Южный федеральный университет
Факультет –математики, механики и компьютерных наук
кафедра математического моделирования

Надолин Константин Аркадьевич

Объектно-ориентированное программирование на C++

Спецкурс для студентов отделения
«Прикладная математика и информатика»

Вводная лекция

- ПОП и ООП
- Концепция ООП, понятие программы
- Роль языка программирования
- Принципы ООП
- Сравнение ООП и ПОП

Литература

- Буч Г. Объектно-ориентированное проектирование с примерами применения. – М.:Конкорд, 1992.–519 с.
- Страуструп Б. Язык программирования C++, 3-е изд. – СПб.; М.: “Невский Диалект” – “Издательство БИНОМ”, 1999. – 991 с. http://www.awl.com/cp/stroustrup3e

Дополнительная литература

- Шилд Г. Теория и практика C++. – СПб.: BVH-Санкт-Петербург, 1996. – 416 с.
- Элджер Дж. C++: библиотека программиста. – СПб.: Питер, 2000. – 320 с.
- Пол А. Объектно-ориентированное программирование на C++. – СПб.; М.: “Невский Диалект” – “Издательство БИНОМ”, 1999. – 462 с.
- Бабэ Б. Просто и ясно о Borland C++. – М.:БИНОМ, 1996. – 416 с.
- Сван Т. Освоение Borland C++5. – К.: Диалектика, 1996. – 576 с.

ПОП и ООП

- ПОП – процедурно-ориентированное программирование – это программирование задач обработки данных (алгоритмически детерминированы), основанное на составлении и реализации ***алгоритма решения задачи.***
- Программа – алгоритм, записанный на языке программирования в виде набора процедур (процедурные абстракции)
- В основе ПОП лежит *декомпозиция действия* (алгоритмическая декомпозиция)
- Структурное и модульное программирование

Концепция ООП

- ООП – это программирование задач имитационного моделирования (алгоритмически не заданных), основанное на формировании (конструировании и реализации) ***языка предметной области.***
- Язык (словарь) предметной области: сущности и действия с ними (типы данных и операции).
  - формирование абстракции понятий
  - формирование иерархии понятий
  - реализация понятий
  - использование понятий
  - модификация понятий
- *Примеры.* 1) Арифметика; 2) Строки и файлы в С

Определение ООП

Объектно-ориентированное программирование – это ***методология*** программирования, которая основана на представлении программы в виде совокупности ***объектов***, каждый из которых является реализацией некоторого ***класса***, а классы образуют ***иерархию***, основанную на принципах ***наследования*** *[Г.Буч]*.

ООП и язык

- ***Объектно-ориентированный язык*** – это язык программирования, который ***не позволяет*** отступать от принципов ООП при написании программ (например, Smalltalk).
- ***Язык, поддерживающий ООП*** – это язык программирования, который имеет все необходимые средства поддержки ООП, но не препятствует ПОП (например, C++).

Принципы ООП

- Абстрагирование
- Инкапсуляция
- Модульность
- Иерархия

- Типизация и полиморфизм
- Параллелизм
- Устойчивость

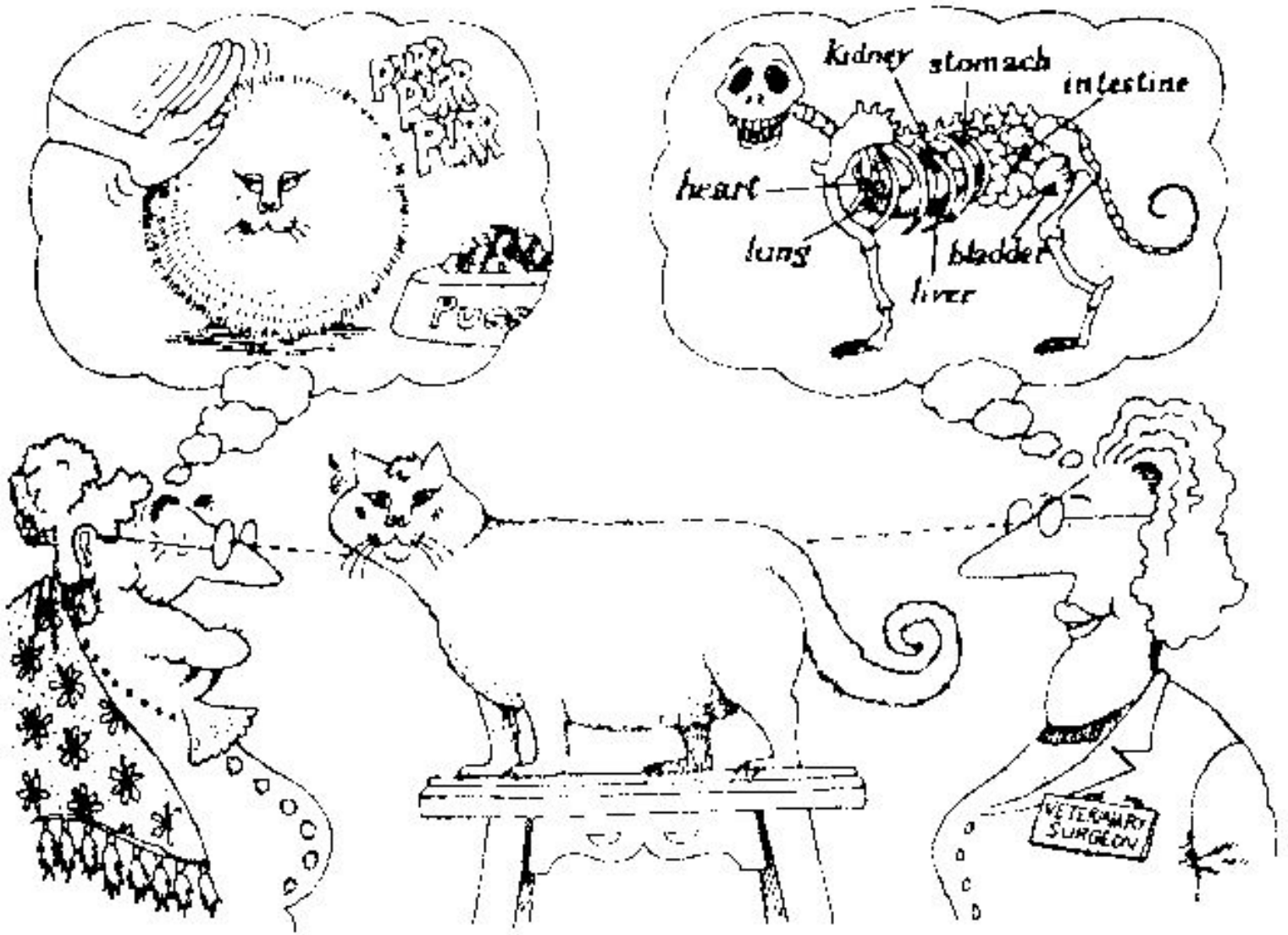
Принцип абстрагирования

Абстрагирование – один из главных способов решения сложных задач

Абстракция – это такие существенные характеристики некоторого объекта, которые отличают его от всех других видов объектов и четко определяют особенности данного объекта с точки зрения дальнейшего рассмотрения и анализа.

Абстрагирование концентрирует внимание на *внешних* особенностях объекта и позволяет отделить самые существенные особенности поведения от деталей их осуществления (*барьер абстракции*).

Формирование абстракции



Абстракция определяет интерфейс класса

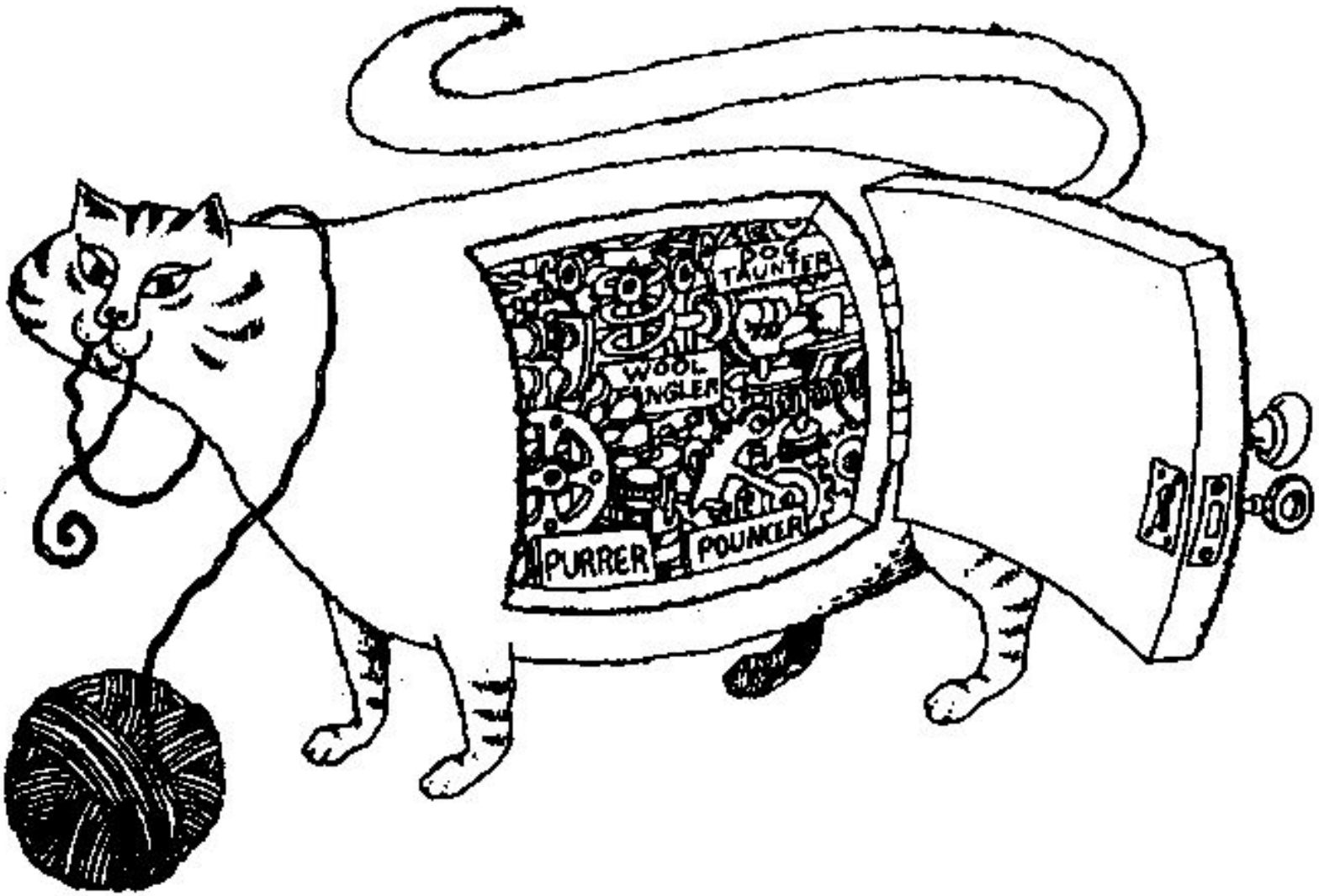
- Формируя абстракцию, мы сосредотачиваем внимание на существенных *с нашей точки зрения* характеристиках объекта
- Абстракция должна охватывать лишь самую суть объекта, не больше, ***но и не меньше***
- Абстракция объекта определяется через ***протокол*** – набор ***определяющих операций***
- Абстракция объекта не должна зависеть от его внутреннего устройства (свойств)
- *Примеры.* 1) Арифметика; 2) Строки и файлы в C

Принцип инкапсуляции

Никакая часть не должна находиться в зависимости от деталей внутреннего устройства других частей

- ***Инкапсуляция*** определяет ***явные*** барьеры между различными абстракциями, скрывает внутреннее устройство объекта и не позволяет объектам-пользователям различить (и использовать) особенности его внутреннего устройства.
- Абстракция и инкапсуляция дополняют друг друга
  - Абстракция определяет интерфейс объектов класса
  - Инкапсуляция определяет реализацию объектов класса
- *Примеры.* 1) Арифметика; 2) Строки и файлы в C

Скрытие данных



Принцип модульности

Модульность, как самостоятельная концепция – это локализация частей программы и ее разделение на раздельно компилируемые фрагменты, имеющие средства взаимосвязи

- Модульная декомпозиция
  - логическая локализация действия (функции), абстракции данных (класс), и имен (пространства имен)
  - физическая локализация и раздельная компиляция (файлы)
  - разделение интерфейса (**.h**-файлы) и реализации (**.cpp**-файлы) и их связь (**#include**)
- Модульное агрегирование
  - файлы выполняют роль физических контейнеров, в которые помещаются определения классов и объектов
  - деление программы на модули бессистемным образом гораздо хуже, чем отсутствие модульности вообще
  - влияние на производительность системы (страничный свопинг)

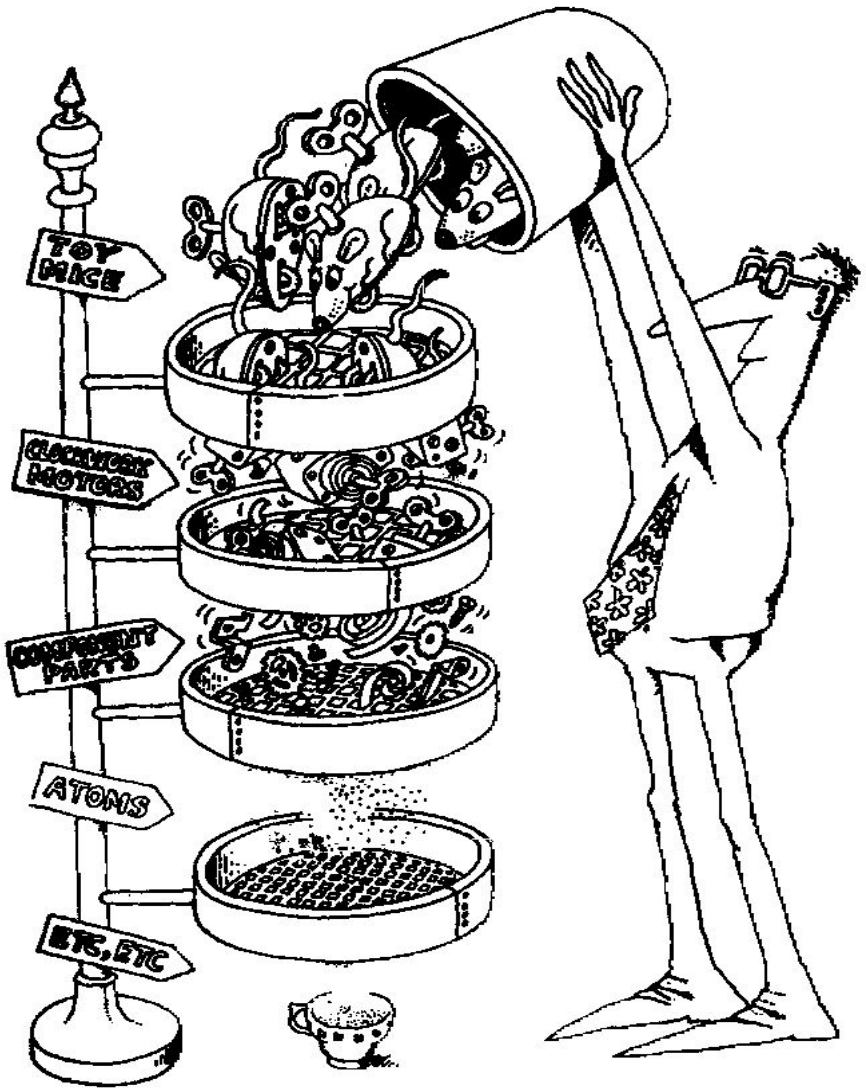
Модульность: декомпозиция и агрегирование

Принцип иерархии

Иерархия – это ранжированная или упорядоченная система абстракций

- Число абстракций в реальных системах велико => для преодоления сложности программы абстракции следует структурировать (иерархия по составу – отношение использования)
- Многие абстракции отражают родственные понятия предметной области => для сокращения описания абстракций их следует структурировать (иерархия по номенклатуре)
- ***Наследование*** – отношение между классами, когда один класс использует строение одного (*простое наследование*) или нескольких (*множественное наследование*) других классов. Это основной вид иерархии по номенклатуре. Наследование – такая иерархия абстракций, в которой подклассы (производные классы) наследуют строение от одного или нескольких суперклассов (базовых классов).

Иерархия



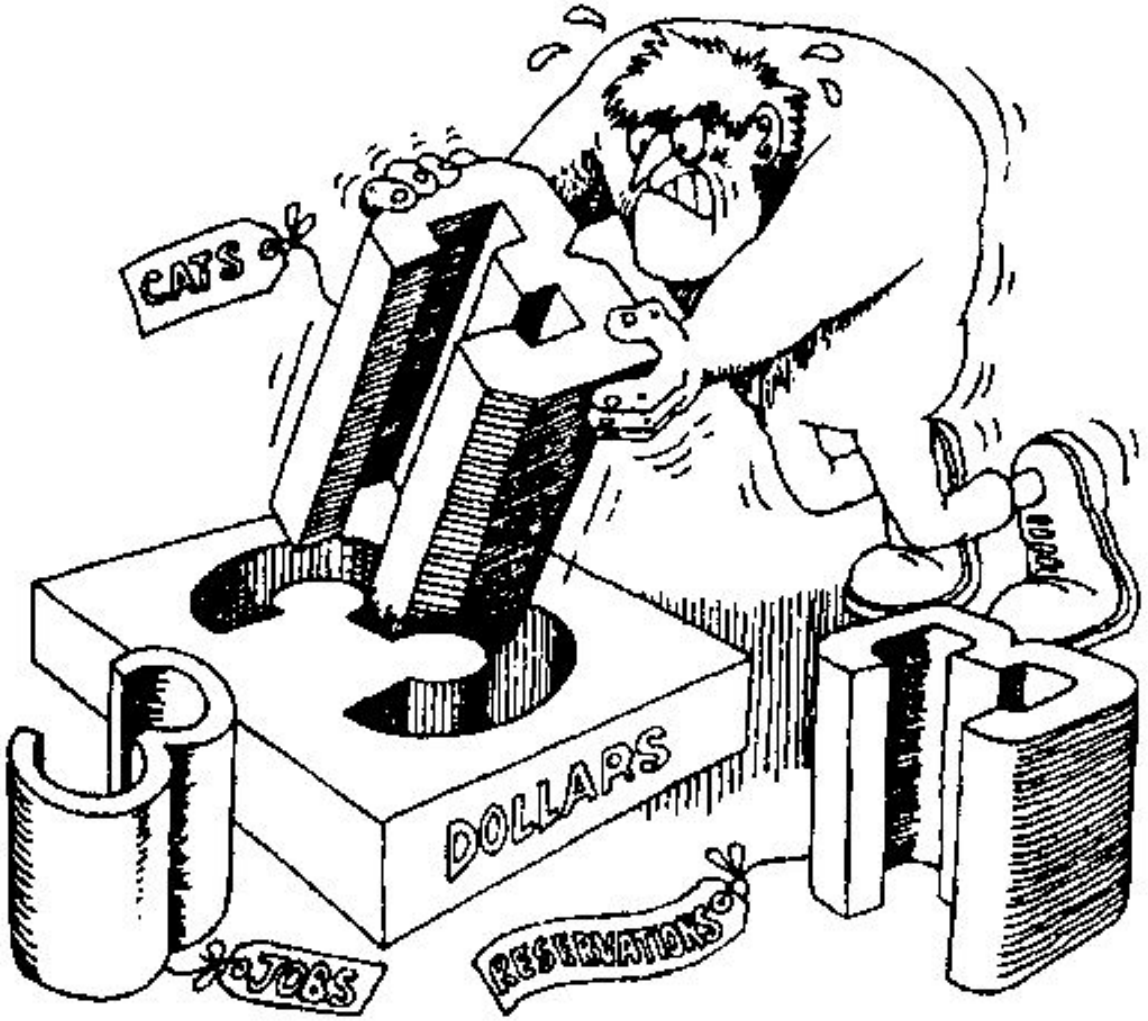
Принцип типизации и полиморфизма

Типизация – это ограничение, накладываемое на класс объектов и препятствующее взаимозамене объектов различных классов (или сильно сужающее возможность такой взаимозамены)

Полиморфизм – важнейшая концепция ООП – свойство, позволяющее *единообразно* оперировать с объектами *разных* типов, *учитывая* их различия

- Связь типизации и полиморфизма
- Контроль соответствия типов
  - на этапе компиляции (раннее или статическое связывание)
  - на этапе выполнения (позднее или динамическое связывание) программы.

Типизация

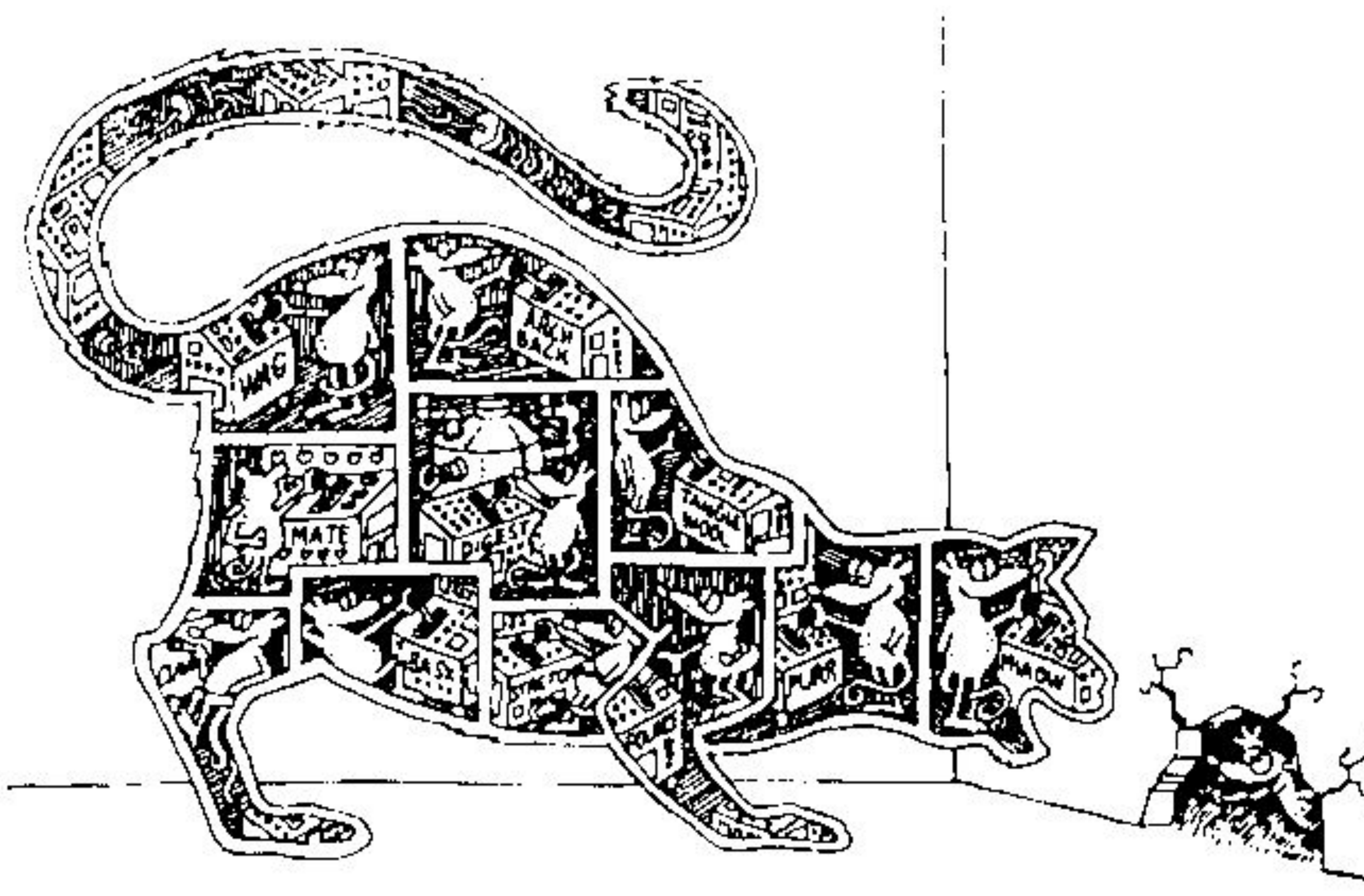


Принцип параллелизма

Параллелизм – свойство объектов находиться в активном, либо пассивном состоянии.

- Для многопроцессорных архитектур объект может представлять собой отдельный канал управления (абстракцию процесса), что упрощает решение вопросов параллелизма (тупики, блокировки и т.п.)
- Для однопроцессорных архитектур реализуется в минимальном виде
- Пример – многооконный интерфейс Windows.

Параллелизм

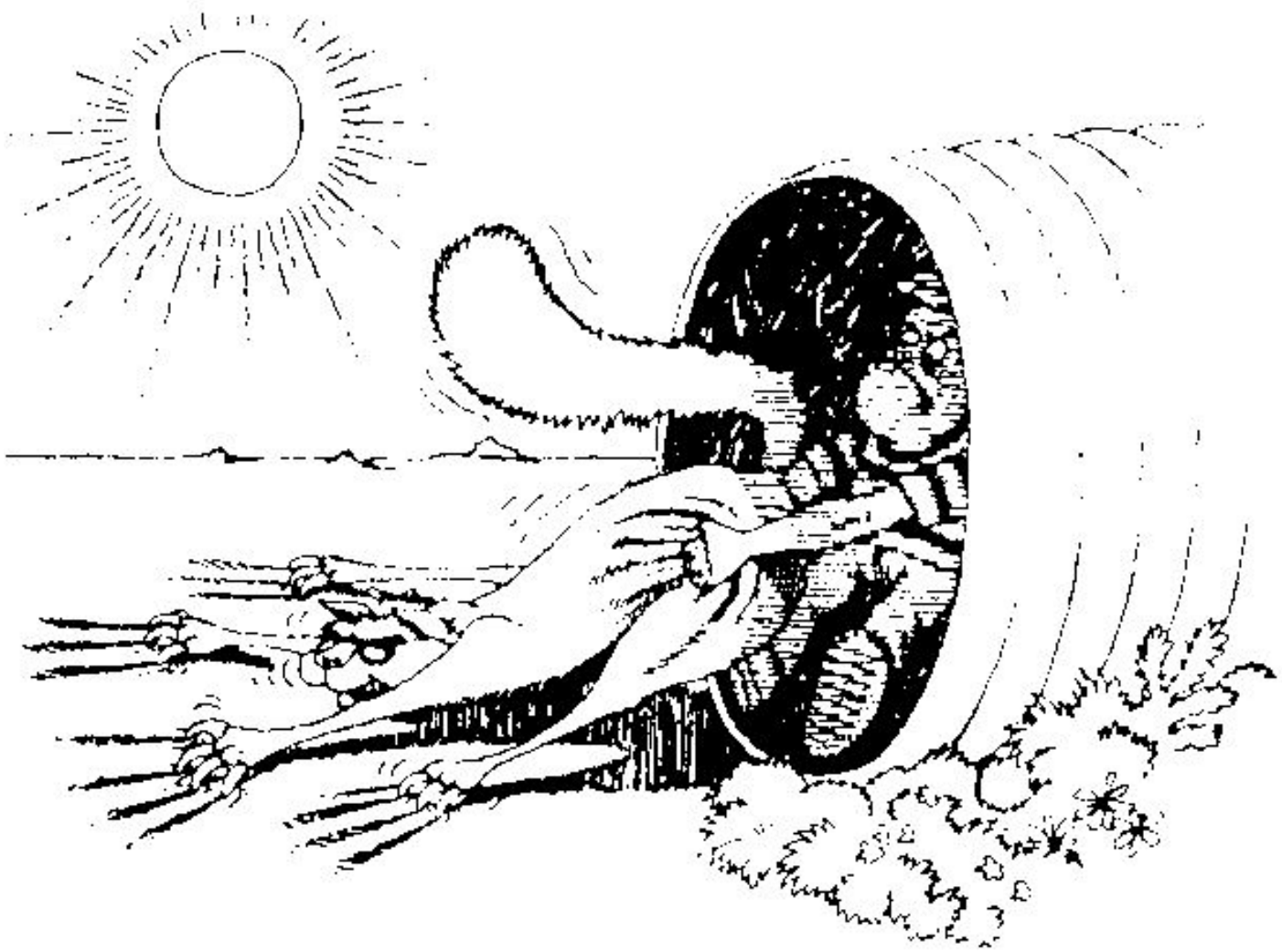


Принцип устойчивости

Устойчивость – свойство объектов сохранять свое состояние и принадлежность к определенному классу.

- Устойчивость в пространстве и времени
  - устойчивость в адресном пространства
  - анонимные объекты (промежуточные результаты)
  - локальные объекты
  - устойчивость относительно породившего процесса
  - устойчивость относительно выполнения программы
  - устойчивость относительно эксплуатации программы
- Корректность объектов
- Инвариант представления

Устойчивость



Сравнение ООП и ПОП

Принцип	ООП	ПОП
Абстракция	Данные	Действия
Инкапсуляция	Свойства	Алгоритмы
Модульность	Классы	Функции
Иерархия	Наследование	Локализация
Типизация	Наследование	Параметры
Параллелизм	Объекты	Функции
Устойчивость	Объекты	Базы данных

Выводы: 1) ООП и ПОП не исключают друг друга
2) Первенство за ООП