Lesson 17 - OOP (Incapsulation)

Повторение

1. Теория из прошлого урока
2. Разбор домашнего задания

Legend

Инкапсуляция, атрибуты и свойства

Ресурс: https://metanit.com/python/tutorial/7.2.php
Дополнительно:

```
Приватные атрибуты (private):
• Обозначение: Имя атрибута начинается с двух подчёркиваний (__), например, __attribute.

• Доступность: Python применяет механизм имённого манглинга (name mangling), который изменяет имя атрибута, чтобы затруднить доступ к нему извне класса. Это создаёт иллюзию строгой приватности.

• Применение: Используются для данных, которые должны быть скрыты от доступа извне (включая подклассы).
```

Задачи

```
1. Создание класса с приватными атрибутами
Создайте класс BankAccount с приватными атрибутами __account_number и __balance. Добавьте методы для:
• Установки и получения номера счета.
• Пополнения счета и снятия средств с проверкой достаточности баланса.

2. Реализация скрытых методов
Создайте класс Employee с публичным методом display_info() и скрытым методом __calculate_salary().
• Пусть display_info() вызывает __calculate_salary() для вычисления зарплаты.

3. Контроль доступа через геттеры и сеттеры
Создайте класс Student с приватным атрибутом _grade. Реализуйте:
• Геттер для получения оценки.
```

• Сеттер для установки оценки, где допустимый диапазон — от 0 до 100. Если значение выходит за пределы, выбрасывайте исключение.

4. Реализация property
Создайте класс Rectangle с приватными атрибутами __width и __height.
• Реализуйте свойства width и height через @property и @width.setter/@height.setter.
• Проверяйте, чтобы значения всегда были положительными.

5. Динамическая защита атрибутов
Создайте класс SecureData:
• Реализуйте приватный атрибут _data.
• Добавьте метод get_data() с запросом пароля для получения значения атрибута.
• Если пароль неверный, выведите на экран ошибку.

6. Моделирование реального объекта
Создайте класс ATM:
• Приватные атрибуты: __cash_balance, __pin.
• Методы:
    • check_balance(): возвращает остаток.
    • withdraw(amount): уменьшает __cash_balance, если введен правильный PIN и хватает средств.
    • change_pin(old_pin, new_pin): изменяет PIN при правильном старом PIN.

ДЗ-17

2. Реализация скрытых методов
Создайте класс Employee с публичным методом display_info() и скрытым методом __calculate_salary().
• Пусть display_info() вызывает __calculate_salary() для вычисления зарплаты.

3. Контроль доступа через геттеры и сеттеры
Создайте класс Student с приватным атрибутом _grade. Реализуйте:
• Геттер для получения оценки.
• Сеттер для установки оценки, где допустимый диапазон — от 0 до 100. Если значение выходит за пределы, выбрасывайте исключение.

4. Реализация property
Создайте класс Rectangle с приватными атрибутами __width и __height.
• Реализуйте свойства width и height через @property и @width.setter/@height.setter.
• Проверяйте, чтобы значения всегда были положительными.

5. Динамическая защита атрибутов
Создайте класс SecureData:
• Реализуйте приватный атрибут _data.
• Добавьте метод get_data() с запросом пароля для получения значения атрибута.
• Если пароль неверный, выведите на экран ошибку.

6. Моделирование реального объекта
Создайте класс ATM:
• Приватные атрибуты: __cash_balance, __pin.
• Методы:
    • check_balance(): возвращает остаток.
    • withdraw(amount): уменьшает __cash_balance, если введен правильный PIN и хватает средств.
    • change_pin(old_pin, new_pin): изменяет PIN при правильном старом PIN.