HW 2 - Decision Making with Weighted Sums

Start Assignment

- Due Monday by 11:59pm
- Points 25
- Submitting a file upload
- File Types py

HW - Decision Making with Weighted Sums

NOTE: All functionality must be done via the method indicated - no hard coding of any type is allowed for credit. Any crash or execution errors that are not gracefully handled, will result in a mandatory 20% reduction.

The decision making formula (weighted average formula) is very common and you may not even have realized its usage. For example, for grade calculation and a final % for a given class, they will typical weight different areas of scoring based on its importance (for example, have you ever wondered why exams are usually the highest weighted)?

So in this problem will consider a class that has the following components to compute the total score for the class.

- Assume that teacher is only assigning grades for the typical 90/80/70 scoring and does not have +/- suffix on the grades.
- Assume the project is a team of 4-5 students.
- Assume there is always at least two students in the course.

	Quizzes	HW	Team Project	Final Exam
Points Possible	120	150	55	80
Weights	15%	25%	25%	35%

You can assume any point totals over the max meant they received extra credit and count like normal points for that component of the grade i.e. they can get more than 100% for that component.

Furthermore for this class, you have students with the scores for each area respectively as given by:

- Student 1 scores (90, 145, 77, 0)
- Student 2 scores (35, 120, 49, 75)
- Student 3 scores are the same as Student2

- Student 4 scores (120, 145, 42, 75)
- Student 5 scores (0, 75, 35, 0)

These are provided in a csv (excel sheet) file in the assignment zip folder. There is also a base Python file for you to use and add your code to.

Download the zip folder here: **WeightedSumBase.zip (https://canvas.asu.edu/courses/234472/files/112833195?wrap=1) (https://canvas.asu.edu/courses/234472/files/112833195/download?download_frd=1)**

Take a minute to look through the csv and the Python file. You are not expected to edit the csv file at all, but since your code should work for any properly formatted csv file with any number of students you may edit it for testing purposes, but ensure that your code works with the provided one.

There are four functions in the Python file that are effectively blank (remember that the pass keyword allows you to make stubs for code). For each of the questions below, implement your solution in the matching function.

Question 1 - Calculate Student Grades

For each student in the course, print their final grade in the course. Remember the scoring assumptions listed above. Your output should have the format:

Student X has a grade of Y in the course.

Points: **7**

Question 2 - Grade Needed on Final

For every student that does NOT have a score for the Final Exam (as in they have a 0), calculate the **minimum score** they would need on the Final in order to get an A in the course. You will display a different message in the event of either of the two edge cases: there are no students without a score on the Final, or even a perfect score would not be enough to earn an A in the course. Your output should have the following format:

Normal case: Student X needs a score of at least Y on the final to get an A.

Max score is not enough: Student X cannot get an A in the course.

No students without a Final score: All students have a Final score.

Points: **6**

Question 3 - Find the Weakness

For every student, find where they lost the most points towards their final score. That is, of the four assessment types, which had the worst score. Importantly, this is not just where the lowest score out of the maximum is, but the weights must be taken into account. A student who lost more points in an assessment with a lower weight but less points in a higher weighted assessment may have lost more final score because of the higher weighting. In the edge case that a student has a perfect score, or equally lost points in multiple areas, unique messages will be printed. Your output should have the following format:

Normal case: Student X lost the most score in Y. (where Y is Quizzes, Homework, Team Project, or the Final Exam)

Perfect case: Student X got a perfect score in the course.

Equal loss case: Student X had multiple areas that held them back.

Points: **6**

Question 4 - Find Equal Students

For all students in the course, search for any students who have identical scores. If any one pair of students have identical scores on each type of assessment, print their names. You may stop after finding your first match, you do not need to find all possible matches. If there are no matches, print a unique message. Your output should have the following format:

Match found: Student X and Student Y have the same scores.

No matches found: No students had matching scores.

Points: **6**

Your submission should be a .py file.

HW 2 - Decision Making with Weighted Sums (Python Assignment)

Criteria	Ratings				Pts
Question 1: Calculate Student Grades	**7 pts** **Full Marks** Outputs accurate final grades for all students based on their weighted scores.	**5 pts** **Competent** Outputs grades with minor issues, such as slight miscalculations or formatting errors.	**3 pts** **Novice** Calculates grades for some students correctly but with major errors or incomplete results.	**0 pts** **No Marks** Does not calculate grades or crashes due to runtime errors.	7 pts
Question 2: Grade Needed on Final	**6 pts** **Full Marks** Correctly determines the minimum required score for students with missing final scores. Handles all edge cases appropriately.	**4 pts** **Competent** Calculates scores for most students but fails to account for some edge cases or special scenarios.	**2 pts** **Novice** Partial results with major gaps, such as miscalculations or errors in logic	**0 pts** **No Marks** Does not calculate the required scores or crashes due to runtime errors.	6 pts
Question 3: Identify Weaknesses	**6 pts** **Full Marks** Accurately identifies the assessment area where each student lost the most points and handles all special cases (perfect or tied scores).	**4 pts** **Competent** Identifies the weakest areas for most students but misses some cases or misinterprets edge cases.	**2 pts** **Novice** Partial implementation with errors in logic or incomplete results.	**0 pts** **No Marks** Does not provide meaningful outputs or crashes during execution.	6 pts
Question 4: Find Equal Students	**6 pts** **Full Marks** Accurately identifies students with identical scores and handles all cases, including no matches or multiple matches.	**4 pts** **Competent** Identifies matching students but with minor issues in implementation or edge case handling.	**2 pts** **Novice** Partially functional implementation, with significant errors or incomplete outputs.	**0 pts** **No Marks**	6 pts
					Total Points: 25