Парадигма ООП

Основы Python

План урока

- Парадигмы программирования
- Классы

Декларативное программирование

это парадигма программирования, в которой описывается желаемый результат, а не конкретные шаги для его достижения

Императивное программирование

это парадигма программирования, в которой описываются конкретные шаги и команды для выполнения задачи

Декларативное программирование

vs

Императивное программирование

Структурное программирование

Эта парадигма вводит новые понятия, объединяющие часто используемые шаблоны написания императивного кода

Новые понятия

- Блок инструкций
- Условный оператор
- Циклы (for и while)

Процедурное программирование

Процедура (или функция) — самостоятельный участок кода, который можно выполнить как одну инструкцию

ooo | код

```
def fact(n):
  val = 1
  for i in range(2, n+1):
    val *= I
  return val
print(fact(5) + fact(10))
```

Объектно-ориентированно программирование

Объектно-ориентированное программирование (ООП) упрощает создание, понимание и поддержку программных систем

Основная идея - организовать программу вокруг объектов, представляющих сущности из реального мира

Объекты имеют состояния (атрибуты) и поведение (методы), взаимодействуя через сообщения

Объектно-ориентированно программирование

Является развитием процедурной парадигмы. Возникают понятия класса и объекта. Мы можем объединять методы и переменные в отдельные объекты

Например, объект может представлять человека с такими свойствами, как имя, возраст и адрес, а также методами, такими как ходьба, разговор, дыхание и бег

Или электронное письмо со свойствами, такими как список получателей, тема и тело, и поведением, таким как добавление вложений и отправка

ООП

VS

Процедурное прогарммирование

Процедурное программирование

ooo | код

```
class Rectangle:
  def __init__(self, color, width, height):
    self.color = color
    self.width = width
    self.height = height

  def get_area(self):
    return self.width * self.height

  def get_perimeter(self):
    return 2 * (self.width + self.height)


rect = Rectangle("red", 5, 4)
print(f"Rectangle area: {rect.get_area()}")
print(f"Rectangle perimeter: {rect.get_perimeter()}")
```