Architecture Design

* **class Main**main()
* **class GradeSystem:**處理登入/登出、指令輸入、成績管理  
  GradeSystem()  
  LoginListener()  
  CommandListener()  
  CommandHandler(command)  
  LoadFile()  
  LoadSingleData()  
  Exit()  
  handleGrade()  
  handleRank()  
  handleAverage()  
  countAverage()  
  handleWeights()  
  changeWeights()  
  Logout()  
  Login()
* **class UI:**顯示需重複顯示之系統提示、成績或配分showWelcomeMsg()  
  showFinishMsg()  
  showAverage()  
  showRank()  
  showWeights()  
  showGrade()
* **class Grade:**儲存單一學生之成績與加權平均  
  Grade(ID, Name, Score, Weights)  
  getID()  
  getName()  
  getScore()  
  getWeightedScore()

Detailed Design 細部設計

/\*\* ##################################################################  
本成績系統紀錄使用者(學生)之總成績、各科成績與排名

總成績基於配分weights來計算，使用者可以自行變更配分，預設配分為{0.1, 0.1, 0.1, 0.3, 0.4}

排名表示該學生之總成績在全班學生中的分數排序。

Input file: 全班學生的學號、姓名與五科成績 例如

955002056 許文馨 88 92 88 98 91

962001044 凌宗廷 87 86 98 88 87

注意data field name如下

ID, Name, lab1, lab2, lab3, mid-term, final exam

以下將依Class為分類依序介紹該Class之Method用途

################################################################## \*\*/

/\* ###########################Class Main Start########################\*\*/

public static void main(String[] args)

1. 建立一個gradesystem

2. 讀取檔案

3. 等待使用者登入

/\*###########################Class Main End##########################\*/

/\*########################Class GradeSystem Start######################\*/

/\*\*變數宣告開始\*\*/

private Scanner scanner;

private LinkedList<Grade> gradeList = new LinkedList<>();

儲存所有學生成績的清單

public boolean locked;

紀錄當前系統有沒有使用者登入，true代表有人登入，false代表沒有

public UI ui = new UI();

private Grade currentUser;

系統當前登入的使用者

private static int[] defaultWeights = {10, 10, 10, 30, 40};

預設的配分

private int[] currentWeights;

系統當前的配分設定

private int[] averages = new int[5];

各科目的平均

private String[] Subjects = {"lab1", "lab2", "lab3", "mid-term", "final exam"};

各科目的名稱

public static final String FileName = "./src/gradeinput.txt";

學生成績檔案的路徑

/\*\*變數宣告結束\*\*/

/\*\*Methods Start\*\*/

public GradeSystem()

1. 初始化locked為false，表示目前沒有使用者登入

2. 初始化currentWeights為預設之權重

public void LoginListener()

1. 顯示登入訊息

2. 等待輸入

3. 如果輸入為"Q"，則結束程式

4. 如果輸入的長度為9，則認知其為使用者ID

5. 否則視輸入為非法輸入，顯示非法輸入提示訊息

private void CommandListener()

1. 抓取指令並呼叫CommandHandler處理

2. 如果locked為false，代表目前無使用者登入，退出迴圈並回到LoginListener

private void CommandHandler(command)

1. 用巢狀if-else判斷指令是甚麼

2. 如果沒有對應到任何關鍵字，顯示"未知的指令"和指令選單

public void LoadFile()

1. 以UTF-8編碼將檔案存取進來

2. 將檔案一行一行以String讀取進來，得到每個人的資料

3. 每行String用空格分隔取出資料並存成

4. 呼叫LoadSingleData將取出的資料傳進去

throw

FileNotFoundException - 給定的路徑找不到檔案

IOException - 無法讀取檔案

private void LoadSingleData(datas)

1. 宣告一個整數陣列來儲存分數

2. 透過確認ID長度來處理UTF-8編碼開頭BOM的問題

3. 將成績的型別由字串轉成數字

4. 新增一個Grade Class來儲存成績

5. 將新增好的Grade放到Grade List內

private void Exit()

當程式關閉時呼叫ui.showFinishMsg()顯示結束訊息

private void handleGrade()

1. 呼叫ui.showGrade()以顯示目前登入系統之使用者的成績

2. 呼叫ui.showWelcomMsg()顯示指令選單

private void handleRank()

1. 將Rank初始化為一

2. 將整個gradeList內每筆grade存取一遍，若存取到的grade比使用者高分，就將Rank加一

3. 呼叫ui.showRank以顯示Rank

4. 呼叫ui.showWelcomMsg以顯示指令選單

private void handleAverage()

1. 呼叫countAverage算出所有科目的平均

2. 呼叫ui.showAverage以顯示平均分數

3. 呼叫ui.showWelcomMsg以顯示指令選單

private void countAverage()

1. 將整個gradeList內每筆grade存取一遍，並把所有人每科的分數加到陣列average內

2. 將陣列average內五個科目的總和分數除以人數，得到所有科目的平均分數

private void handleWeights()

1. 呼叫ui.showWeights顯示當前的配分

2. 顯示"請輸入新配分"提示使用者輸入配分

3. 宣告整數sum紀錄使用者輸入配分的總和

4. 用一個執行五次的for迴圈提示使用者輸入到哪一項科目並抓取使用者輸入的數字

5. 用scanner.nextLine()過濾掉換行字元

6. 若總和不為100，提示使用者輸入錯誤並回到指令選單

7. 若總和為100則呼叫changeWeights修改配分

throw

InputMismatchException - 使用者輸入的不是數字

NoSuchElementException - 使用者未輸入字元

private void changeWeights(weights)

1. 提示訊息詢問使用者新配分是否正確

2. 若使用者輸入EOF中止修改配分則顯示"偵測到EOF 回到選單"

3. 若使用者輸入"Y"則將currentWeights改成使用者輸入的配分，

並呼叫ui.showWeights將修改後的配分顯示出來

4. 若使用者輸入"N"則顯示"取消修改 回到選單"

5. 若使用者輸入的不是"Y"也不是"N"則提示輸入不合法，回到選單

private void Logout()

1. 呼叫ui.showFinishMsg顯示登出訊息

2. 將locked設為false，表示當前系統無使用者登入

private void Login(command)

1. 存取gradeList內每個元素逐一比對每個元素的ID與使用者輸入的ID

2. 若有元素的ID與使用者輸入的ID相同，將currentUser設為該元素，顯示登入成功並呼叫ui.showWelcomeMsg顯示指令選單，將locked設為true表示系統目前有使用者登入

3. 如沒有元素的ID與使用者輸入的ID相同，顯示登入失敗

/\*\*Method end\*\*/

/\*########################Class GradeSystem End#######################\*/

/\*########################Class Grade Start############################\*/

/\*\*變數宣告開始\*\*/

private String id, name;

該學生的學號與姓名

private int[] scores;

該學生的各科成績

public String[] Subjects = {"lab1", "lab2", "lab3", "mid-term", "final exam"};

紀錄科目的名字

public int weightedScore;

該學生的加權平均

/\*\*變數宣告結束\*\*/

/\*\*Method Start\*\*/

public Grade(ID, Name, Score, Weights)

1. 將id, name, scores儲存好

2. 用一個for迴圈計算加權總分

3. 將加權後的總分除以100得到加權平均

public String getID()

1. 回傳該學生的ID

public String getName()

1. 回傳該學生的姓名

public int[] getScores()

1. 回傳該學生的各科成績

public int getWeightedScore()

1. 回傳該學生的加權平均

/\*\*Method End\*\*/

/\*########################Class Grade End############################\*/

/\*########################Class UI Start###############################\*/

UI()

1. 建立一個UI class

public void showWelcomeMsg()

1. 顯示登入後選項

public void showFinishMsg(flag, name)

1. 依照flag判斷當前系統是否有使用者登入

2. 如果有使用者登入則顯示登出訊息

3. 如果沒有使用者登入則顯示結束程式訊息

public void showAverage(averages, subjects)

1. 使用StringBuilder將"各項成績平均"與averages內每個元素串在一起

2. 將串好的builder轉成String並顯示

public void showRank(rank, name)

1. 顯示該使用者的名次

public void showWeights(weights, Subjects)

1. 使用StringBuilder將各科目名稱與weights內對應的權重串在一起

2. 將串好的builder轉成String並顯示

public void showGrade(currentUser, Subjects)

1. 使用StringBuilder將使用者的姓名與各科目的名字與成績串在一起

2. 如果該科目的成績不及格，則在成績後面加上"\*"

3. 再串上該使用者的加權平均

4. 將串好的builder轉成String並顯示