

跨校推動敏捷方法 **Grade System Project** based from 陳振炎 2014.09.10

學生以 Pair Programming 兩人一組開發方式 使用陳教授補充 Extreme Programming (XP) 而得的 myAgile 方法 來開發 Java 的 Grade System Project.

學生必須:

- 1) 使用 Eclipse 開發含 header 及 pseudo-code 的易讀的 Java source code.
- 2) 使用 JUnit 開發 Java unit test code.
- 3) 使用 eUML2 自動獲得 UML class diagram.

因此 請下載工具 (放在 "台灣敏捷方法苗圃" 的其他教材/Java Labs)

- 1) Eclipse (Java Lab 4),
- 2) JUnit (Java Lab 22), Java 單元測試工具,
- 3) eUML2 (Java Lab 27).

每個 method 應不超過 10 行. 如太長 則拆成下層的 private method.

每個 Public method 要有 Big-O time estimate.

最後要完成 3 文件:

- 1) **User manual & Acceptance test cases (.doc file)**, 下面已有 6 cases 請增 3 cases
- 2) **Source code (.java file)**,
- 3) **Unit test code (.java file)**.

Project 將依可讀性評分. 還有二份問卷收集你使用 MOOC 及 myAgile 的心得.

下面依序描述 scenarios, 開發用小的 input files, 驗收acceptance test cases, 設計文件, unit test code 部分例子, a real input file, 及本專案用到的敏捷方法實務

Scenarios 使用情節

1. A user interface (aUI) promptID 或 結束使用 (quit)

使用者要 quit

aUI showFinishMsg

2. aUI promptID 或 quit

使用者輸入 ID

aUI showWelcomeMsg

aUI promptCommand:

顯示四種指令 1) 顯示成績 2) 顯示排名 3) 更新配分 4) 離開選單 (exit)

使用者要 exit

aUI promptID 或 quit

使用者要 quit

aUI showFinishMsg

3. aUI promptID 或 quit

使用者輸入 ID

aUI showWelcomeMsg

aUI promptCommand

使用者要顯示成績

aGradeSystem showGrade(ID)

aUI promptCommand

使用者要 exit

aUI promptID 或 quit

使用者要 quit

aUI showFinishMsg

4. aUI promptID 或 quit

使用者輸入 ID

aUI showWelcomeMsg

aUI promptCommand

使用者要顯示排名

aGradeSystem showRank(ID)

aUI promptCommand

使用者要更新配分

aGradeSystem updateWeights()

顯示舊配分

使用者輸入新配分

aUI promptCommand

使用者要 exit

aUI promptID 或 quit

使用者要 quit

aUI showFinishMsg

Exceptional scenarios 異常使用情節

5. aUI promptID 或 quit

使用者輸入錯誤的 ID

aUI 顯示錯誤訊息

6. aUI promptID 或 quit

使用者輸入 ID

aUI showWelcomeMsg

aUI promptCommand

使用者輸入不正確指令

aUI 顯示錯誤訊息

下面是開發時 trace 用的極簡 input file. 真正的 input file "gradeInput.txt" 要備齊(見後附)
但最後才 run

Minimal Input File 極簡輸入檔

962001044 凌宗廷 87 86 98 88 87
962001051 李威廷 81 98 84 90 93

注意 data field names 如下：

ID name lab1 lab2 lab3 midTerm finalExam

Acceptance Test Cases (紅色表示使用者輸入) 驗收測試案例

1. A user interface (aUI) 顯示：輸入 ID 或 Q(結束使用)？ **Q**

aUI 顯示：結束了

2. aUI 顯示：輸入 ID 或 Q(結束使用)？ **962001051**

aUI 顯示：Welcome 李威廷

輸入指令 1) G 顯示成績 (Grade)
 2) R 顯示排名 (Rank)
 3) W 更新配分 (Weight)
 4) E 離開選單 (Exit)

使用者輸入： **E**

aUI 顯示：輸入 ID 或 Q(結束使用)？ **Q**

aUI 顯示：結束了

3. 開始至輸入指令如上

使用者輸入： **G**

aGradeSystem 顯示：

李威廷成績：lab1： 81

 lab2： 98

 lab3： 84

 mid-term： 90 註 依英文用字 mid 及 term 用 “-” 連結為一字

 final exam：93 註 依英文用字 final 及 exam 為兩字

 total grade：91

輸入指令如上

使用者輸入： **E**

aUI 顯示：輸入 ID 或 Q(結束使用)？ **Q**

aUI 顯示：結束了

4. 開始至輸入指令如上

使用者輸入：R

aGradeSystem 顯示：

李威廷排名第 22 (例:李陳張 totalGrade 分別為 86 86 83 則其 rank 為 1 1 3)

輸入指令如上

使用者輸入：W

aGradeSystem 顯示：

舊配分

lab1	10%
lab2	10%
lab3	10%
mid-term	30%
final exam	40%

輸入新配分

lab1	20
lab2	20
lab3	20
mid-term	20
final exam	20

請確認新配分

lab1	20%
lab2	20%
lab3	20%
mid-term	20%
final exam	20%

以上正確嗎? Y (Yes) 或 N (No)

使用者輸入：Y

輸入指令如上

使用者輸入：E

aUI 顯示：輸入 ID 或 Q (結束使用)? Q

aUI 顯示：結束了

5. aUI 顯示：輸入 ID 或 Q (結束使用)? 123456789

aUI 顯示：ID 錯了!

6. aUI 顯示：輸入 ID 或 Q (結束使用)? 962001051

輸入指令如上

使用者輸入：K

aUI 顯示：指令錯了!

User's Manual 使用手冊

本成績系統讀入全班成績 有 lab1, lab2, lab3, mid-term, final exam 等成績 內建各成績的配分為 0.1, 0.1, 0.1, 0.3, 0.4 依配分算出總成績建檔後 供使用者輸入指令查詢成績及排名 並可更新配分重算總成績 系統可處理兩種異常 1) 輸入錯誤 ID 2) 輸入不正確指令

系統主要畫面如下

UI 顯示：輸入 ID 或 Q (結束使用)? 使用者輸入 ID 如李威廷的 ID 962001051

UI 顯示：Welcome 李威廷 (註 嚴格說是"aUI 顯示" 但使用手冊讀者較不技術 要易讀)

- 輸入指令
- 1) G 顯示成績 (Grade)
 - 2) R 顯示排名 (Rank)
 - 3) W 更新配分 (Weight)
 - 4) E 離開選單 (Exit)

如輸入指令 G 則 UI 顯示：

李威廷成績：lab1： 81
lab2： 98
lab3： 84
mid-term： 90
final exam: 93
total grade： 91

如輸入指令 R 則 UI 顯示：

李威廷排名第 22

如輸入指令 W 則 UI 顯示：

舊配分

lab1	10%
------	-----

lab2	10%
lab3	10%
mid-term	30%
final exam	40%
輸入新配分	
lab1	20
lab2	20
lab3	20
mid-term	20
final exam	20
請確認新配分	
lab1	20%
lab2	20%
lab3	20%
mid-term	20%
final exam	20%
以上正確嗎? Y (Yes) 或 N (No)	

使用者接著不斷輸入上述指令 直到輸入 E (離開選單)

UI 再度顯示：輸入 ID 或 Q (結束使用)？ 使用者輸入 Q (結束使用) 系統就結束了

Design by CRC (粗體表示 **objects**, 底線表示 **public methods**)

a user interface (aUI) 讀取 input file 建構 **aGradeSystem**, 它是 a list of grades.

aGrade 包含 ID, name, lab1, lab2, lab3, midTerm, finalExam, 及 totalGrade.

(注意, 要開發的 classes 因而是 UI, GradeSystems, Grades 當然還有 Main class)

main() pseudo-code as below:

aUI 不斷 prompt 使用者 ID.

aUI check 使用者 ID,

要 **aGradeSystem** 做 containsID(ID) 看 ID 是否含在 aGradeSystem 內

aUI showWelcomeMsg,

aUI 不斷 prompt 使用者 command. 要 **aGradeSystem** 做下列 command

showGrade,

showRank,

updateWeights

要每個 **aGrade** 依照新的 weights 做 calculateTotalGrade

exit

直到使用者 exit

直到使用者 quit.

最後 **aUI** showFinishMsg.

(注意, quit exit 後來並未形成 methods 而是退化為 variables)

Architectural Design (class interface) 架構設計

class Main

main ()

class UI (user interface)

checkID(ID)

promptCommand()

promptID()

showFinishMsg()

showWelcomeMsg()

UI() 建構子 建構 aGradeSystem

class GradeSystems 儲存 a list of student grades.

containsID(ID) // 看 aGradeSystem 有否含此 ID

GradeSystems () //建構子

showGrade(ID)

showRank(ID)

updateWeights ()

class Grades 儲存 ID, name, lab1, lab2, lab3, midTerm, finalExam, and totalGrade

calculateTotalGrade(weights)

Grades () {} //建構子

Detailed Design (design sketch and pseudo-code) 細部設計

/** #####

本 Grade system 讓使用者(學生)取得他的總成績 total grade 及排名 rank.

Total grade 基於配分 weights 來算 而 weights 可以 update.

Rank 表示此 total grade 在全班學生的分數排序

Input file: 全班學生的分數 例如

962001044 凌宗廷 87 86 98 88 87

962001051 李威廷 81 98 84 90 93

注意 data field names 如下:

ID name lab1 lab2 lab3 midTerm finalExam

*/

public class Main extends Object {

public static void main (String args[])

```

    try { call UI() 建構 aUI } end try
    catch (NoSuchIDExceptions e1) {print msg1} //ex ID 錯了!
    catch (NoSuchCommandExceptions e2) {print msg2} //ex 指令錯了!
end class Main

```

```

public class NoSuchIDExceptions extends Exception { }
public class NoSuchCommandExceptions extends Exception { }

```

```

/** *****

```

```

class UI (user interface)

```

```

    checkID(ID)

```

```

    promptCommand()

```

```

    promptID()

```

```

    showFinishMsg()

```

```

    showWelcomeMsg()

```

```

    UI() 建構子 建構 aGradeSystem

```

```

    ***** */

```

```

public class UI

```

```

    /** -----

```

```

    UI() 建構子 throws NoSuchIDExceptions, NoSuchCommandExceptions

```

```

    ----- */

```

```

    try

```

```

    1.call GradeSystems() to 建構 aGradeSystem

```

```

    2.loop1 until Q (Quit)

```

```

        1. promptID() to get user ID 輸入 ID 或 Q (結束使用)?

```

```

        2. checkID (ID) 看 ID 是否在 aGradeSystem 內

```

```

        3. showWelcomeMsg (ID) ex. Welcome 李威廷

```

```

        4. loop2 until E (Exit)

```

```

            promptCommand() to prompt for inputCommand

```

```

        end loop2

```

```

    end loop1

```

```

    3.showFinishMsg() 結束了

```

```

end try

```

```

finally {}

```

```

    /** -----

```

```

    checkID (ID) throws NoSuchIDExceptions return Boolean

```

parameter: ID a user ID ex: 123456789

time: O(n) n is aGradeSystem 內全班人數

-----*/

1. 要 aGradeSystem 做 containsID(ID) 看 ID 是否含在 aGradeSystem 內
2. if not, throw an object of NoSuchIDExceptions
3. 回傳 true

/**-----

promptCommand () throws NoSuchCommandExceptions

-----*/

1. prompt user for inputCommand
2. if inputCommand is not G (Grade), R (Rank), W (Weights), or E (Exit),
throws an object of NoSuchCommandException
3. if inputCommand is E (Exit) then break
else: G aGradeSystem.showGrade(ID), R showRank(ID), W updateWeights() end if

promptID()

showFinishMsg()

showWelcomeMsg()

end class UI

/** *****

class GradeSystems 儲存 a list of student grades.

containsID(ID) //看 aGradeSystem 有否含此 ID

GradeSystems () //建構子

showGrade(ID)

showRank(ID)

updateWeights ()

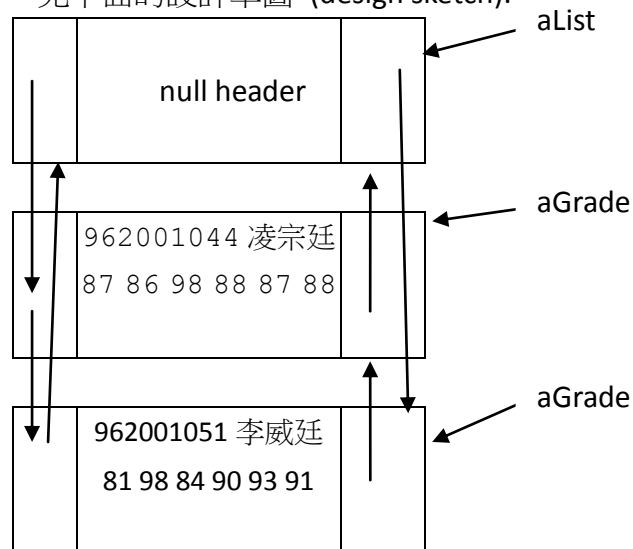
***** */

public class GradeSystems

float[5] weights; (初始值 lab1 0.1, lab2 0.1, lab3 0.1, midTerm 0.3, finalExam 0.4)
(注意 weights 之和須等於 1.0)

用 Java LinkedList class 的 object 叫 **aList** (它是 doubly-linked list) 來儲存 **aGrade** objects.

見下面的設計草圖 (design sketch):



```
public GradeSystems () //建構子
```

1. 開檔 input file
2. 用 **Java LinkedList** 建構 an empty list of grades 叫 aList
3. read line
4. while not endOfFile
 - 1.call Grades() 建構 aGrade
 - 2.用 **Java Scanner** 來 scan line 把各欄位存入 aGrade
 3. aGrade.calculateTotalGrade(weights) 回傳 aTotalGrade 把它存入 aGrade
 4. 把 aGrade 存入 aList
- end while

```
public showGrade(ID) { show 這 ID 的 grade }
```

```
public showRank(ID)
```

1. 取得這 ID 的 theTotalGrade
2. 令 rank 為 1
3. loop aGrade in aList if aTotalGrade > theTotalGrade then rank 加 1(退 1 名) end loop
4. 回傳 rank

```
public updateWeights ()
```

1. showOldWeights()
2. getNewWeights()
3. setWeights(weights)

4. loop aGrade in aList to calculateTotalGrade(weights) end loop

public containsID(ID) 看 ID 是否含在 aGradeSystem 內
end class GradeSystems

```
/** *****  
class Grades 儲存 ID, name, lab1, lab2, lab3, midTerm, finalExam, and totalGrade  
  
calculateTotalGrade(weights)  
Grades () {} //建構子  
*****/  
public class Grades  
    String name, ID; int lab1, lab2, lab3, midTerm, finalExam, totalGrade;  
    (注意 input file 內無 totalGrade )  
  
    public Grades () {} //建構子  
    public calculateTotalGrade(weights) {totalGrade 須四捨五入}  
end class Grades
```

Examples of Unit Test Code

請用不同 input parameters 的組合 形成 test cases 來測試每一個 public method

下面是 method “calculateTotalGrade (weights)” 的部分的 unit test code:

```
/** -----  
testCalculateTotalGrade  
aGrade: ID 962001051 name 李威廷 lab1 81 lab2 98 lab3 84 midTerm 90 finalExam 93  
  
case 1: lab1 0.1 lab2 0.1 lab3 0.1 midTerm 0.3 finalExam 0.4  
case 2: lab1 0.2 lab2 0.2 lab3 0.2 midTerm 0.2 finalExam 0.2  
case 3: lab1 0.1 lab2 0.1 lab3 0.0 midTerm 0.4 finalExam 0.4  
----- */  
public testCalculateTotalGrade1()  
    1. 手算 expected result (81*0.1+98*0.1+84*0.1+90*0.3+93*0.4=90.5 四捨五入 91)  
    2. 呼叫 aGrade.calculateTotalGrade (weights) 算 actual result  
    3. assert equal of the two results  
  
public testCalculateTotalGrade2()  
    1. 手算 expected result (81*0.2+98*0.2+84*0.2+90*0.2+93*0.2 四捨五入 89)
```

2. 呼叫 `aGrade.calculateTotalGrade(weights)` 算 actual result
3. `assert equal` of the two results

```
public testCalculateTotalGrade3() { ... }
```

下面是 method “`promptID()`” 的部分的 unit test code 用 `setIn` 設定 console input

1. /*使用者輸入的字串 如 “Q”*/

```
final ByteArrayInputStream inContent = new ByteArrayInputStream("Q".getBytes());
```

2. `System.setIn(inContent);`

下面是 method “`showFinishMsg()`” 的部分的 unit test code 用 `setOut` 設定 console output

```
public testShowFinishMsg()
```

1. final `ByteArrayOutputStream outContent = new ByteArrayOutputStream();`

2. `System.setOut(new printStream(outContent));`

3. 呼叫 `showFinishMsg();`

4. `assertEquals("結束了\r\n", outContent);`

A Real Input File “`gradeInput.txt`”

ID name lab1 lab2 lab3 midTerm finalExam (input file 不含這一行)

```
955002056 許文馨 88 92 88 98 91
962001044 凌宗廷 87 86 98 88 87
962001051 李威廷 81 98 84 90 93
965002038 蘇亨玠 95 89 93 81 83
965002044 商揚夏 85 86 80 81 88
966002031 陳建豪 91 95 85 83 90
975002021 楊祺賢 81 97 90 82 84
975002026 楊永成 85 90 82 93 85
975002039 許琇筑 97 84 87 99 89
975002069 練俊民 99 81 84 91 95
975002070 楊宗穎 93 93 97 96 94
975002501 張廷瑄 80 86 98 94 87
985002001 蔡立雯 93 88 93 84 94
985002002 林芯妤 99 81 91 92 95
985002003 呂映萱 83 89 94 97 80
985002004 朱燕雲 95 85 84 93 99
985002005 謝玉茹 86 90 81 91 90
```

985002006 楊佳蓉 82 88 85 89 92
985002007 張苑倫 93 82 81 93 86
985002008 容家駒 96 92 93 84 94
985002009 呂哲光 81 83 86 82 90
985002010 楊君山 96 95 91 86 96
985002011 陳柏蒼 92 88 90 99 88
985002012 史易秦 94 80 84 86 86
985002014 王祉鈞 99 89 81 83 87
985002015 許貴松 86 96 95 94 95
985002016 王淳 97 96 89 85 80
985002017 余智安 95 81 96 89 89
985002018 周信彰 94 84 86 95 90
985002020 曾則瑜 90 92 92 91 95
985002021 陳柏彰 81 81 93 96 98
985002022 高嘉靖 88 91 85 91 94
985002024 魏秀龍 85 84 92 92 80
985002025 彭康瑋 89 89 97 80 89
985002026 葉民凱 82 95 87 95 80
985002027 張軒滄 99 94 91 83 99
985002028 許哲浩 99 82 88 95 94
985002029 張瑞麟 91 80 91 97 89
985002030 朱啟文 84 93 96 82 85
985002031 陳冠宇 94 87 80 99 92
985002032 陳大慶 91 86 81 84 91
985002033 林宗瑋 90 88 97 91 94
985002034 陳忠晟 90 93 80 81 87
985002036 陳健新 99 99 96 83 87
985002037 張峻寧 89 87 94 83 89
985002038 吳德毅 99 87 86 96 89
985002039 曾裕庭 92 95 89 97 92
985002040 邢豪森 96 88 96 88 88
985002201 蘇亮 81 91 85 84 90
985002501 林佩穎 93 83 94 91 89
985002503 許佳華 94 92 83 94 97
985002504 張婉庭 97 92 96 83 93
985002505 陳以理 92 81 95 91 82
985002506 施俊浩 84 93 95 84 85
985002507 李明遠 96 98 87 89 95

985002508 蔡諄穎 87 85 94 86 88
985002509 蔡宗衛 84 92 98 94 99
985002510 黃朝偉 91 97 82 98 81
985002511 余孟昕 85 82 83 87 85
985002512 鄭達群 87 83 96 85 89
985002513 黃靖崴 85 86 94 80 85
985002514 黃安磊 90 89 90 90 86
985002515 辜麟傑 98 96 98 98 91
986002026 闕裕柏 90 85 89 87 91
995002901 舒晨馨 82 90 85 84 93

敏捷團隊一般含十位以內開發人員 外加駐點客戶 在開放式辦公室內 兩人一組快樂享受的工作 本專案用到的敏捷方法實務如下

雙人組開發 **Pair Programming**

兩人肩並肩 坐在一台螢幕前 一人主導開發用鍵盤key-in 叫driver 另一人即時檢查螢幕剛key-in的內容 叫reviewer 兩人不斷的討論內容 兩人隨時可互換角色 兩人要合力快樂用心享受的工作 這樣才能推出高品質的產品 兩個快樂臭皮匠可抵諸葛亮

持續整合 **Continuous Integration**

整合要依呼叫順序之反向 由下而上持續地整合一個個 method

例如 呼叫順序是 "UI()" "GradeSystems()" "calculateTotalGrade(weights)"

則整合順序是 "calculateTotalGrade(weights)" "GradeSystems()" "UI()"

所以要先整合Grades class的 methods 再來是GradeSystems的 然後UI的 最後是Main.

實際上 一開始只有上面的detailed design .java檔 先在其上補上

"calculateTotalGrade(weights)" 的source code 執行它的單元測試 通過後 表示此method已整合成功 接著 補下一個 method 的 source code 依此逐一整合所有 methods.

規劃活動 **Planning Game**

敏捷方法要預估 "實做及測試一個 class" 所需工作天數 如遇不確定狀況 則須啟動 攻堅突破方案 (spike solution) 使預估精準

但是 目前你是初學者 只要記錄每個 public method 各工作所耗小時數 以培養預估能力即可

程式規範 **Coding Convention**

你要依範例寫 header 及 pseudo-code 使程式易讀易維修.

程式共有 **Collective Code Ownership**

各組要遵循程式規範 加上雙人開發的輪調 久而久之 團隊成員大家都可了解修改所有的程式

勇於重整 Re-factor mercilessly (Design Improvement)

敏捷方法的各文件 要不斷狠心修改 例如陳教授不斷閱讀本文件 不斷大刀闊斧修改之 以不斷改善設計品質 請注意文件右上角的修改日期 即代表文件版本 不要閱讀到舊版文件喔

永續步伐 Sustainable Pace

絕不加班 這樣你才能每天睡飽飽 精神飽滿 永續健康 快樂有創意的上班到六十五歲 怡然退休 不至於壯年抑鬱病發 甚至早逝