

# [2014 JAVA 物件導向程式設計 Homework 8]

## 注意事項

請使用 JAVA 語言，配合 Eclipse 寫本次作業並進行測試，並安裝、使用 JAVA SE Development Kit(JDK) 7 函式庫。

請依據作業規定設定 Eclipse 專案名稱與 package name，若未依照規定將根據狀況扣分。

嚴禁抄襲其他同學作業，參與者(抄襲與被抄襲)均以零分計算。

請對你的程式碼有深入瞭解，demo 時助教會問。

對題目有問題可以寄信問助教群([java\\_ta@net.nsysu.edu.tw](mailto:java_ta@net.nsysu.edu.tw))或是到實驗室 (EC5018)詢問，但不幫忙 debug。

逾期以零分計算，不接受補交，有任何因素導致無法如期繳交，請事先告知；Demo 時間會另外通知。

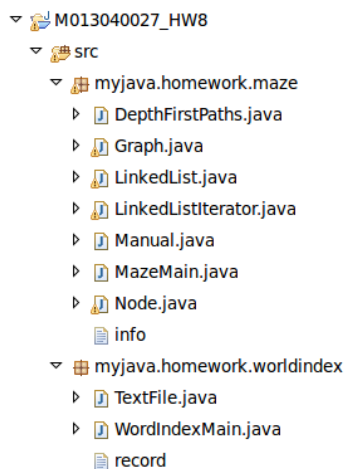
## 作業規定與上傳

Eclipse 專案名稱：<學號>\_HW8

作業請繳交專案之 tar 或 zip archive 並上傳至網路大學，Demo 時若為 rar 或其他形式之壓縮檔先扣十分。

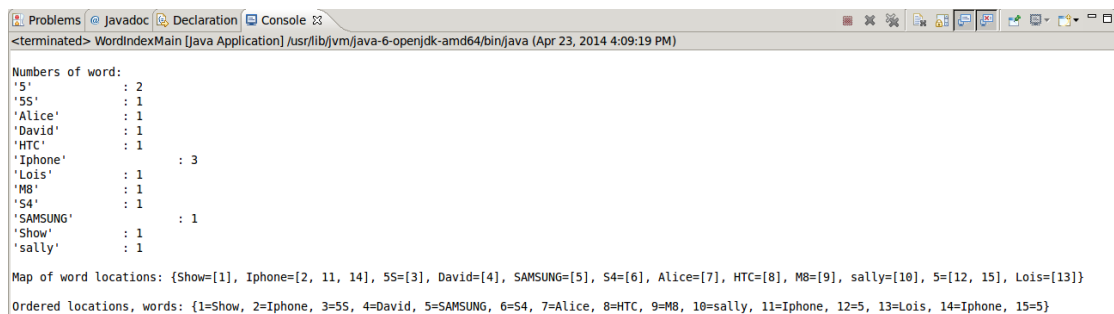
請於 2014 年 5 月 5 日(週一) 23:59 前上傳完畢，逾期以零分計算，不接受補交，有任何因素導致無法如期繳交，有問題請事先告知，再次強調，Demo 時間會另外通知。

Example of eclipse package explorer:



## 1. 字串位置搜尋與統計 (Package path : myjava.homework.wordindex)

- (1) 本題作業目的在於開發一個字串位置搜尋與統計的應用程式，同學必須產生 **Map<String , ArrayList<Integer>>**，並使用附件中所提供的 TextFile 開啟文字檔(TextFile 建構式中請用 "\\W+" 做為第二個引數)，讀取並計算單字數量。在 ArrayList<Integer> 中記錄某單字出現時已經讀取的單字數量，換句話說就是紀錄該單字在檔案中出現的位置(e.g. This is example. => example 位置是 3)。
- (2) 除了字串出現位置之外，同學尚需統計出每個單字所出現的次數，並**利用(1)所得之 Map**，依照原檔案中出現的順序，重新排列這群單字。
- (3) 輸出結果如下圖：



```
<terminated> WordIndexMain [Java Application] /usr/lib/jvm/java-6-openjdk-amd64/bin/java (Apr 23, 2014 4:09:19 PM)

Numbers of word:
'5' : 2
'SS' : 1
'Alice' : 1
'David' : 1
'HTC' : 1
'Iphone' : 3
'Lois' : 1
'M8' : 1
'S4' : 1
'SAMSUNG' : 1
'Show' : 1
'sally' : 1

Map of word locations: {Show=[1], Iphone=[2, 11, 14], SS=[3], David=[4], SAMSUNG=[5], S4=[6], Alice=[7], HTC=[8], M8=[9], sally=[10], 5=[12, 15], Lois=[13]}

Ordered locations, words: {1=Show, 2=Iphone, 3=SS, 4=David, 5=SAMSUNG, 6=S4, 7=Alice, 8=HTC, 9=M8, 10=sally, 11=Iphone, 12=5, 13=Lois, 14=Iphone, 15=5}
```

輸出說明：

Numbers of word:

'HTC' : 1 => 表示 HTC 在文件中出現過一次

Map of word locations:

Iphone[2, 11, 14] => 表示 Iphone 在文件中位置 2, 11, 14 之處出現

Ordered locations, word:

1=show => 表示文件中第一個單字是 show

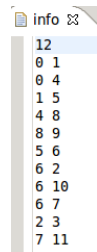
5=SAMSUNG => 表示文件中第五個單字是 SAMSUNG

註 1. 單字數量統計(Numbers of word)結果請依照 0-9，a(A)-z(Z)排序。

e.g. 1as, 75d, 9df, a1l, Bill, zero

## 2. 路徑搜尋 (Package path : myjava.homework.maze)

請實作一個迷宮遊戲，其遊戲模式分成自動搜尋路徑與手動遊戲兩種方式，其中本次作業自動遊戲模式請採用 DFS(Depth First Search)演算法搜尋路徑；手動遊戲模式則必須顯示鄰近節點讓使用者選擇下一步要前進的目標。



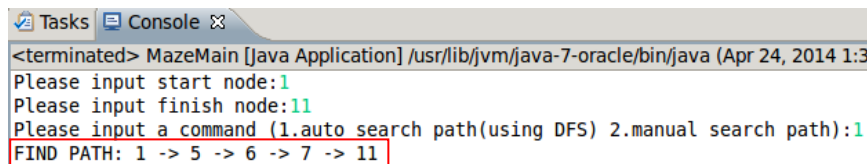
12
0 1
0 4
1 5
4 8
8 9
5 6
6 2
6 10
6 7
2 3
7 11

說明:

- (1) 請從 info 檔案中讀取節點數量與節點之間的連線關係，如右圖所示，info 檔案中第一列數字(12)表示總共有 12 個節點，第二列以後的數字則表示各個節點之間的連線關係(e.g.第二行的 0 1 表示節點 0 與 1 之間有連線)。
- (2) 請用 **Linked List** 實作你的作業，並清楚定義節點之結構
- (3) 輸出結果如下:

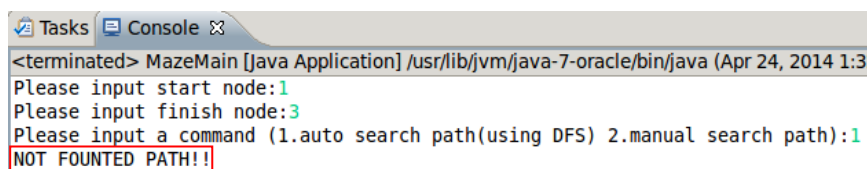
### i. DFS 搜尋:

Find path:



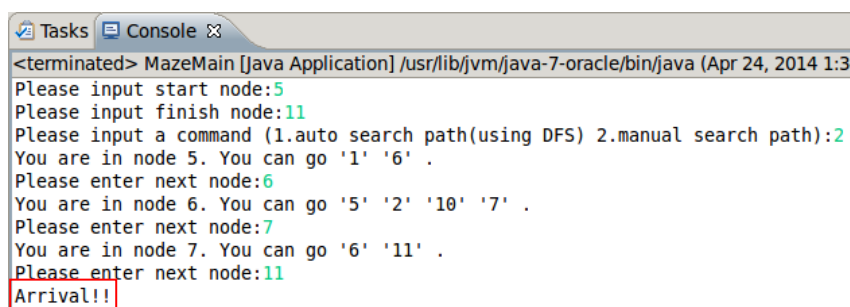
```
<terminated> MazeMain [Java Application] /usr/lib/jvm/java-7-oracle/bin/java (Apr 24, 2014 1:3
Please input start node:1
Please input finish node:11
Please input a command (1.auto search path(using DFS) 2.manual search path):1
FIND PATH: 1 -> 5 -> 6 -> 7 -> 11
```

Do not find path:



```
<terminated> MazeMain [Java Application] /usr/lib/jvm/java-7-oracle/bin/java (Apr 24, 2014 1:3
Please input start node:1
Please input finish node:3
Please input a command (1.auto search path(using DFS) 2.manual search path):1
NOT FOUND PATH!!
```

### ii. 手動遊戲:



```
<terminated> MazeMain [Java Application] /usr/lib/jvm/java-7-oracle/bin/java (Apr 24, 2014 1:3
Please input start node:5
Please input finish node:11
Please input a command (1.auto search path(using DFS) 2.manual search path):2
You are in node 5. You can go '1' '6' .
Please enter next node:6
You are in node 6. You can go '5' '2' '10' '7' .
Please enter next node:7
You are in node 7. You can go '6' '11' .
Please enter next node:11
Arrival!!
```

註 2. 附件中的 record 與 info 檔案僅供同學測試之用，Demo 時內容將會有所更動。

註 3. 請**依照規定**撰寫作業，如與敘述不符將以 **0 分** 計算。

預祝各位作業如期完成 ~ ^\_^