ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Лекция № | / **01**
05.02.2018 г.

РЕАЛИИ НАШЕГО ДНЯ

- ООП везде. Из **10** самых популярных языков программирования **7** поддерживают ООП на уровне синтаксиса.
  http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html
- Программы на C++ работают быстро.
- **Ergo:** Занимаемся ООП на C++.

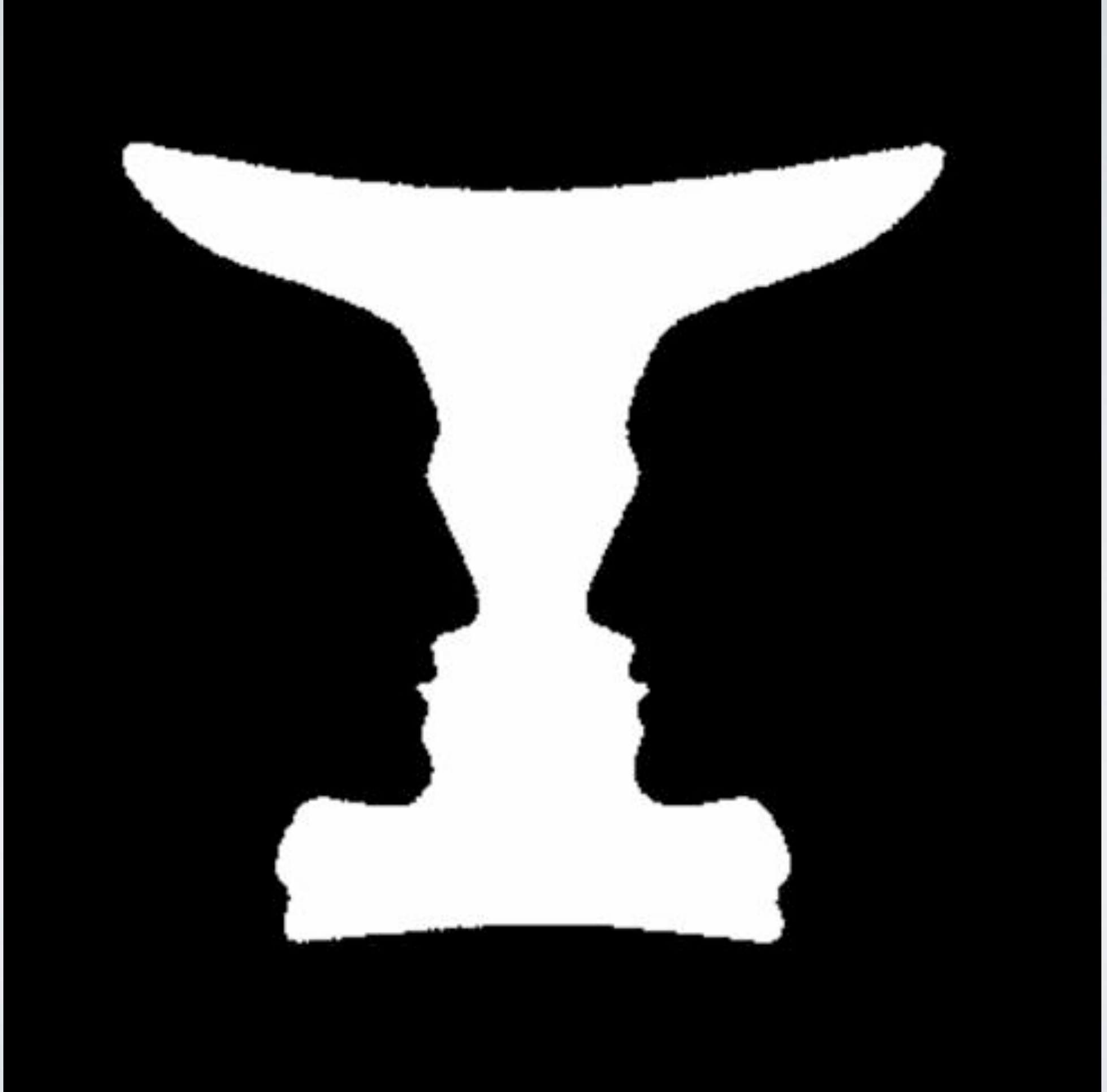
О КУРСЕ

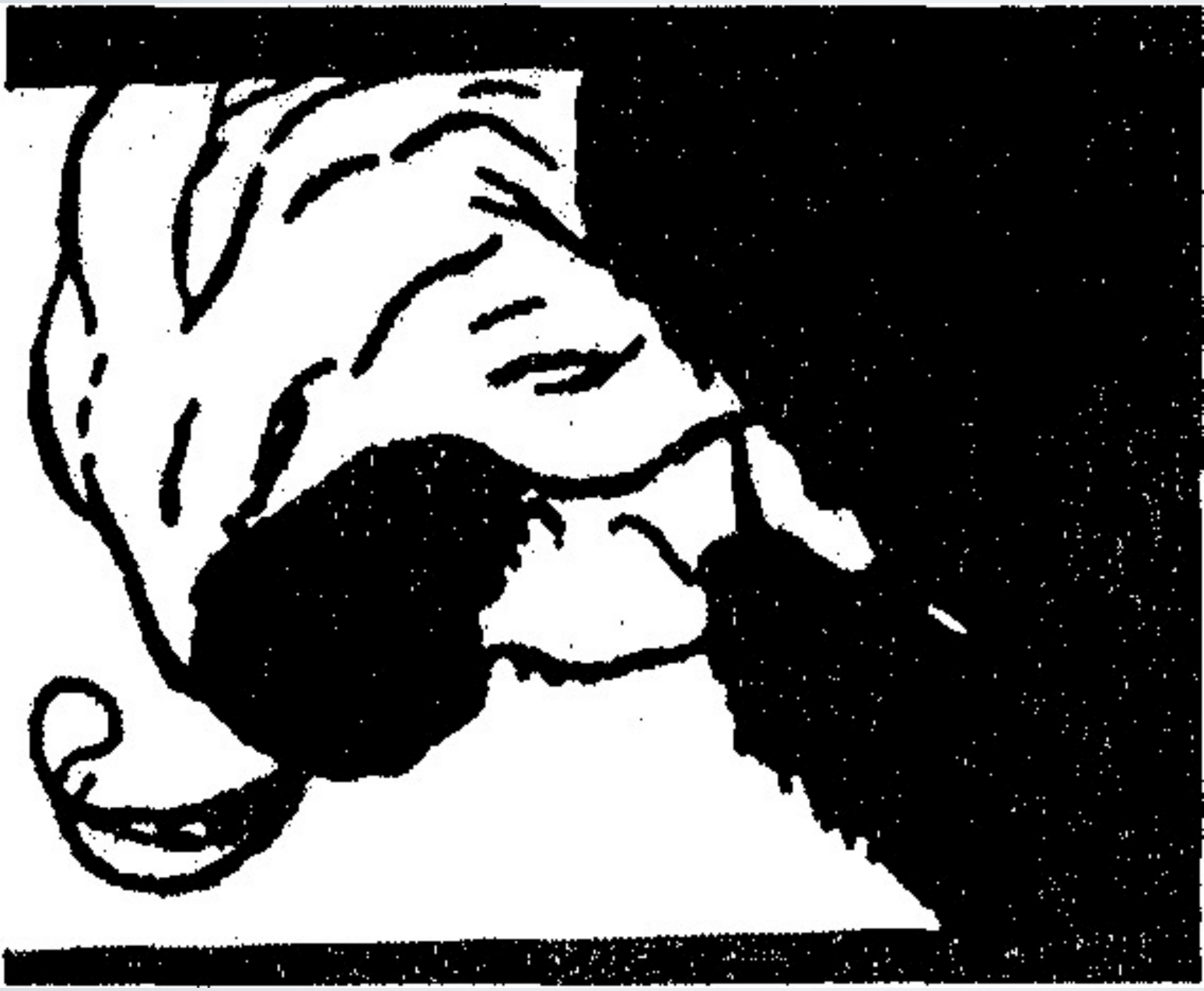
- Две основных темы:
  - C++.
  - Принципы ООП.
- Орг. часть:
  - «Как на ОПК».
  - *I часть:* проект, теория, семинары => оценка.
  - *II часть:* проект, теория, семинары, I часть => оценка.

ЛАДНО, КРУТО!

А ЧТО ТАКОЕ ООП?


"Object-oriented programming
is an exceptionally bad idea
which could only have
originated in California."
-- Edsger Dijkstra



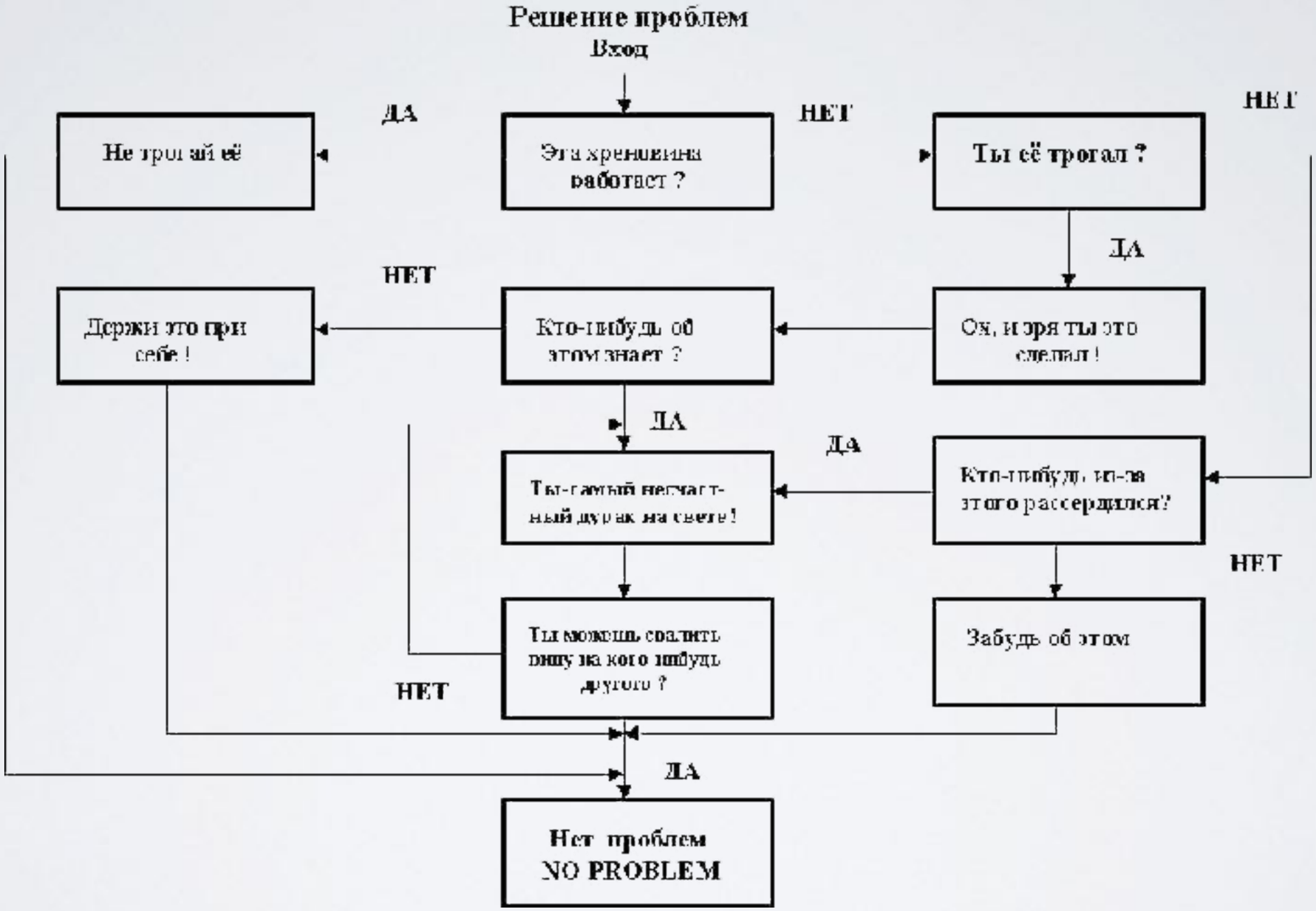


ФИГУРА И ФОН!

ПАРАДИГМЫ

- **Фигура:** заметное, яркое, важное, об этом все говорят.
- Известные нам парадигмы:
  - императивная (алгоритмическая);
  - функциональная;
  - логическая.

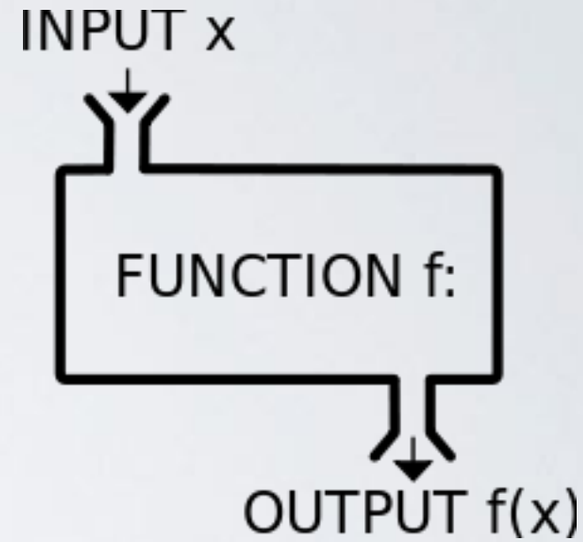
ИМПЕРАТИВНАЯ ПАРАДИГМА



Фигура: последовательность действий

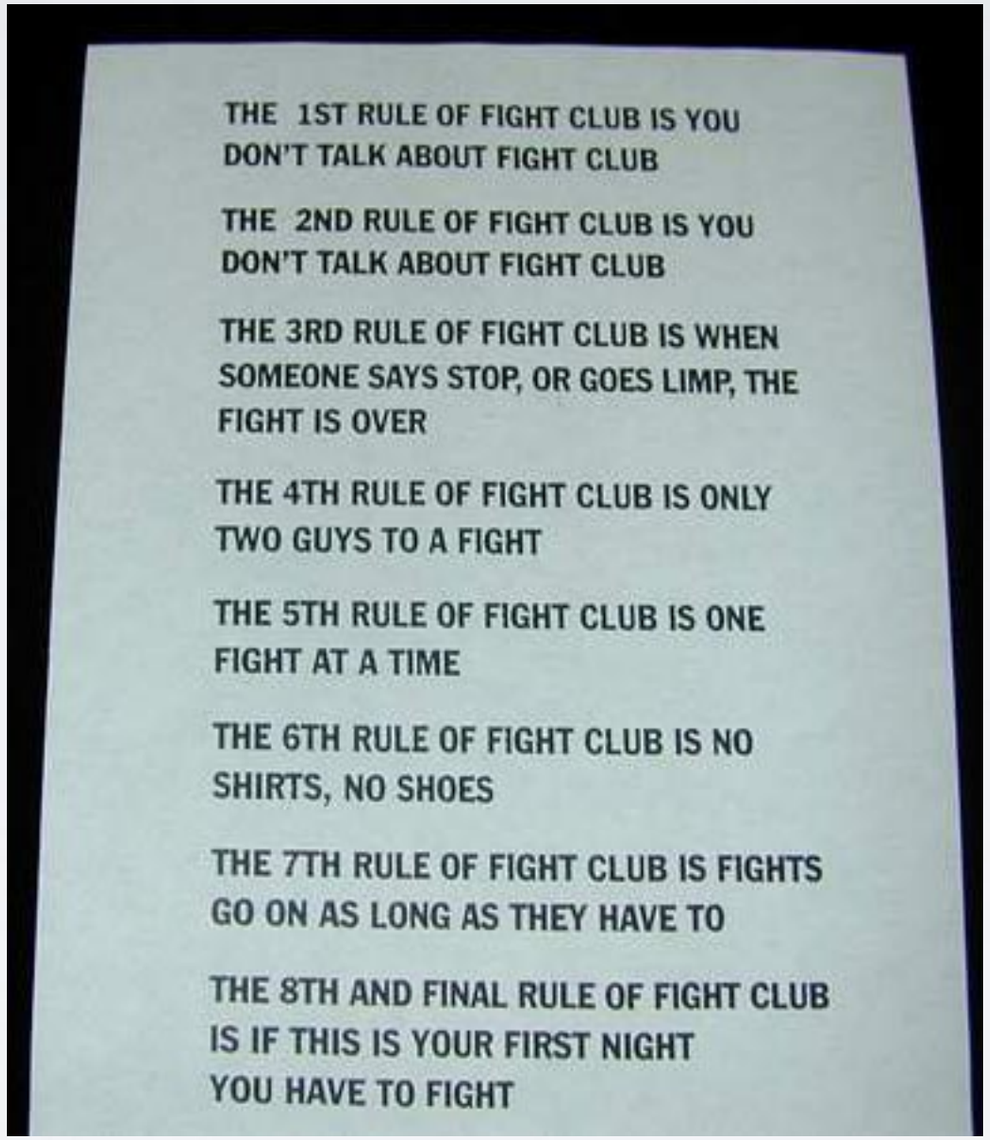
ФУНКЦИОНАЛЬНАЯ ПАРАДИГМА

- result = f1(a, b) + f2(b, c)
- Данные являются неизменяемыми.
- Последовательность действий не важна.
- Текущее состояние (≈ слепок памяти) не важно.



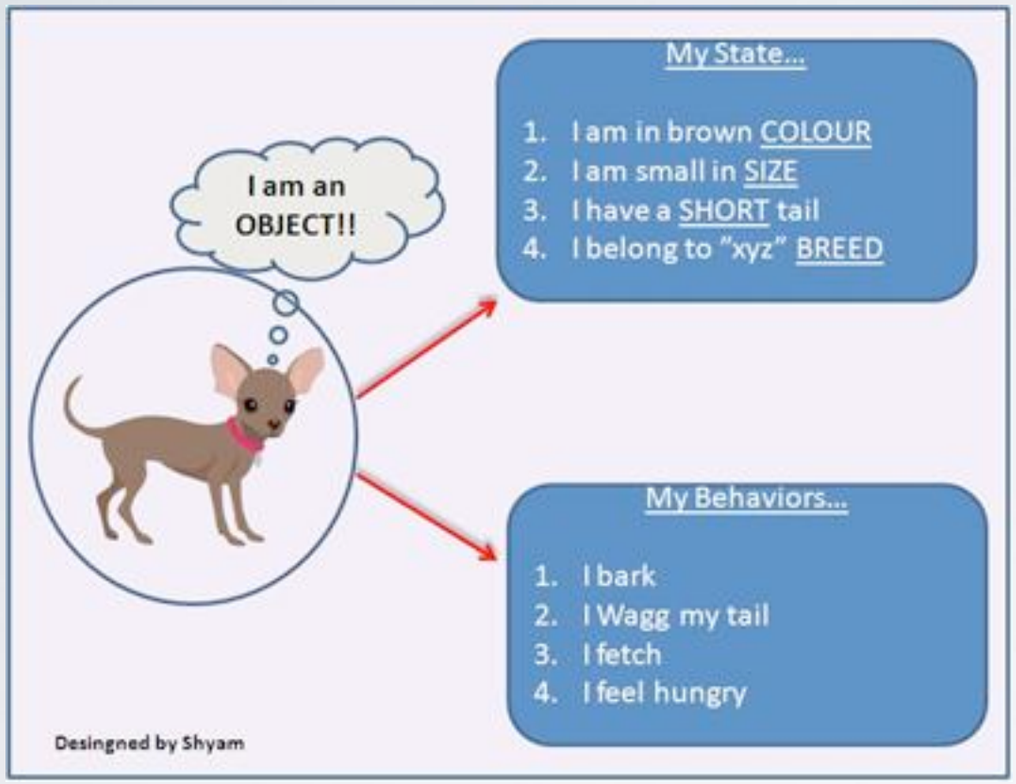
Фигура: данные и преобразования над ними

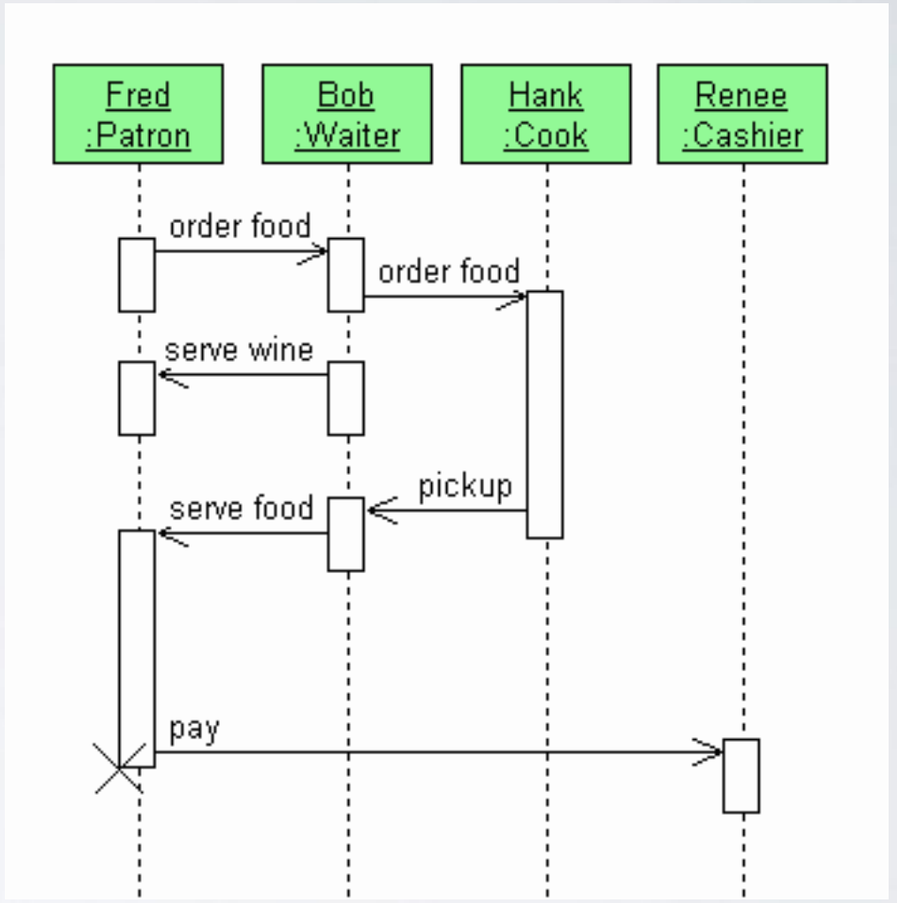
ЛОГИЧЕСКАЯ ПАРАДИГМА



Фигура: предикаты, правила

О. О. ПАРАДИГМА





Фигура: объекты, их «поведение» и взаимодействие

ОБЪЕКТЫ

- Программы так или иначе моделируют мир.
- В мире есть «объекты», т. е. некие отдельные сущности.
- Пусть в программах тоже будут объекты!
- Для нас важны **свойства** и **поведение** объекта исходя из решаемой задачи !!

ВЗАИМОДЕЙСТВИЕ ОБЪЕКТОВ

- Важен момент **рождения** объекта и время его **жизни**.

- Принцип **иерархии**: есть более глобальные объекты, которые живут дольше и порождают менее глобальные объекты.

- Принцип **черного** ящика: взаимодействие с любым объектом через интерфейс.

КАКОЙ ЯЗЫК САМЫЙ ООП-ИСТЫЙ?


КАКОЙ ЯЗЫК ЛУЧШЕ ВСЕГО
ПОДХОДИТ ДЛЯ ООП?
ООП - ЭТО КОНЦЕПЦИЯ

Объект окружающего мира

?

Объект программного мира

1. Свойства

2. Поведение
(реакция на сообщения)

МЫСЛЕННЫЙ ЭКСПЕРИМЕНТ

Много объектов с одинаковым поведением, но разными свойствами.

СХОДСТВА И РАЗЛИЧИЯ

Свойства:

- Порода
- Рост
- Вес
- Цвет
- ...

Поведение:

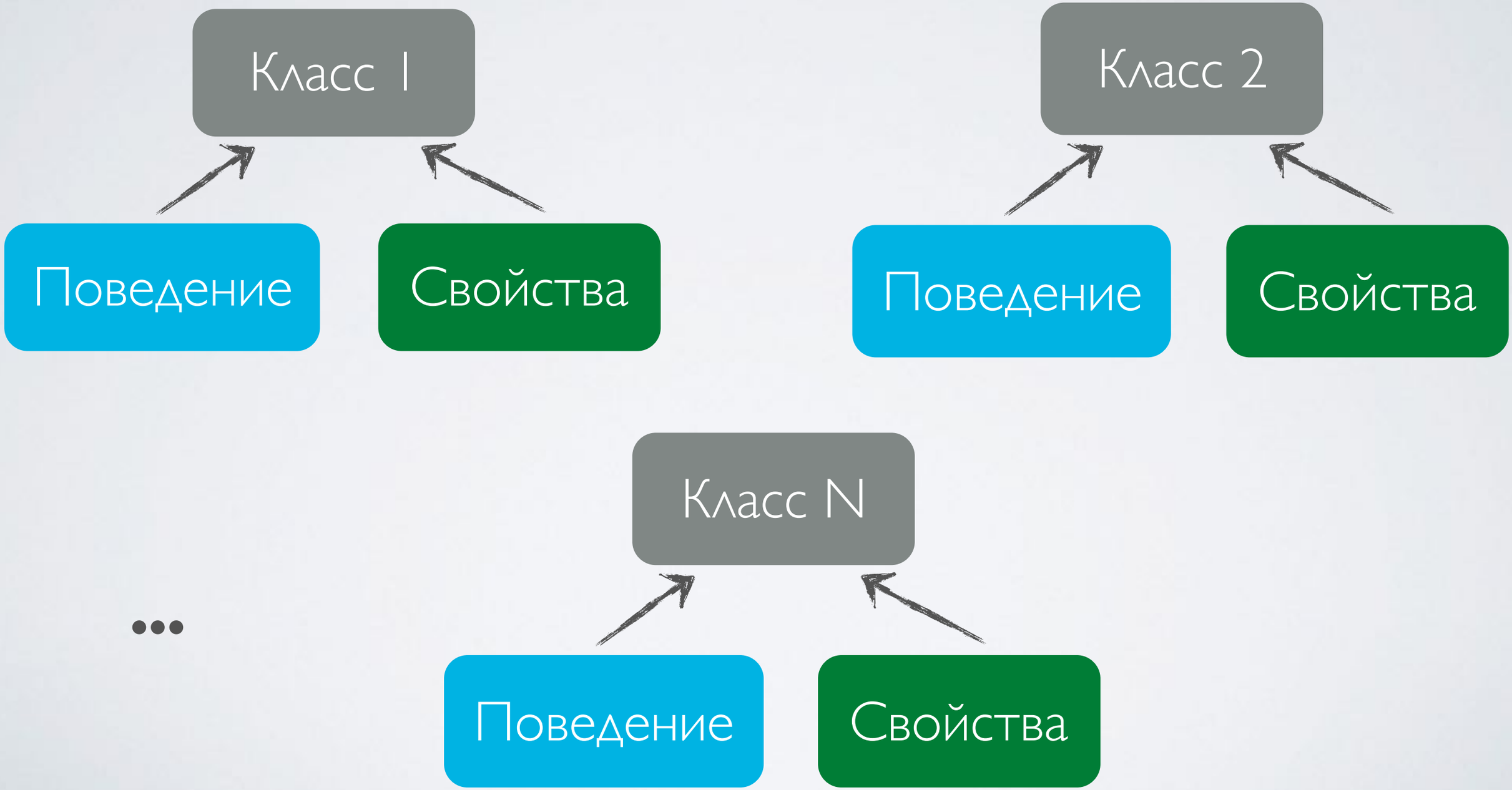
- Лаять
- Выть
- Кусать
- Приносить палочку
- ...

Объекты класса **Dog**

vs.

Объекты класса **Cat**

ОБЩАЯ СТРУКТУРА



ПРЕДСТАВЛЕНИЕ ОБЪЕКТА

- **Свойства** — данные, находятся где-то в памяти.
- **Поведение** — код, имеющий доступ к данным.
- Вопрос: как это все реализовать, например, на языке C?

ГДЕ ХРАНИТЬ ДАННЫЕ?

- typedef struct { ... } Dog;
- Виды памяти в C:
  - глобальная;
  - автоматическая (стековая);
  - динамическая (**malloc** / **free**).

ФУНКЦИИ, СВЯЗАННЫЕ С ОБЪЕКТОМ

- «Методы» объекта.
  - **void dog_bark(Dog *dog, int loudness);**
- Создание / инициализация:
  - **void dog_initialize(Dog *dog);**
  - **Dog *dog_create();**
- Уничтожение:
  - **void dog_destroy(Dog *dog);**

Конструктор

Деструктор

RATIONAL.H

```c
#ifndef __rational_h
#define __rational_h

#include <stdio.h>

struct Rational {
    int numer;  /* numerator, числитель */
    int denom; /* denominator, знаменатель */
};

void rat_add(struct Rational *result, struct Rational *a, struct Rational *b);
void rat_sub(struct Rational *result, struct Rational *a, struct Rational *b);
void rat_mul(struct Rational *result, struct Rational *a, struct Rational *b);
void rat_div(struct Rational *result, struct Rational *a, struct Rational *b);
/* возведение в целочисленную степень. power может быть отрицательным! */
void rat_power(struct Rational *result, struct Rational *r, int power);
void rat_create(struct Rational *res, int a, int b);
/* возвращает -1 (a < b), 0 (a == b), 1 (a > b) */
int rat_compare(struct Rational *a, struct Rational *b);
int rat_to_i(struct Rational *a); /* округление до ближ. целого */
double rat_to_d(struct Rational *a); /* преобразование в число с плав. точкой */
void rat_print(struct Rational *a, FILE *fp);  /*выводит в формате p/q */

#endif
```

Ничего не напоминает?

КОНЕЦ ПЕРВОЙ ЛЕКЦИИ