Разбор домашнего задания

Роман Булгаков
Спикер курса

Skillbox

Классы

Роман Булгаков
Спикер курса

Skillbox

Задача «Учётная запись»

Входные данные:

- имя — Admin
- пароль — qwerty
- забанен — False

Выходные данные:

- структура для хранения данных

```
user_tuple = ('Admin', 'qwerty', False)

user_list = ['Admin', 'qwerty', False]

user_dict = {
    'user_name': 'Admin',
    'password': 'qwerty',
    'is_banned': False
}
```

Реализация

```python
user_tuple = ('Admin', 'qwerty', False)

user_list = ['Admin', 'qwerty', False]

user_dict = {
    'user_name': 'Admin',
    'password': 'qwerty',
    'is_banned': False
}
```

- Непонятен порядок
- Неизменяемый тип данных
- Непонятен порядок
- Можно случайно изменить значения
- Можно случайно изменить значения

Общая проблема: при большом количестве пользователей получается очень много повторяющегося и ужасного кода.

Парадигмы программирования

```
user_name = 'Admin'
password = 'qwerty'
is_banned = False

# код с обработкой переменных
# с помощью условий, циклов...
```

Структурное программирование

```
class User:
    user_name = 'Admin'
    password = 'qwerty'
    is_banned = False

# работа с объектами класса
# с помощью его методов (ну и циклов, условий...)
```

Объектно-ориентированное программирование

Методы класса, аргумент self

Роман Булгаков
Спикер курса

Skillbox

Задача «Семья»

Условие задачи:

- реализовать класс «Семья»

Выходные данные:

- экземпляр класса

Атрибуты класса:

- фамилия
- деньги
- наличие дома

Методы класса:

- информация о семье
- заработок денег
- покупка дома (может быть скидка)

Итоги урока

- ```
  class Person:
      name = 'Tom'
      def info(self):
      ...
  ```

- ```
  def info(self):
      print(self.name)
  ```

- ```
  def buy(self, price, discount=0):
  ```


  ```

Конструктор __init__ и работа с несколькими классами

Роман Булгаков
Спикер курса

Skillbox

Задача «Весёлая ферма»

Условие задачи:

- два класса: «Картошка» и «Грядка»

Выходные данные:

- пример работы грядки

Атрибуты класса «Картошка»:

- номер
- стадия зрелости

Атрибуты класса «Грядка»:

- список картошки

Методы класса «Картошка»:

- информация о зрелости
- рост

Методы класса «Грядка»:

- информация о зрелости всей картошки на грядке
- рост всей грядки

Итоги урока

- def __init__
- def __init__(self, name, salary):
- def __init__(self, name, salary):
  self.name = name
  self.salary = salary
- emp = Employee('Tom', 10000)
- def __init__(self, name, salary=0):
  self.name = name
  self.salary = salary
  self.age = 0

Определение классов в модулях и их подключение

Роман Булгаков
Спикер курса

Skillbox

Разработка классов

```python
class Cow:

    def __init__(self, state):
        self.state = state
        # и ещё какие-то атрибуты

    def grow(self):
        # реализация роста коровы
        self.state += 1
        # ...

    # тут другие методы
```

Реализация класса Корова от Васи

```python
class Lawn:

    def __init__(self):
        self.gross_state = 10
        # атрибуты

    def eat(self, cow):
        self.gross_state -= 1
        cow.grow()

    # разработка методов для лужайки
```

Петя использует реализацию Васи для Лужайки

Парадигмы программирования

Структурное программирование — на первом месте логика, понимание последовательности действий и работы каждого объекта.

Объектно-ориентированное программирование — программа представляется в виде системы взаимодействующих друг с другом объектов, где нет упора на логику работы каждого объекта.

Итоги модуля

- `class User:`
- ```
  class User:
      name = 'Noname'
      def info(self):
      ...
  ```
- ```
  def info(self):
      print(self.name)
  ```
- ```
  def __init__(self, name):
      self.name = name
  ```
- ```
  def __init__(self, count):
      self.potatoes = [Potato(index) for index in range(1, count + 1)]
  ```
- `from garden import PotatoGarden`