**Домашнее задание к лекции «Объекты и классы. Инкапсуляция, наследование и полиморфизм»**

​Привет! Домашняя работа по данной теме является продолжением квиза к предыдущей лекции. Мы продолжим реализовывать логику функционала учебной группы в парадигме ООП. Удачи!

**Задание № 1. Наследование**

Исходя из квиза к предыдущему занятию, у нас уже есть [класс преподавателей и класс студентов](https://github.com/netology-code/py-homeworks-basic/blob/new_oop/6.classes/students_and_mentor.py) (вы можете взять этот код за основу или написать свой). Студентов пока оставим без изменения, а вот преподаватели бывают разные, поэтому теперь класс Mentor должен стать родительским классом, а от него нужно реализовать наследование классов Lecturer (лекторы) и Reviewer (эксперты, проверяющие домашние задания). Очевидно, имя, фамилия и список закрепленных курсов логично реализовать на уровне родительского класса. А чем же будут специфичны дочерние классы? Об этом в следующих заданиях.

**Задание № 2. Атрибуты и взаимодействие классов.**

В квизе к предыдущей лекции мы реализовали возможность выставлять студентам оценки за домашние задания. Теперь это могут делать только Reviewer (реализуйте такой метод)! А что могут делать лекторы? Получать оценки за лекции от студентов :) Реализуйте метод выставления оценок лекторам у класса Student (оценки по 10-балльной шкале, хранятся в атрибуте-словаре у Lecturer, в котором ключи – названия курсов, а значения – списки оценок). Лектор при этом должен быть закреплен за тем курсом, на который записан студент.

**Задание № 3. Полиморфизм и магические методы**

1. Перегрузите магический метод \_\_str\_\_ у всех классов.

У проверяющих он должен выводить информацию в следующем виде:

print(some\_reviewer)

Имя: Some

Фамилия: Buddy

У лекторов:

print(some\_lecturer)

Имя: Some

Фамилия: Buddy

Средняя оценка за лекции: 9.9

А у студентов так:

print(some\_student)

Имя: Ruoy

Фамилия: Eman

Средняя оценка за домашние задания: 9.9

Курсы в процессе изучения: Python, Git

Завершенные курсы: Введение в программирование

1. Реализуйте возможность сравнивать (через операторов сравнения) между собой лекторов по средней оценке за лекции и студентов по средней оценке за домашние задания.

**Задание № 4. Полевые испытания**

Создайте по 2 экземпляра каждого класса, вызовите все созданные методы, а также реализуйте две функции:

1. для подсчета средней оценки за домашние задания по всем студентам в рамках конкретного курса (в качестве аргументов принимаем список студентов и название курса);
2. для подсчета средней оценки за лекции всех лекторов в рамках курса (в качестве аргумента принимаем список лекторов и название курса).

**Как сдавать задачи**

Пишите код в IDE (рекомендуем [Pycharm](https://www.jetbrains.com/ru-ru/pycharm/download/" \l "section=windows), версия Community, инструкцию по установке вы найдете на сайте).

* Почему лучше работать в IDE? — Ускоряет работу, есть подсветка ошибок, отладка по шагам.
* Для более подробной информации изучите [инструкцию по работе с Pycharm](https://github.com/netology-code/guides/blob/master/python/Pycharm.md).
* Опирайтесь на принятые [правила оформления кода](https://github.com/netology-code/codestyle/tree/master/python), чтобы выработать привычку писать профессионально. При несоблюдении принятого стиля домашние задания могут быть отправлены на доработку.

1. Инициализируйте на своём компьютере пустой Git-репозиторий
2. Добавьте в этот же каталог необходимые файлы
3. Сделайте необходимые коммиты
4. Создайте публичный репозиторий на GitHub и свяжите свой локальный репозиторий с удалённым
5. Сделайте пуш (удостоверьтесь, что ваш код появился на GitHub)
6. Ссылку на ваш проект отправьте в личном кабинете на сайте [netology.ru](http://netology.ru/)
7. Любые вопросы по решению задач задавайте в чате Slack, но мы не сможем проверить или помочь, если вы пришлете:

* архивы;
* скриншоты кода;
* теоретический рассказ о возникших проблемах.