OOP Person Task

You have been tasked with the creation of a collection of people. As you may know, a person can have different jobs in society. Your main goal is to make a base person class that you can later use to create more dynamic versions of a person.

Person 👱

Attributes:

- First name — Indicates first name of the person.
- Last Name — Indicates last name of the person.
- Age — Indicates age of the person.
- Gender — Indicates gender of the person.
- Height — Indicates the height of the person.
- Weight — Indicates the weight of the person.
- person_count — This is a class attribute that will count each person that is created. Increment in constructor.

Methods:

Setters and getters for each attribute must be created.

set first name — set a value to the object's first name attribute

set last name — set a value to the object's last name attribute

set age — set a value to the object's age attribute

set gender — set a value to the object's gender attribute

set height — set a value to the object's height attribute

set weight — set a value to the object's weight attribute

get first name — returns object's first name attribute

get last name — returns object's last name attribute

get age — returns object's age attribute

get gender — returns object's gender attribute

get height — returns object's height attribute

get weight — returns object's weight attribute

talk() - Will print out information about the certain person

Worker 👨‍💼

- This class will inherit the attributes and methods of the person class:
  - Additional Attributes:
    - Salary — Indicates the total salary of the worker
    - Weekly Hours — Indicates the total hours of the week of the worker.
  - Additional Methods:
    - set salary — set a value to the object's salary attribute
    - set weekly hours — set a value to the object's weekly hours attribute
    - get salary — returns object's salary attribute
    - get weekly hours — return object's weekly hour attribute

Student 👩‍🎓

- This class will inherit the attributes and methods of the person class:
  - Additional Attributes:
    - institution — String that represents university.
    - major — String that represents the student's major.
  - Additional Methods:
    - set institution — set a value to the object’s institution attribute.

        - set major — set a value to the object's major attribute.
        - get institution — returns object's institution attribute.
        - get major — returns object's major attribute.
        - calculate(kwargs) — Given a set of grades, calculate & return the final grade of the student. (kwargs = key word arguments)
- talk() — "Hello! I am first_name last name. I am a student at institution, studying major." (Red keywords are attributes of the class).

Engineer 👨🏽‍🔬

- This class will inherit the attributes and methods of the worker class:
    - Additional Attributes:
        - type — String that represents the type of engineer
        - company — String that represents the engineer's company
        - has_masters — Boolean to check if the engineer has a master's degree
        - has_doctorate — Boolean to check if the engineer has a doctorate's degree
    - Additional Methods:
        - set type — set a value to the object's type attribute.
        - set company — set a value to the object's company attribute.
        - get type — returns object's type attribute.
        - get company — returns object's company attribute.
- talk() — "Hello! I am first_name last name. I work at company as a type engineer." (Red keywords are attributes of the class).

Doctor 👩🏽⚕️

- This class will inherit the attributes and methods of the worker class:
    - Additional Attributes:

        - type — String that represents the type of Doctor
    - Additional Methods:
        - set type — set a value to the object's type attribute.
        - get type — returns object's type attribute.
- talk() — "Hello! I am first_name last name. I am a type doctor." (Red keywords are attributes of the class).

Lawyer 💼

- This class will inherit the attributes and methods of the worker class:
    - Additional Attributes:
        - law firm — String attribute that represents the firm associated with the given lawyer.
    - Additional Methods:
        - set law firm — set a value to the object's law firm attribute.
        - get law firm — returns object's law firm attribute.
- talk() — "Hello! I am first_name last name. I am a lawyer that works at law firm." (Red keywords are attributes of the class).

Python Implementation 🐍 (Juan)

The python side of things must include **PEP-8 style documentation** and use of python concepts such as data classes, decorators and **usage of the four pillars of OOP** (Encapsulation, Inheritance, Polymorphism and Abstraction).

A github repository must be created on your personal account with the following code structure:

src — SOURCE FOLDER (HOLDS ALL OF THE SCRIPTS).

Person.py

Worker.py

Student.py

Engineer.py

Doctor.py

Lawyer.py

main.py — Used for testing the person collection.

README.md — Describes the project at hand.

Cpp Implementation 💻 (Ana)

Cpp side of things must include the documentation used for the elevator system prototype, usage of header files and **usage of the four pillars of OOP** (Encapsulation, Inheritance, Polymorphism and Abstraction).

A github repository must be created on your personal account with the following code structure:

src — SOURCE FOLDER (HOLDS ALL OF THE SCRIPTS).

Person.h

Worker.h

Student.h

Engineer.h

Doctor.h

Lawyer.h

Person.cpp

Worker.cpp

Student.cpp

Engineer.cpp

Doctor.cpp

Lawyer.cpp

main.cpp — Used for testing the person collection.

README.md — Describes the project at hand.