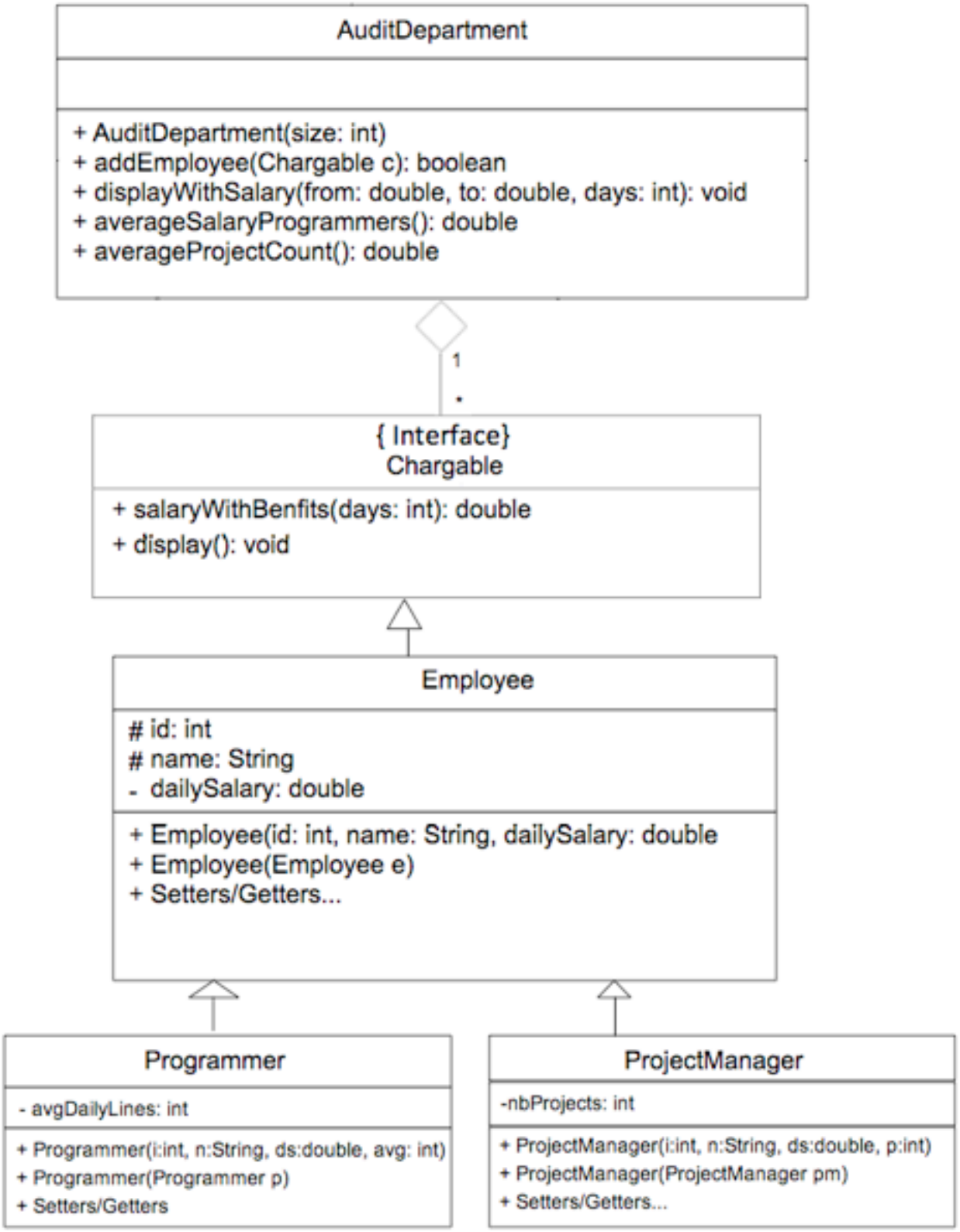
LAB 8: Interfaces and Exceptions



A company has two kinds of employees: Programmers, and Project Managers. There is a daily salary for both of them, but they differ in how their final salaries are calculated.

- Programmers are paid based on their daily salaries, as well as 10 SR for each line of their average lines of code (all multiplied by days).

  Pay = (dailySalary + 10 * avgLinesOfCode) * days

- Project Managers are only paid based on their daily salaries (multiplied by days). An additional 500 SR for each project they work on should be added to their calculated salaries.

  Pay = dailySalary * days + 500 * noProject

The company came up with the above UML diagram, and you are required to implement all of the classes. You should also:

- Implement/Override any method when required.
- Protect any class that is not intended to be instantiated (making it abstract).
- Protect any method that is not intended to be overridden (making it final).
- Protect any class that is not intended to be inherited (making it final).

The company has Audit Department, which provides the company with several information through the following methods:

- displayWithSalary(from: double, to: double, days: int): displays all the employees having salaries between from and to for the giving days.
- averageSalaryProgrammers(): returns the average salary of all programmers.
- averageProjectCount(): returns the average of project count of all project managers.

Exception handling:

- check for (ArithmeticException) when calculating the average not to divide by zero.
- check for (NegativeArraySizeException) when creating the array in the constructer.
- check for (ArrayIndexOutOfBoundsException) in addEmployee().
- check for (IllegalArgumentException) displayWithSalary() in the following:
  1. if ***from*** is larger than ***to***.
  2. if ***days*** is negative.

Finally, write a main to test your implementation of the Audit Department.