Assignment 1

Task 1: Base Class Implementation (25%)
Create a base class named Person with the following attributes:

- name (string)
- surname (string)
- age (integer)
- gender (bool)

Include the following methods: A constructor to initialize the attributes

Override a method toString that returns a message introducing the person (e.g., "Hi, I am [name] [surname], a [age]-year-old [gender (either Male or Female)].").

Task 2: Derived Classes (35%)
Create two derived classes Student and Teacher that inherit from Person.

Student Class:
Additional attributes:

- studentID (integer) - must be generated in a background
- grades (list of integers: 0-100)

Additional methods:

- addGrade: adds a grade to the list of grades
- calculateGPA: Computes and returns the average gpa from grades.

Override toString: return a message as the base class, but add, "I am a student with ID [studentID]."

Teacher Class:
Additional attributes:

- subject (string)
- yearsOfExperience (int)
- salary (int)

Additional methods:

- giveRaise: Takes a percentage and increases the teacher's salary by that percentage.

Override toString: return a message as the base class, but add, "I teach [subject]."

Task 3: Class Relationships
Create a class School with:
A list of attribute members that stores both Student and Teacher objects.
A method addMember that takes a Person object and adds it to the members list.
Override a method toString that iterates through all members and combines all members` toString methods results.

Task 4: Testing and Demonstration (20%)
Write a main program to demonstrate the functionality:
Create instances of Student and Teacher from students.txt and teachers.txt.
Add them to a School object. Test the Student and Teacher specific methods like calculateGPA and giveRaise (e.i. give raise of salary for teachers who have experience more than 10 years).

Print all members.

Hint: You may also sort members by surname.