Student Database

In this problem, you will be dealing with the students. Each student has three pieces of information to identify him/her - **name**, **roll,** and **email**. Our current system contains the following courses, **ArtificialIntelligence**, **Security**, **OperationResearch, Networking** and **EmbeddedSystems**. A student needs to pick **three courses from here as his/her major** and one as **optional**. The difference between a major and an optional course is, that the prior one has 3.0 credits and the later one has 1.5 credits.

Now, we are going to talk about the evaluation system. It can be broken down into three entities, **Midterm**, **RegularAssesment** and **Final**. There can be three types of midterm systems - midterm of 30 marks, midterm of 20 marks, and midterm of 40 marks. Regular assessment mechanisms can be of two types - Assignment of 10 marks, Assignment of 5 marks, and 5 marks from attendance. The final can be of two types, a final of 60 marks and a final of 70. Students can pick any combination as per his/her choice from Midterm, RegularAssesment, and Final. But, the only constraint here is that the total marks need to be 100.

There will be a module for the grade calculation. This module should integrate the following functionalities,

- Should be able to calculate grades per course, based on the achieved number from the evaluation categories
- Should be able to calculate the total GPA., e.g. - $\frac{\sum credit \times grade}{\sum credit}$
- Should be able to find a course-based rank list
- Should be able to find the overall rank list based on the GPA
- Should be able to provide a comprehensive list regarding a particular course to state the students that have enrolled in that course

Evaluation Criteria:

This is a design-related problem. So, you will be evaluated on how many object-oriented designs (modularity, inheritance, design principles, etc.) you have applied in your solution. More specifically, we will evaluate the classes and the in-between relationships you have established in your code - their novelty, logical decision-related paradigms, uses, and connections. There can be multiple valid solutions, but some solutions are more flexible, object-oriented, and easily extendable or easily modifiable. The evaluation metric will bear these points.