1. Khi chương trình được chạy, thì đầu tiên, sẽ hiển thị thông báo mô tả ngắn gọn về chương trình để người dùng hiểu được hệ thống này sẽ hoạt động thế nào, có chức năng gì. Ví dụ trong log trên thì thông báo hiển thị là: “This program reads exam/homework scores and reports your overall course grade”.

2. Sau đó, người dùng sẽ nhập điểm thi giữa kỳ - Midterm: Trong điểm thi giữa kỳ thì thông tin cần nhập là:

* + Weight (0-100): tức là trọng số của điểm thi giữa kỳ, và trọng số có giá trị là số nguyên dương sẽ nằm trong khoảng từ 0 tới 100. Ví dụ trong này là 20.
  + Score earned: là điểm số mà người dùng đã đạt được. Ví dụ trong này là 78.
  + Were scores shifted (1 = yes, 2=no): là điểm thi bạn có được tăng không, chọn 1 nếu có, chọn 2 nếu không. Ví dụ trong này là 2, có nghĩa là không được tăng.  Vì không được tăng, nên dòng thông tin sau sẽ hiển thị luôn total points.
  + Total points: là tổng số điểm, bao gồm điểm ban đầu và số điểm được tăng thêm, điểm tối đa của total point là 100. Ví dụ trong này điểm sẽ là 78 + 0  = 78.
  + Weighted score: Là điểm số đã tính dựa trên trọng số. Ví dụ trong này sẽ là weighted score =  (78/100) \* 20 = 15.6 . 15.6 là điểm số tính trên thang điểm 20.

3. Tiếp theo, người dùng sẽ nhập điểm thi cuối kỳ - Final: Trong điểm thi cuối kỳ thì thông tin cần nhập là:

* + Weight (0-100): tức là trọng số của điểm thi giữa kỳ, và trọng số có giá trị là số nguyên dương sẽ nằm trong khoảng từ 0 tới 100. Ví dụ trong này là 35.
  + Score earned: là điểm số mà người dùng đã đạt được. Ví dụ trong này là 95.
  + Were scores shifted (1 = yes, 2=no): là điểm thi bạn có được tăng không, chọn 1 nếu có, chọn 2 nếu không. Ví dụ trong này là 1, có nghĩa là được tăng. Vì là được tăng, nên sau dòng này sẽ có dòng nhập Shift amount.
  + Shift amount:  là số điểm mà đã được tăng. Ví dụ trong này là 10.
  + Total points: là tổng số điểm, bao gồm điểm ban đầu và số điểm được tăng thêm, điểm tối đa của total point là 100. Ví dụ trong này điểm sẽ là 95 + 10 = 105. Nhưng maximum của total points là 100 thôi, nên total sẽ chỉ được tính là 100.
  + Weighted score: Là điểm số đã tính dựa trên trọng số. Ví dụ trong này sẽ là weighted score =  (100/100) \* 35 = 35.0/35 . 35 là điểm số tính trên thang điểm 35.

4. Cuối cùng, người dùng sẽ nhập điểm bài tập về nhà- Homework: Trong điểm bài tập về  thì thông tin cần nhập là:

- Weight (0-100): tức là trọng số của điểm bài tập về nhà, và trọng số có giá trị là số nguyên dương sẽ nằm trong khoảng từ 0 tới 100. Ví dụ trong này là 45. (phải đảm bảo tổng 3 trọng số chính xác là 100).

- Number of assignments: là tổng số bài tập về nhà cần làm. Ví dụ trong này là 3 assignment.

- Theo đó số lượng đầu điểm cần nhập là 3 đầu điểm assignment. Với mỗi đầu điểm assignment thì sinh viên cần nhập cả 2 tham số là score và max.

* + Assignment 1 score and max:  18 20 trong đó 18 là điểm số thực của sinh viên đạt được, và 20 là điểm tối đa của bài assignment 1.
  + Assignment 2 score and max:  29 32 trong đó 29 là điểm số thực của sinh viên đạt được, và 32 là điểm tối đa của bài assignment 2.
  + Assignment 3 score and max:  31 40 trong đó 31 là điểm số thực của sinh viên đạt được, và 40 là điểm tối đa của bài assignment 3.

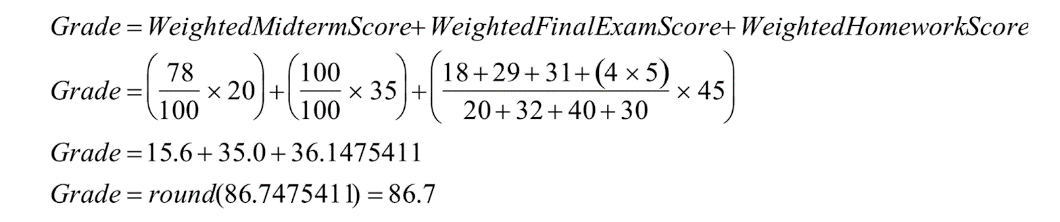
- How many sections did you attend: 4 là số lượng buổi học sinh viên đã đi học và được điểm danh. Điểm này có thể hiểu là tính điểm chuyên cần cho sinh viên.

- Section points: Là tổng số điểm chuyên cần của sinh viên, với mỗi buổi được điểm danh, thì sinh viên sẽ được tính 5 điểm. Và tối đa điểm tính cho phần chuyên cần (attend) này là 30. Trong trường hợp này điểm chuyên cần của sinh viên là: 4 x 5 = 20. Total section points là 20/30.

- Total points: là tổng điểm số của sinh viên. Ví dụ trong trường hợp này tổng số điểm số của sinh viên đạt được là total = 18 + 29 + 31 + 20 = 98. Điểm tối đa của phần bài tập là: 20 + 32 + 40 + 30 = 122.

- Weighted score: là điểm tính theo trọng số thực tế. Ví dụ trong này weighted score = 98/122 \* 45 =  36.1/45

Bạn có thể tham khảo cách tính tổng quát trong công thức sau:



5. Hiển thị kết quả tổng cộng theo môn

- Overall percentage: 86.7 đây là tổng điểm sinh viên đạt được của 3 môn tính theo thang điểm 100. Ví dụ trong này thì overall percentage = 20 + 35 + 45

- Your grade will be at least: 3.0 là grade tối thiểu sinh viên có thể đạt được dựa vào điểm số trung bình. Grade max là 4.0. Grade tối thiểu (min grade) dựa vào overall percentage Theo công thức sau:

- Quy tắc tính min grade: 85% and above: 3.0; 84.99% - 75%: 2.0; 74.99% - 60%: 0.7; under 60%: 0.0. Trong ví dụ này là 86.7, tức là rơi vào khoảng lớn hơn 85%, nên min GPA sẽ là 3.0.

- Sau phần hiển thị min GPA bạn có thể hiển thị thông báo (nhận xét) tuỳ ý dựa vào GPA của sinh viên.

*Các điều kiện cần đảm bảo*

* + Tổng điểm trọng số: trọng số của 3 phần điểm thi phải có tổng chính xác là 100. Nhỏ hơn hoặc lớn hơn 100 đều không được. Ví dụ trong này là 20 + 35 + 45 = 100.
  + Điểm tối đa của phần Assignment là 150, nếu vượt quá thì cũng chỉ tính là 150 điểm.
  + Điểm tối đa của phần Attend là 30, nếu vượt quá 30 thì vẫn chỉ tính là 30.
  + Phần Weighted score sẽ được làm tròn tới 1 chữ số đằng sau dấu thập phân.

**Phần 2. Tổ chức code**

Toàn bộ code sẽ đặt trong file GradeStudent.java

Chương trình sẽ có hàm main() điều khiển luồng chính của chương trình. Trong hàm main sẽ gọi các hàm con như sau:

Hàm **begin**() để hiển thị thông điệp chào mừng.

Hàm **midTerm**() để nhập và tính toán điểm thi giữa kỳ.

Hàm **finalTerm**() để nhập và tính toán điểm thi cuối kỳ.

Hàm **homework**() để nhập và tính toán điểm bài tập về nhà.

Hàm **report**() để tính toán về hiển thị kết quả GPA cũng như thông báo nhận xét tương ứng.

**Phần 3. Tài nguyên**

Không có tài nguyên ban đầu.

**Phần 4. Hướng dẫn chi tiết**

1. Chương trình sẽ có hàm main() điều khiển luồng chính của chương trình. Trong hàm main sẽ gọi các hàm con như sau:

2. Hàm begin() để hiển thị thông điệp chào mừng.

3. Hàm midTerm() để nhập và tính toán điểm thi giữa kỳ.

4. Hàm finalTerm() để nhập và tính toán điểm thi cuối kỳ.

5. Hàm homework() để nhập và tính toán điểm bài tập về nhà.

6. Hàm report() để tính toán về hiển thị kết quả GPA cũng như thông báo nhận xét tương ứng.

|  |
| --- |
| import java.util.Scanner; |
|  | import java.math.\*; |
|  | // Ari Madian |
|  | // CSC 142 |
|  | // Ravi Gandham |
|  | // 10/20/18 |
|  |  |
|  | public class Gradanator { |
|  |  |
|  | private static final Scanner scanner = new Scanner(System.in); |
|  |  |
|  | public static void main(String[] args) { |
|  | System.out.println("This program read exam/homework scores\nand reports your overall course grade.\n"); câu chào |
|  | double weightedMidtermScore = midtermOrFinal("Midterm:"); |
|  | double weightedFinalScore = midtermOrFinal("Final:"); |
|  | double weightedHomeWorkScore = homework(); |
|  | double totalGrade = weightedMidtermScore + weightedFinalScore + weightedHomeWorkScore; |
|  |  |
|  | printFinalGrades(totalGrade); |
|  | } |
|  |  |
|  | /\* |
|  | <midtermOrFinal> - Gets all relevant values, calculates weighted midterm or final score. |
|  | <param> String mof - Either "Midterm:" or "Final:" tells the method what to print for the section header. |
|  | <returns> double - The rounded, weighted, score for the given section. |
|  | \*/ |
|  | **private static double midtermOrFinal**(String mof){ |
|  | System.out.println(mof); |
|  |  |
|  | // Get Starting Values |
|  | System.out.print("Weight (0-100)? "); double Weight = Double.parseDouble(scanner.nextLine()); |
|  | System.out.print("Score Earned? "); double ScoreEarned = Double.parseDouble(scanner.nextLine()); |
|  | System.out.print("Were Scores Shifted (1=yes, 2=no)? "); int ScoresShifted = Integer.parseInt(scanner.nextLine()); |
|  |  |
|  | double weightScalar = Weight \* .01; |
|  |  |
|  | // If Scores Are Shifted, Get Shift Amount |
|  | if (ScoresShifted == 1){ |
|  | System.out.print("Shift Amount? "); int shiftAmount = Integer.parseInt(scanner.nextLine()); |
|  | if ((shiftAmount + ScoreEarned) >= 100) { // Check to see if shifted amount is > 100, if yes, = 100. |
|  | ScoreEarned = 100; |
|  | } else { |
|  | ScoreEarned += ScoresShifted; |
|  | } |
|  | } |
|  |  |
|  | // Print totals |
|  | printSectionFinal(ScoreEarned, 100.0, weightScalar); |
|  |  |
|  | System.out.println(); |
|  | return round(ScoreEarned \* weightScalar); |
|  | } |
|  |  |
|  | /\* |
|  | <homework> - Gets all relevant values, calculates weighted homework score. |
|  | <return> double - The rounded, weighted, homework section score. |
|  | \*/ |
|  | private **static double homework(){** |
|  | double earnedPoints = 0.0; |
|  | double totalPoints = 0.0; |
|  |  |
|  | // Get Starting Values |
|  | System.out.println("Homework:"); |
|  | System.out.print("Weight (0-100)? "); double Weight = Double.parseDouble(scanner.nextLine()); |
|  | System.out.print("Number Of Assignments? "); int numAssignments = Integer.parseInt(scanner.nextLine()); |
|  | double weightScalar = Weight \* .01; |
|  |  |
|  | // For number of assignments given, get assignment score info. |
|  | for (int assignmentNum = 1; assignmentNum <= numAssignments; assignmentNum++){ |
|  | // Get Input |
|  | System.out.print("Assignment " + assignmentNum + " score and max? "); String scoreAndMax= scanner.nextLine();//điểm đạt và điểm max |
|  |  |
|  | // Extract Values From String |
|  | double earned = Double.parseDouble(scoreAndMax.substring(0, 2)); |
|  | double total = Double.parseDouble(scoreAndMax.substring(scoreAndMax.length() - 2, scoreAndMax.length())); |
|  |  |
|  | // Add Points Up |
|  | earnedPoints += earned; |
|  | totalPoints += total; |
|  | } |
|  |  |
|  | // Section Points |
|  | System.out.print("How many sections did you attend? "); int sectionsAttended = Integer.parseInt(scanner.nextLine()); |
|  | if ((sectionsAttended \* 3) >= 20){ // Cap points if greater than max allowed |
|  | earnedPoints += 20; |
|  | System.out.println("Section Points = 20 / 20"); |
|  | } else { |
|  | earnedPoints += sectionsAttended \* 3; |
|  | System.out.println("Section Points = " + sectionsAttended \* 3 + " / 20"); |
|  | } |
|  | totalPoints += 20; |
|  |  |
|  | printSectionFinal(earnedPoints, totalPoints, weightScalar); |
|  |  |
|  | System.out.println(); |
|  | return round((earnedPoints / totalPoints) \* Weight); |
|  | } |
|  |  |
|  | /\* |
|  | <round> - Rounds a double to the first decimal point. |
|  | <param> double value - The value to round |
|  | <returns> double - The rounded value |
|  | \*/ LÀM TRÒN |
|  | **private static double round(double value) {** |
|  | BigDecimal bd = new BigDecimal(value); |
|  | bd = bd.setScale(1, RoundingMode.HALF\_UP); |
|  | return bd.doubleValue(); |
|  | } |
|  |  |
|  | /\* |
|  | <printSectionFinal> - Prints the total points earned out of points possible, and the weighted score for the section. |
|  | <param> double earnedPoints - The number of points earned for the given section. |
|  | <param> double totalPoints - The total number of points possible for the given section. |
|  | <param> double scalarValue - The section's weighing scalar value. |
|  | \*/ |
|  | private static void printSectionFinal(double earnedPoints, double totalPoints, double scalarValue){ |
|  | double oneHunTScalar = 100 \* scalarValue; |
|  | System.out.println("Total Points = " + round(earnedPoints) + " / " + (int)totalPoints); |
|  | System.out.println("Weighted Score = " + round((earnedPoints \* oneHunTScalar) / totalPoints) + " / " + (int)oneHunTScalar); |
|  |  |
|  | } |
|  |  |
|  | /\* |
|  | <printFinalGrades> - Prints the overall grade, rounded to the first decimal point, minimum GPA, and a custom message. |
|  | <param> double grade - The student's total, rounded, grade. |
|  | \*/ |
|  | private static void printFinalGrades(double grade){ |
|  | System.out.println("Overall Percentage = " + grade); |
|  | if (grade >= 85.0){ System.out.println("Your grade will be at least: 3.0\nGood Job!"); |
|  | } else if (grade >= 75.0){ System.out.println("Your grade will be at least: 2.0\nNot too bad..."); |
|  | } else if (grade >= 60.0){ System.out.println("Your grade will be at least: 0.7\nYou should probably study more next time."); |
|  | } else { System.out.println("Your grade will be at least: 0.0\nReally?"); } |
|  | } |
|  | } |