Модуль #2
Введение в ООП

Модуль 2

- **Парадигмы программирования**
- Классы и объекты
- Наследование
- Полиморфизм
- Инкапсуляция
- Цели ООП

Парадигмы программирования

- Парадигма программирования – это совокупность идей и понятий, определяющая стиль написания программ.

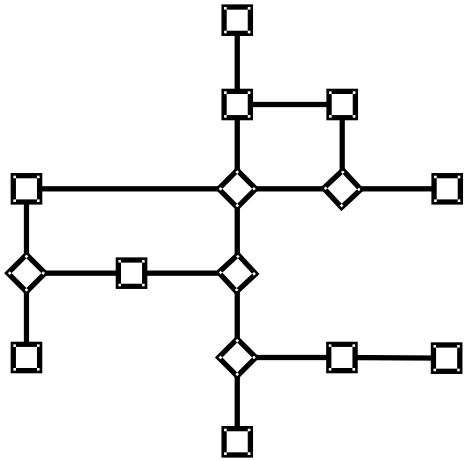
- Парадигма программирования определяет то, в каких терминах программист описывает логику программы.

- Парадигма программирования не определяется языком однозначно.

Виды парадигм программирования

- Структурное
- Логическое
- Функциональное
- Объектно-ориентированное
- Аспектно-ориенированное
- и др.

Структурное рограммирование



- Структурное программирование – это парадигма программирования, которая описывает процесс вычисления в виде инструкций, изменяющих состояние программы.

- Структурная программа очень похожа на приказы, выражаемые повелительным наклонением в естественных языках, то есть это последовательность команд, которые компьютер должен выполнить.

- Пример – язык C.

Логическое программирование

- Логическое программирование – это программирование, основанное на автоматическом доказательстве теорем.

- Логические языки программирования, обычно определяют что надо вычислить, а не как это надо делать.

- Пример - Prolog

Нахождение факториала:

```
fact(1,1).

fact(N,F) :- N>1, N1 is N-1,
fact(N1,F1), F is F1*N.

?- fact(4,X).

X=120
```

Функциональное программирование

- Функциональное программирование – это парадигма программирования, в которой процесс вычисления трактуется как вычисление значений функций.

- Любая функция – суперпозиция других функций.

- Примеры: Lisp, Haskel, Closure.

- ООП – парадигма, в которой основными концепциями являются понятия классов и объектов.

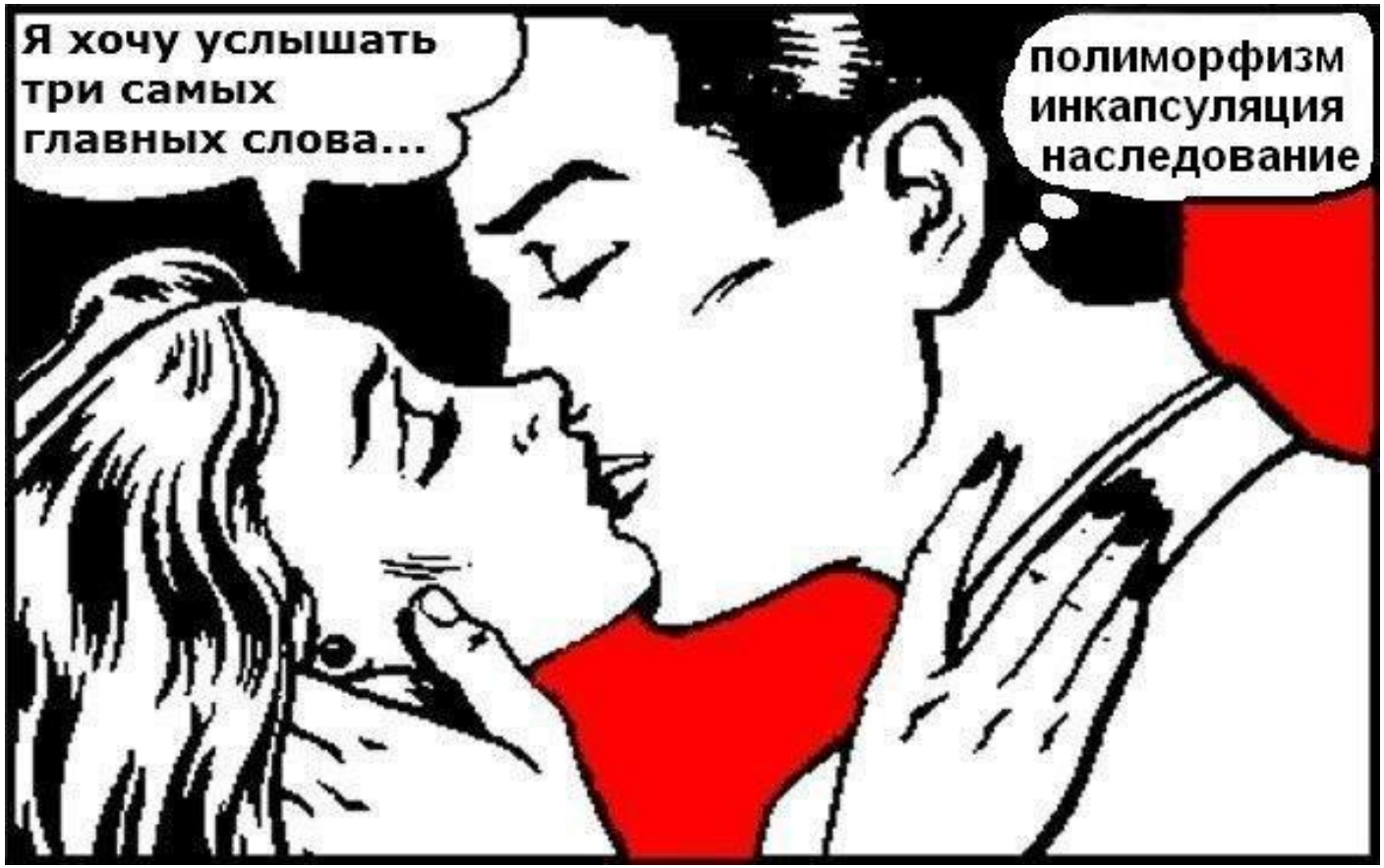
- ООП возникло на базе процедурного программирования, где было предложено данные и методы их обработки объеденить в классы.

- ООП – методология, при которой программа организуется как совокупность сотрудничащих объектов, каждый из которых – экземпляр класса.

ООП

- ООП является наиболее распространенной парадигмой программирования.
- Java является полностью ООП языком, т.е. не поддерживает программирование в процедурном стиле.

Три самых главных слова ООП



Модуль 2

- Парадигмы программирования
- **Классы и объекты**
- Наследование
- Полиморфизм
- Инкапсуляуция
- Цели ООП

Классы и объекты

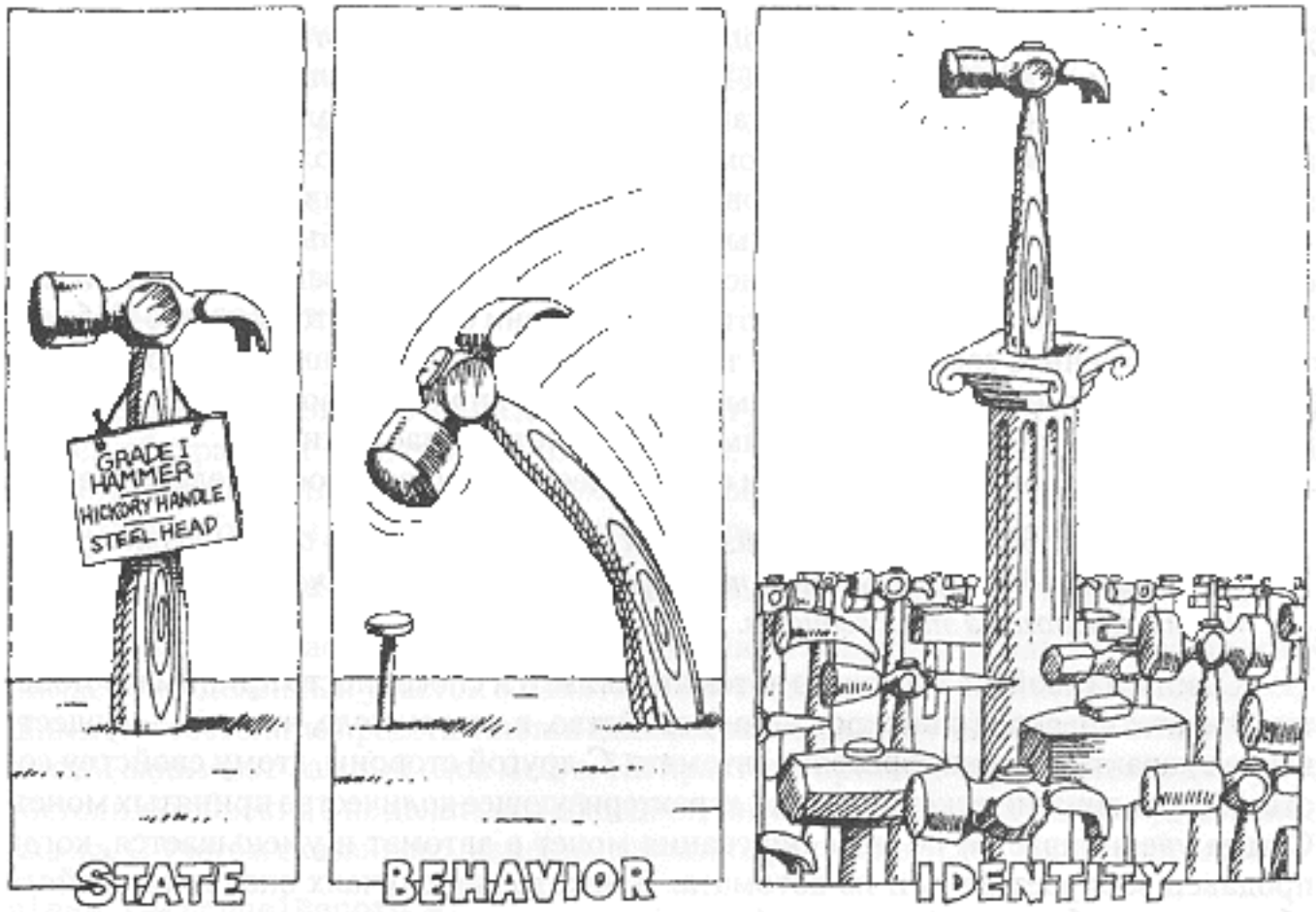
- Класс – конструкция ООП языка, представляющая шаблон, используемый для создания экземпляров класса.

- Объект – это мыслимая или реальная сущность, обладающая характерным поведением и отличительными характеристиками и являющаяся важной в предметной области. // Гради Буч

- Объект может принимать сообщения.

Классы и объекты

- Что является и что не является объектом?



Классы и объекты

- Класс инкапсулирует состояние и поведение предмета реального мира, который он моделирует.
- Класс инкапсулирует состояние посредством данных, называемых атрибутами.
- Т.к. класс является шаблоном объекта, то при создании объекта данного класса в памяти будет создана копия данных, которые определяются классом.

Классы и объекты

- Класс инкапсулирует поведение с помощью повторно используемых фрагменов кода, оперирующих состоянием.
- Эти фрагменты кода называются методами класса.
- Иногда удобно, чтобы данные класса не дублировались при создании каждого экземпляра, а принадлежали классу.

Модуль 2

- Парадигмы программирования
- Классы и объекты
- **Наследование**
- Полиморфизм
- Инкапсуляция
- Цели ООП

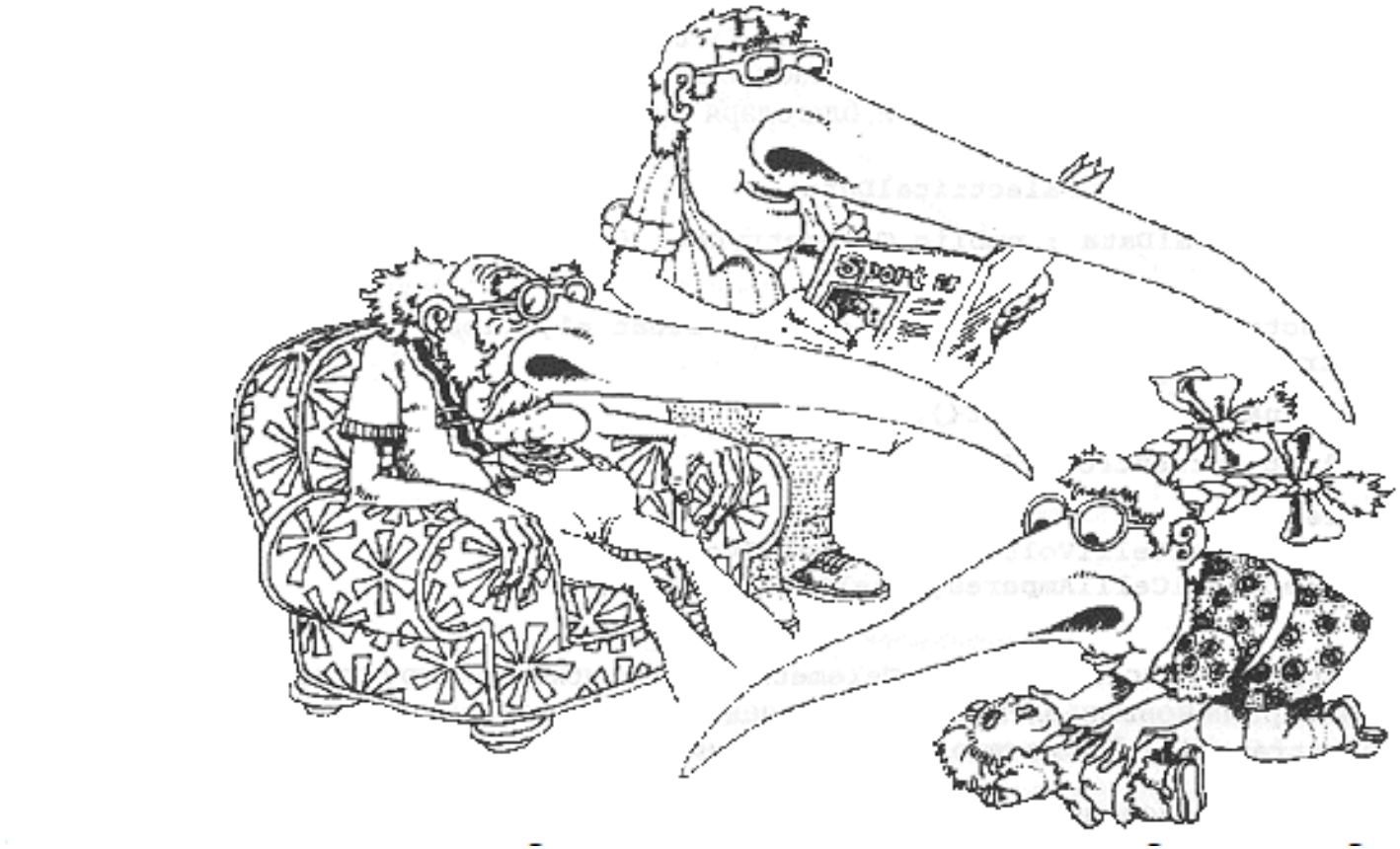
Наследование

- Наследованием называется возможность порождать один класс от другого с сохранением всех свойств и методов класса-предка и добавляя, при необходимости, новые свойства и методы.

- Набор классов, связанных отношением наследования, называют иерархией.

- Наследование призвано отобразить такое свойство реального мира, как иерархичность.

Наследование



Дочерний класс может унаследовать структуру и поведение родительских классов

Наследование

- Наследование вводит иерархию “общее/частное” в которой подкласс наследует от одного или нескольких более общих суперклассов.

- Подклассы обычно дополняют или переопределяют унаследованную структуру или поведение.

Модуль 2

- Парадигмы программирования
- Классы и объекты
- Наследование
- **Полиморфизм**
- Инкапсуляция
- Цели ООП

Полиморфизм

- Возможность объектов с одинаковой спецификацией иметь разную реализацию.

  (разная обработка сообщений в разных классах)

- Методам с одним и тем же именем соответствует разный программный код, в зависимости от того, объект какого класса используется при вызове данного метода.

- Полиморфизм обеспечивается тем, что в классе-потомке изменяют реализацию класса-предка с обязательным сохранением сигнатуры метода.

Полиморфизм

- В общем виде полиморфизм формулируется как взаимозаменяемость объектов с одинаковым интерфейсом.

Интерфейс

- Интерфейс – это семантическая и синтаксическая конструкция в коде программы, используемая для спецификации услуг, предоставляемых классом или компонентом.

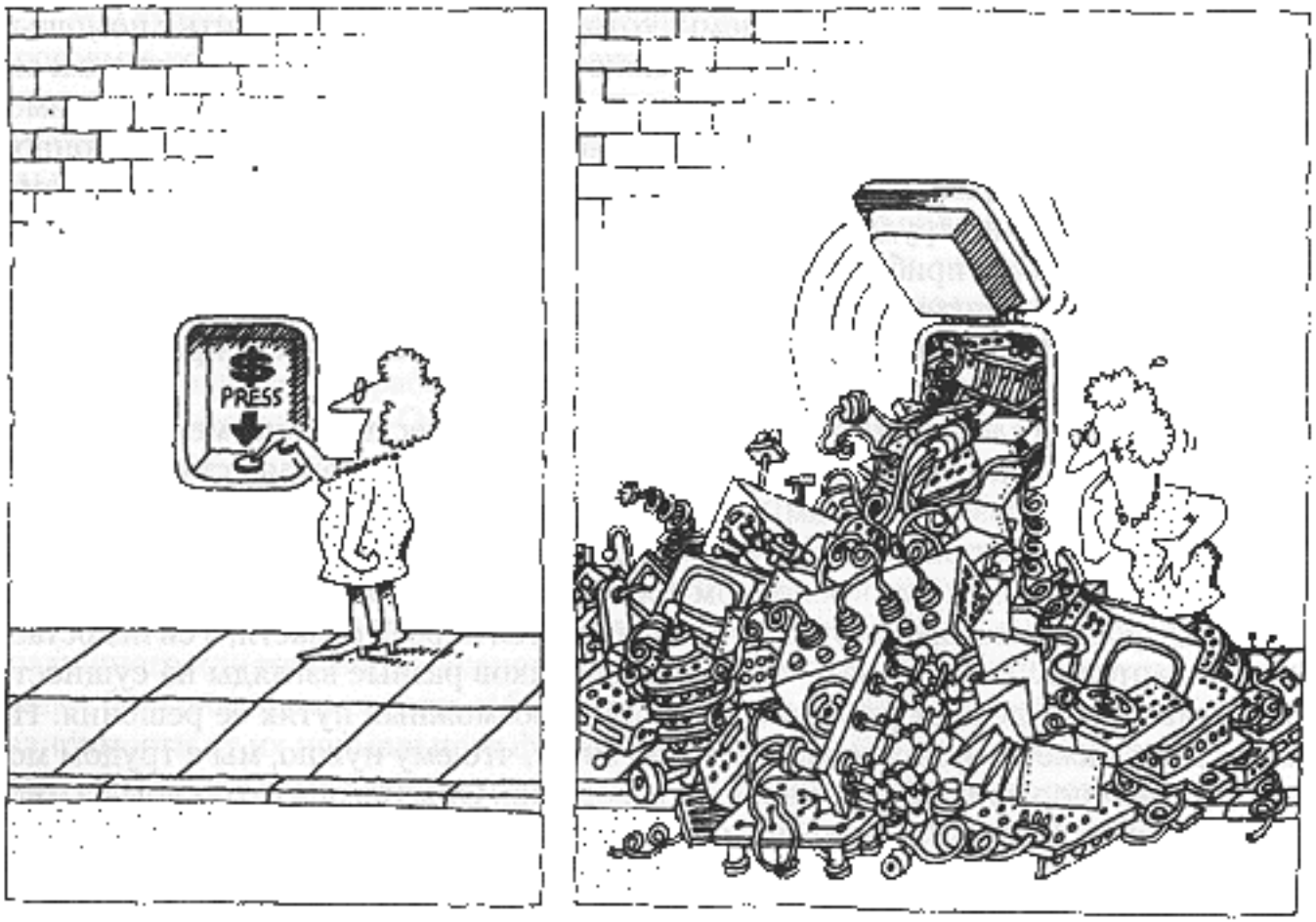
- Семантическая нагрузка – пользователям необходимо знать как пользоваться вещью, не вдаваясь в ее реализацию.

Интерфейс

- Например, интерфейс “ехать” понятен и прост в использовании.

- Реализаций при этом может быть много и они могут подменяться: ехать на машине, на автобусе, на велосипеде и т.д.

Интерфейс



Задача разработчиков программной системы - создать иллюзию простоты.

Модуль 2

- Парадигмы программирования
- Классы и объекты
- Наследование
- Полиморфизм
- **Инкапсуляция**
- Цели ООП

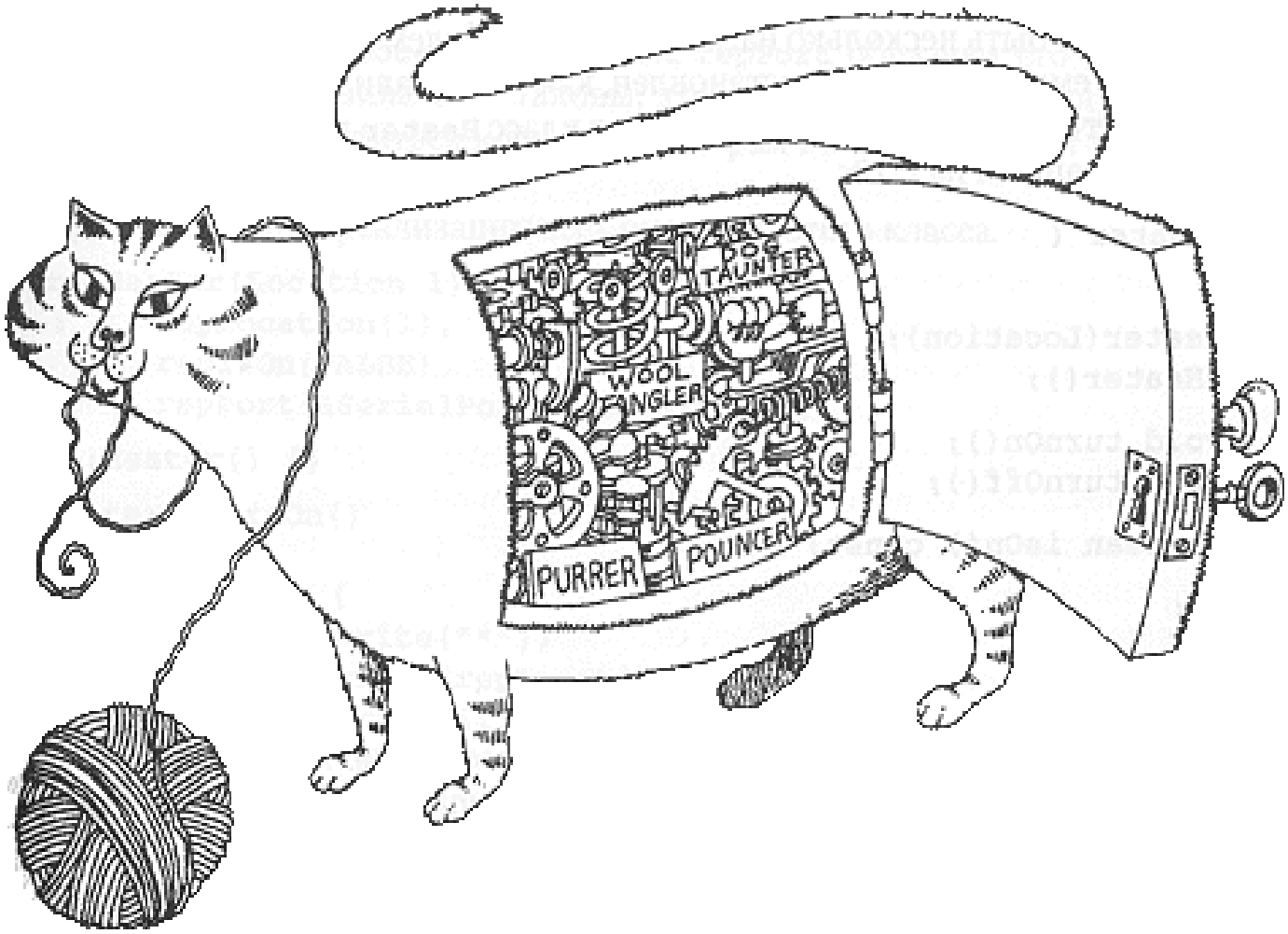
Инкапсуляция

- Инкапсуляция – механизм языка программирования, который ограничивает доступ к составляющим объект компонентам.

  (сокрытие деталий реализации за набором допустимых сообщений)

- К инкапсулированной переменной можно обратиться, когда пишешь реализацию класса.

Инкапсуляция



Инкапсуляция

- Область применения:
  - При необходимости максимальной локализации предстоящих изменений, когда изменяется только работа объекта, а не программа.
  - Необходимость очистки глобальной области видимости.

```
class A:
   hidden number a
   hidden number b

   hidden doSomething():

   returnSomething():
      return a
```

Модуль 2

- Парадигмы программирования
- Классы и объекты
- Наследование
- Полиморфизм
- Инкапсуляция
- **Цели ООП**

Цели ООП

- ООП вводит значительное число новых концепций, требует привлечения определенных усилий на создание правильного дизайна системы, при этом достигаются следующие преимущества.
  - Высокая степень повторно используемого кода
  - Упрощение понимания человеком
  - Возможность локализации модификаций кода
  - Создание абстракций – способствует лучшему пониманию приложения.

Упражнение 3

Анализ предметной области в произвольной нотации.

Модуль 2

- Парадигмы программирования
- Классы и объекты
- Наследование
- Полиморфизм
- Инкапсуляция
- Цели ООП