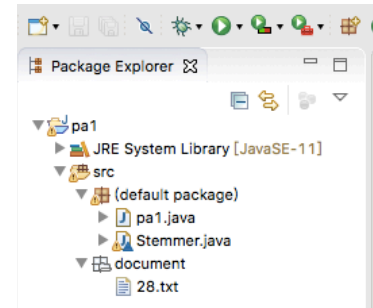
(1) 新增名為 pa1 的 java project，在 src 下建立新的 package 及 pa1 class (pa1.java)

(2) 在 src 之下新增名為 “document” 的資料夾，並在此資料夾之下新增一個內文為 ”And Yugoslav ... The World.” 的 txt 檔，名為 28.txt，供後續讀取檔案用

(3) 在 default package 新增一個名為 “Stemmer” 的 class，貼入 Porter’s Stemmer 的 code（來源：<https://tartarus.org/martin/PorterStemmer/java.txt>）

(4) 為了讀取及寫入 txt 檔案，有 import [java.io](#) 的部分套件，第4點作業處理邏輯有詳細說明

(5) 執行 pa1.java 後會在 document 的資料夾中建立 result.txt 檔，並將 output 結果寫入（如下圖所示），console 中也會印出同樣的 output 結果



4. 作業處理邏輯說明：

(1) Read document

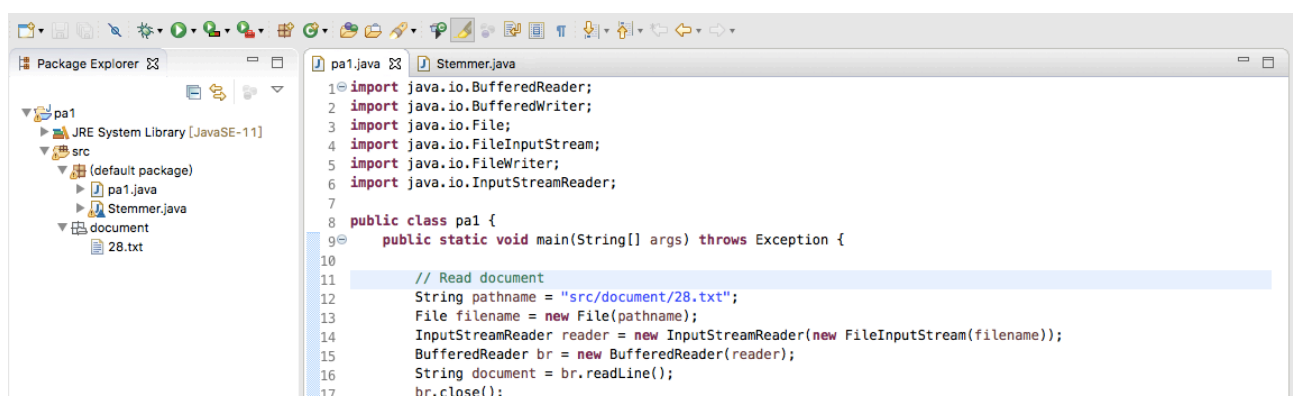
a. 我先在 src 下建了一個名為 “document” 的資料夾，再新增一個內文為 ”And Yugoslav ... The World.” 的 txt 檔，名為 28.txt，所以文件的 path 就是 “src/document/28.txt”

b. import java.io.File 來讀取路徑為 “src/document/28.txt” 的 input

c. import java.io.FileInputStream 及 java.io.InputStreamReader 建立一個輸入物件流及 reader

d. import java.io.BufferedReader 建立一個名為 “br” 的物件，讓程式能讀懂檔案內容

e. 從 br 讀入一行資料，建立名為 “document” 的 String



(2) Tokenization and lowercasing

- a. 建立一個名為“token”的字串，將 document 中非英文字母（例如：標點符號）以 .replaceAll() 的 method 替換成空值，透過 .toLowerCase() 的 method 將大寫轉換成小寫，再用 .split() 的 method 以空白（正則表示式 \s+）分割字串

```
19 // tokenization and lowercasing
20 String[] token = document.replaceAll("[^a-zA-Z]", "").toLowerCase().split("\\s+");
```

(3) Stemming using Porter's algorithm

- a. 在 default package 新增一個名為“Stemmer”的 class，貼入 Porter's Stemmer 的 code（來源：<https://tartarus.org/martin/PorterStemmer/java.txt>）
 - b. 建立一個名為“stemToken”的字串來存取待會 stem 後的結果，字串長度和 token 一致
 - c. 以迴圈的方式對每一個 token 做 stem：先建立一個 Stemmer 物件，將字串轉換為字元才能執行 .add() 的 method，放入 Stemmer buffer 之後才能以 .stem() 對 token 做 stem，最後再將 stem 後的 token 以 .toString() 取出並存入 stemToken 的字串中
- * 有在 console 中印出 stem 前後的結果來檢查是否正確執行，以及存入的字串是否正確

```
22 // Porter's Algorithm
23 String[] stemToken = new String[token.length];
24 for(int i = 0; i < token.length; i++) {
25     Stemmer s = new Stemmer();
26     char charToken[] = new char[token[i].length()];
27     charToken = token[i].toCharArray();
28     s.add(charToken, token[i].length());
29     s.stem();
30     stemToken[i] = s.toString();
31     // check output
32     // System.out.println("原本：" + token[i]);
33     // System.out.println("Stem：" + s.toString());
34     // System.out.println(stemToken[i]);
35 }
```

(4) Stopword removal

- a. 自己定義 stopwords list：and, the, to, are, of, in, on, with, that, for
- b. 先建立一個空的，名為 result 的字串，以便後續加入去除 stopwords 後的 tokens
- c. 以迴圈的方式比較 stopwords list 中是否包含各個 stemToken，若 stopwords list 中沒有 stemToken，則將該 stemToken 存入 result 字串中，以換行做區隔

* 有在 console 中印出結果

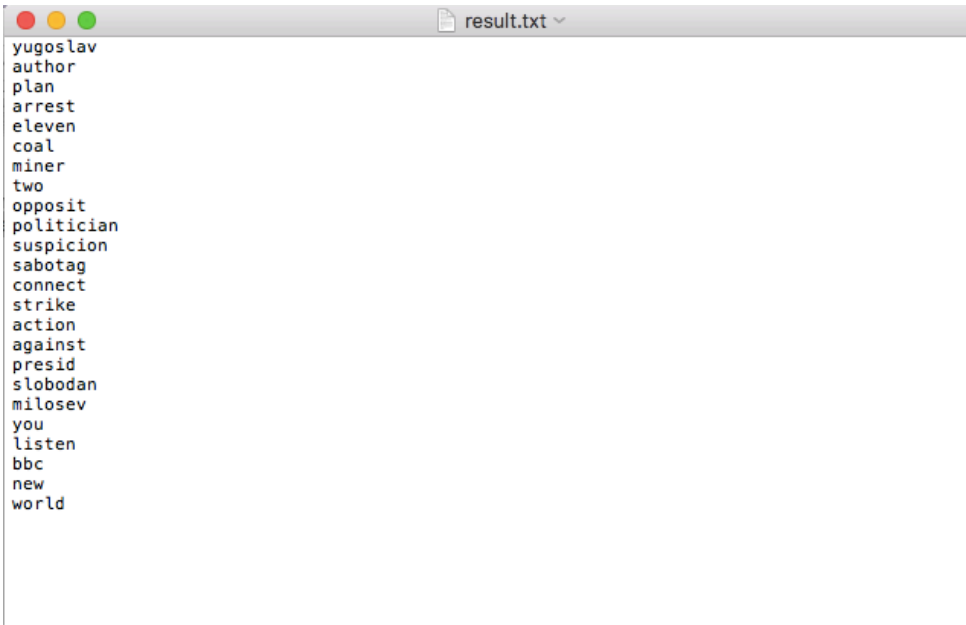
```
37 // 自定義 stop word list
38 String stopwords = "and,the,to,are,of,in,on,with,that,for";
39 String result = "";
40 for(int i = 0; i < stemToken.length; i++) {
41     if(!stopwords.contains(stemToken[i])) {
42         result = result + stemToken[i] + "\n";
43     }
44 }
45 // check output
46 System.out.println(result);
```

(5) Save the result as a txt file

- a. import java.io.File 並用 .createNewFile() 建立路徑為“src/document/result.txt”的檔案
- b. import java.io.BufferedWriter 和 java.io.FileWriter 建立一個名為“bw”的物件，以 .write() 將 result 結果寫入，再以 .flush() 將緩衝區的內容壓入檔案，最後將檔案關閉

```
48 // Write document
49 File writepath = new File("src/document/result.txt");
50 writepath.createNewFile();
51 BufferedWriter bw = new BufferedWriter(new FileWriter(writepath));
52 bw.write(result);
53 bw.flush();
54 bw.close();
55 }
56 }
```

Output 結果：



The screenshot shows a text editor window with the title bar 'result.txt'. The window contains a list of words, one per line, which are the results of a search or extraction process. The words are: yugoslav, author, plan, arrest, eleven, coal, miner, two, opposit, politician, suspicion, sabotag, connect, strike, action, against, presid, slobodan, milosev, you, listen, bbc, new, and world.

yugoslav
author
plan
arrest
eleven
coal
miner
two
opposit
politician
suspicion
sabotag
connect
strike
action
against
presid
slobodan
milosev
you
listen
bbc
new
world