Conceitos de Orientação a Objetos

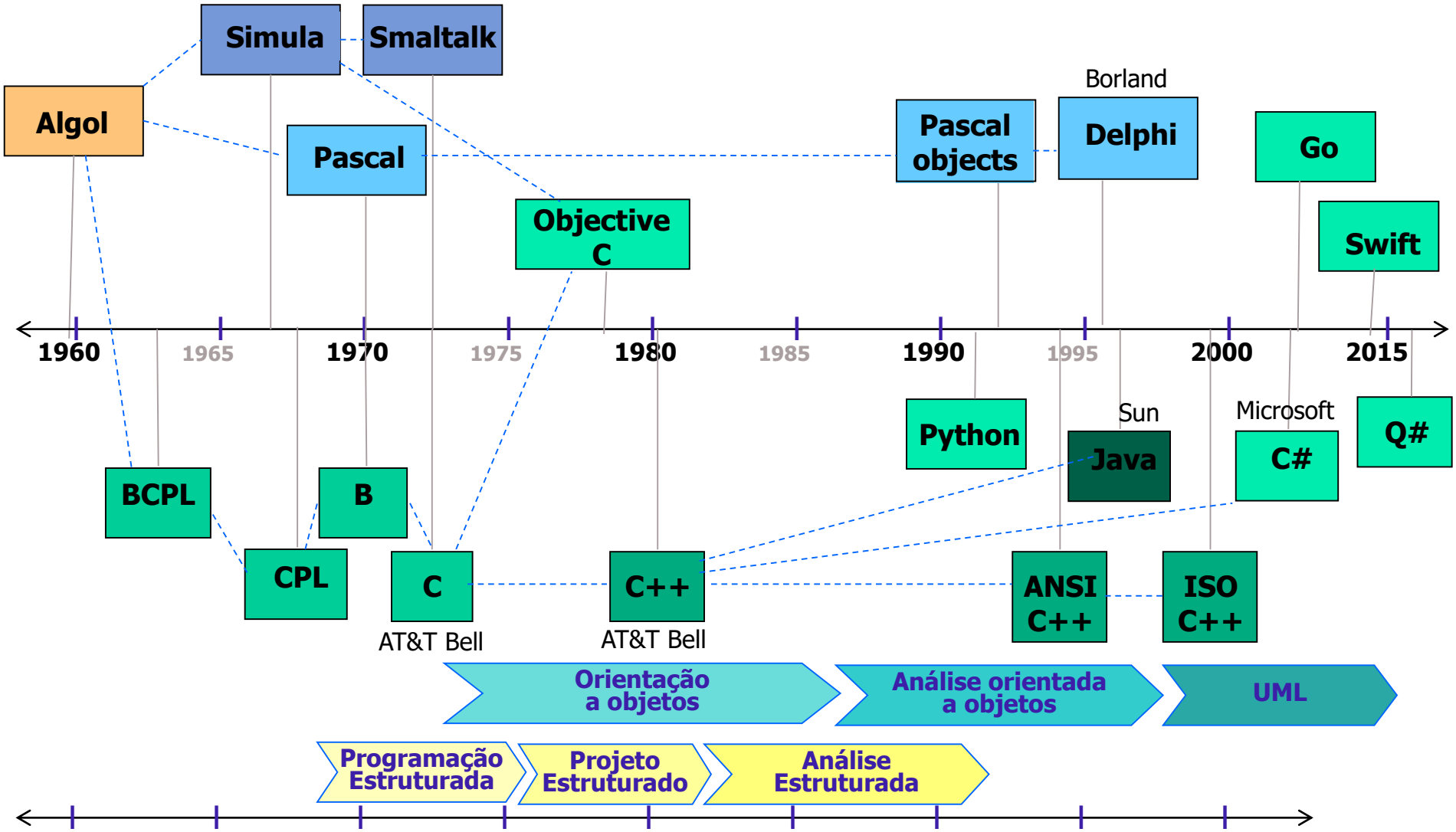
Objetivos

- Comparação Programação Estrutura e POO
- Compreender os conceitos de Programação Orientada a Objetos (POO);
- Entender as vantagens de se utilizar POO;
- Entender os conceitos de POO aplicado a uma linguagem de programação (Java como exemplo);
- Aprender a criar classes e utilizá-las em um contexto prático.

Histórico de Metodologias de Software

- Início Anos 70 - Programação Estruturada
  - Niklaus Wirth
- Fim Anos 70 - Projeto Estruturado
  - Constantine, Yourdon
- Anos 80 - Análise Estruturada
  - Yourdon/DeMarco, James Martim, Chris Gane
- Anos 90
  - Orientação a Objetos
    - Baseado no paradigma estruturado
  - Qualidade de Software
  - UML
  - Componentização

Evolução


Simula
Smaltalk
Algol
Pascal
Objective C
Borland
Pascal objects
Delphi
Go
Swift
1960
1965
1970
1975
1980
1985
1990
1995
2000
2015
Python
Sun
Java
Microsoft
C#
Q#
BCPL
B
CPL
C
AT&T Bell
C++
AT&T Bell
ANSI C++
ISO C++
Orientação a objetos
Análise orientada a objetos
UML
Programação Estruturada
Projeto Estruturado
Análise Estruturada

Estruturada x OO

- Com a orientação a objetos procura-se eliminar as diferenças entre as etapas de análise, projeto e implementação, reabilitando a difamada tarefa de implementação
- O segredo é fazer com que os conceitos de programação, e as notações para programação, sejam suficientemente de alto-nível para que possam servir apropriadamente como ferramentas de modelagem.

Diferenças de Ciclo de Vida

- Paradigma Tradicional:
  - Análise (Pouca), Projeto (Pouco), Codificação (Muita), Teste (Muito, geralmente confundido com codificação), Manutenção (Muita);
- Paradigma OO:
  - Análise (Muita), Projeto (Muito), Re-Análise, Re-Projeto, Codificação (Simplificada), Re-Análise, Re-Projeto, Re-Codificação, Teste (Reduzido), Re-Análise, etc…
- No Projeto OO você usa conceitos de Engenharia de Software para produzir o produto, e não somente como documentação.

Metodologia 'estruturada'

- Criada nos anos 60/70
  - Guerra Fria
  - Grandes Projetos
  - Ambientes de Grande Porte
  - Pouca maturidade em Programação
- Funciona!
  - Quando é bem utilizada…
  - Quando os projetos tem início, meio e fim…
  - Quando os projetos são longos…
  - Quando você tem legado de mainframe.

Metodologia 'estruturada'

- Tradicionalmente, a literatura tratava análise, projeto e implementação como atividades diferentes e, até certo ponto, desconexas pois tinham:
  - diferentes métodos;
  - diferentes notações e
  - diferentes objetivos.

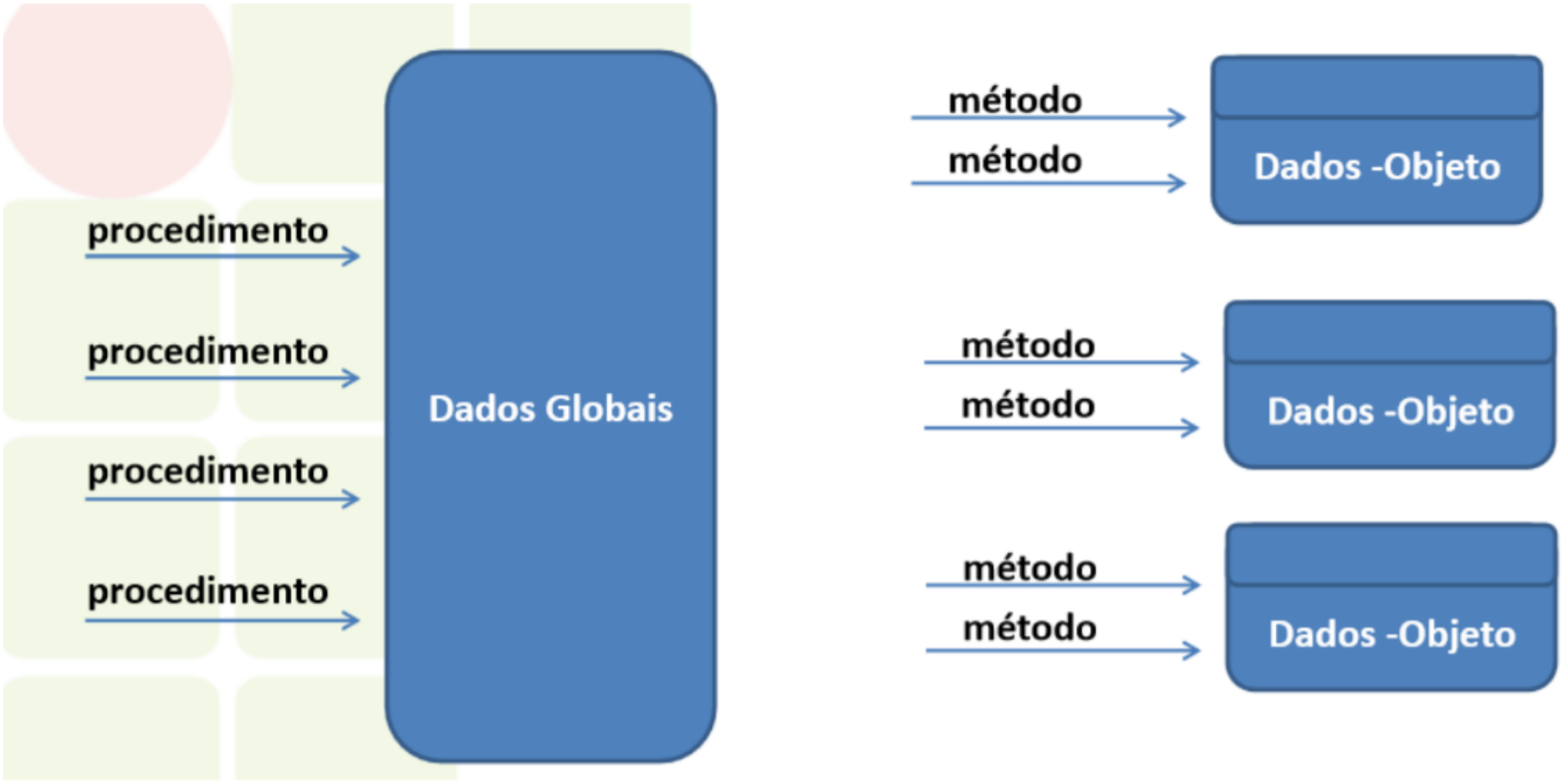
- Na necessidade de especificar o que está prestes a ser implementado tratavam a análise e o projeto como as únicas coisas que realmente interessam. A implementação seria apenas algo inevitável.

Paradigma
Estrutural x Orientação a Objetos

- Na Estruturada
  - Os sistemas são divididos em subprogramas;
  - Fixa a atenção muito mais nos procedimentos que nos dados.

- Na Orientação a Objetos
  - Dados e Procedimentos possuem a mesma importância;

Paradigma
Estrutural x Orientação a Objetos



Paradigma
Estrutural x Orientação a Objetos

- Consiste no mapeamento do problema no mundo real a ser resolvido num modelo computacional.
- Programação Estrutura
  - Consiste na criação de um conjunto de procedimentos (algoritmos) para resolver o problema
  - Encontrar modos apropriados de armazenar os dados

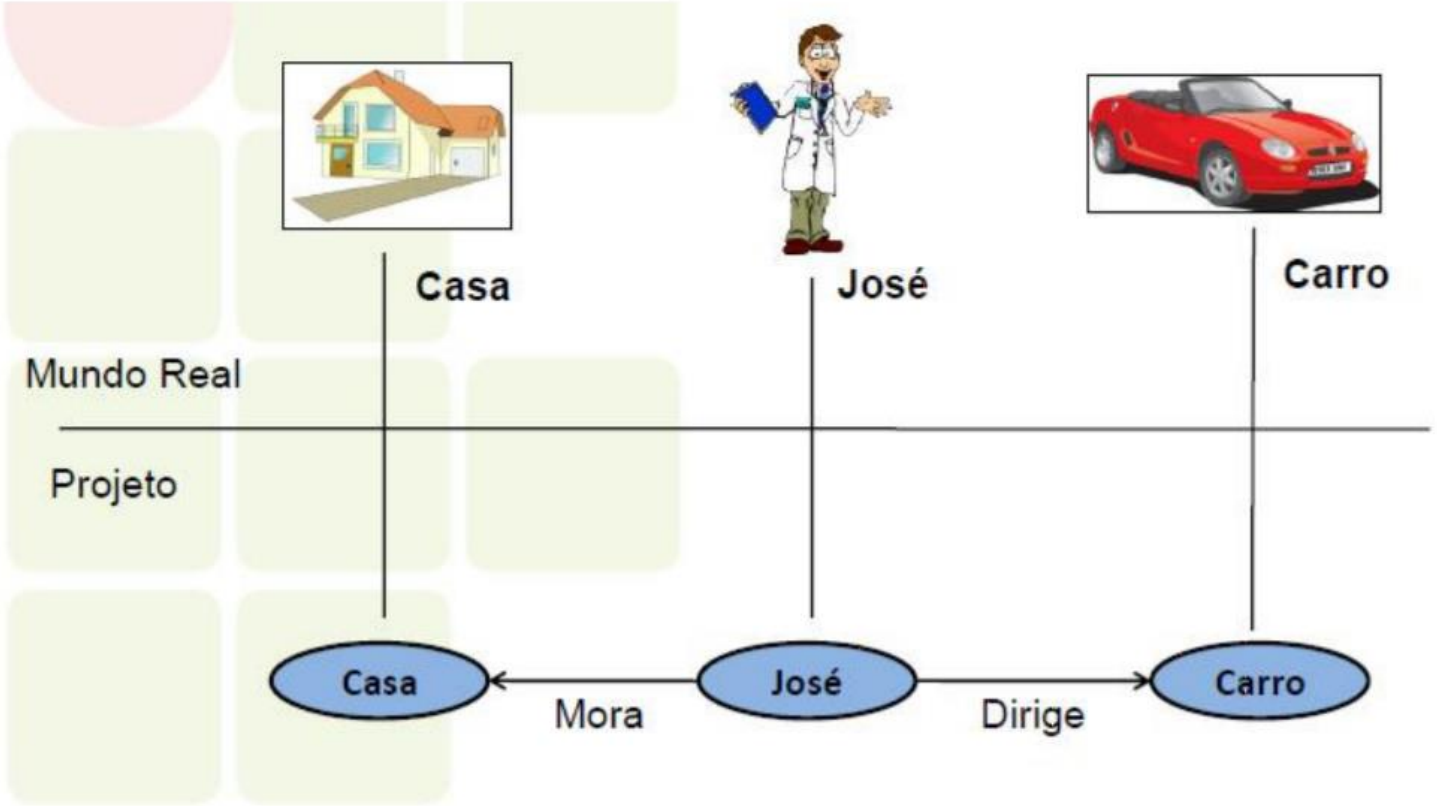
- Programação Orientada a Objetos
  - Consistem em identificar os objetos e as operações relevantes no mundo real
  - O mapeamento desses em representações abstratas no espaço de soluções

Programação Orientada a Objetos

- Paradigma de Programação
  - Dominante nos dias atuais
- Substituiu as técnicas de programação procedimental (estruturada)
- “Fornece um mapeamento direto entre o mundo real e as unidades de organização utilizadas no projeto”
- Diversas unidades de software, chamadas de objetos, que interagem entre si
- Separa claramente a noção de o que é feito de como é feito

Programação Orientada a Objetos

- Representação:



Definição de Objetos

- Um objeto é algo do mundo real :
  - Concreto ou Abstrato
- As percepção dos seres humanos é dada através dos objetos
- Um objeto é uma entidade que exibe algum comportamento bem definido.

Objetos

- Características
  - Dados representam características
    - São chamados atributos
    - São as variáveis do objeto
- Comportamento
  - Operações definem comportamento
    - São os métodos de um objeto
    - São as funções que são executadas por um objeto

Objetos - Propriedades

- Estado
  - Representado pelos valores dos atributos de um objeto
- Comportamento
  - Definido pelo conjunto de métodos do objeto
  - Estado representa o resultado cumulativo de seu comportamento
- Identidade
  - Um objeto é único, mesmo que o seu estado seja idêntico ao de outro;
  - Seu valor de referência
- Os valores dos **DADOS** são modificados a partir das **OPERAÇÕES** sobre estes dados

Objetos - Propriedades

- Estado

Acesa

Apagada

- Comportamento

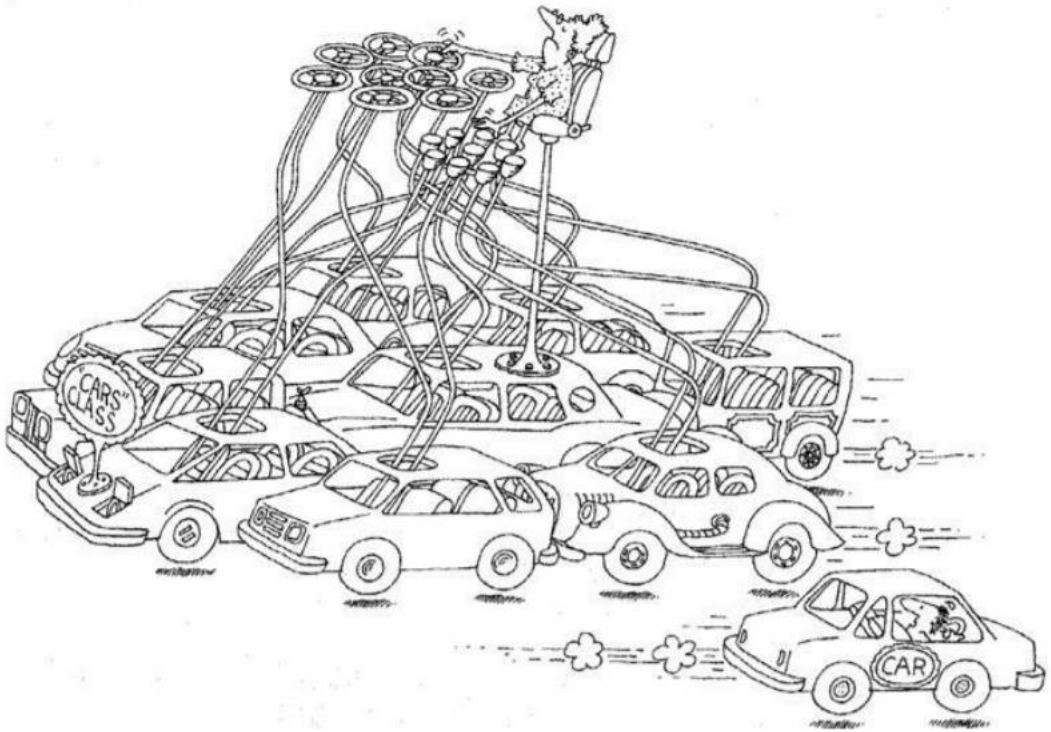
Acender

Apagar

- Identidade

Classes

- São especificações para objetos;
- Representam um conjunto de objetos que compartilham características e comportamentos comuns.



Todo carro tem em comum:

Característica

Cor

Pneu

Direção

Comportamento

Dirigir

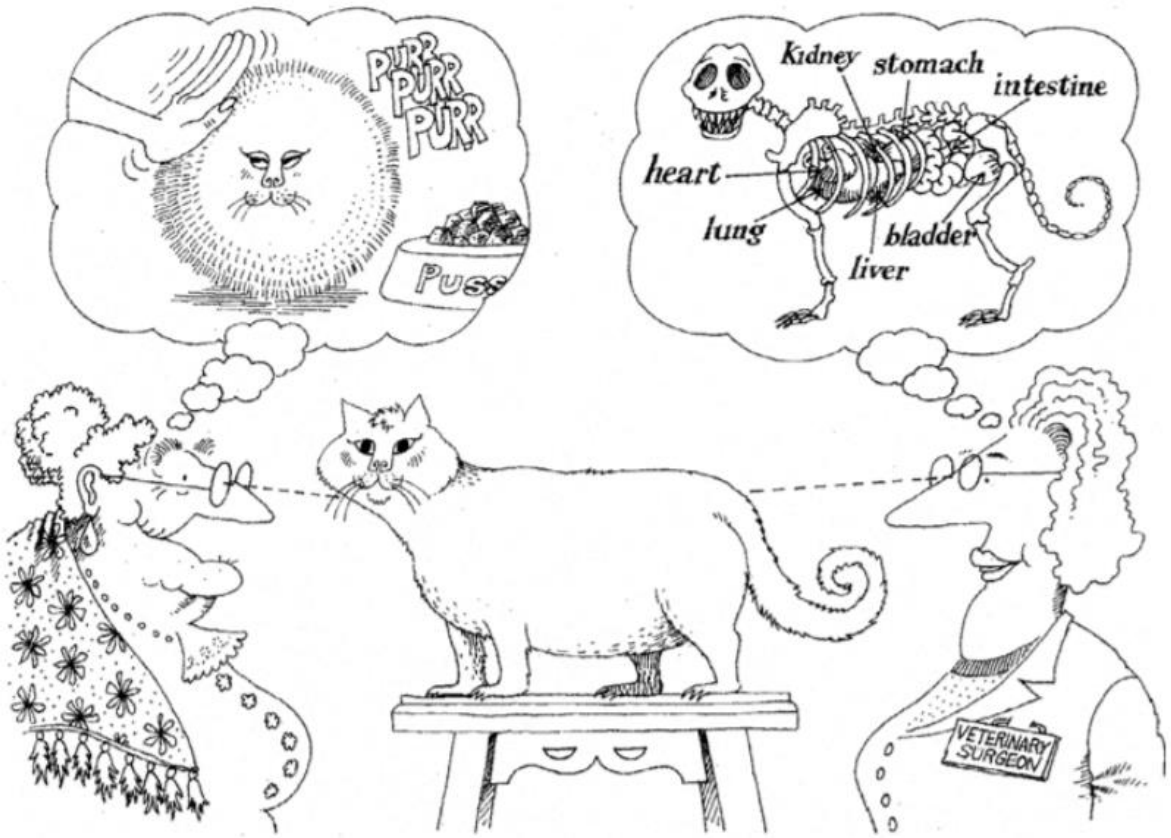
Frear

Abstração

- Abstração é uma das formas fundamentais que nós lidamos com a complexidade.;
- Quando queremos diminuir a complexidade de alguma coisa, ignoramos detalhes sobre as partes para concentrar a atenção no nível mais alto de um problema;
- Não se analisa o “todo”, em POO é importante analisar as partes para entender o todo.

Abstração

- Foca a característica essencial de alguns objetos relativo a perspectiva do visualizador

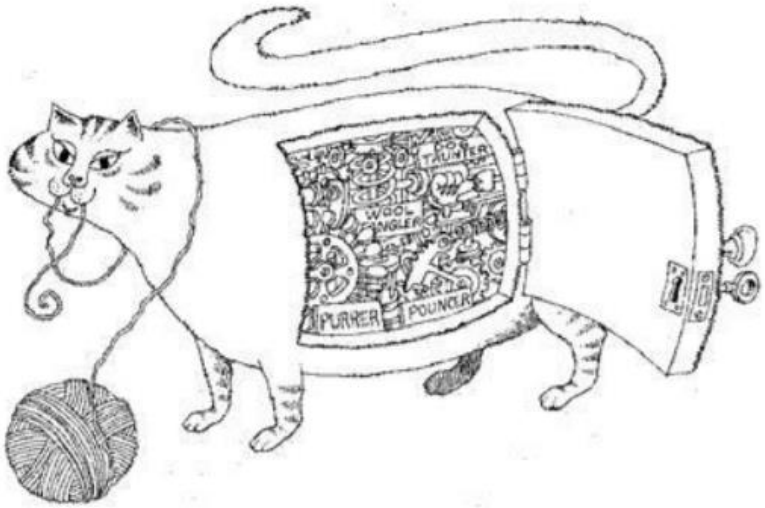


Encapsulamento

- Encapsulamento é o processo de esconder todos os detalhes de um objeto que não contribuem para as suas características essenciais;

- O encapsulamento é o modo de dar ao objeto seu comportamento “caixa-preta”, que é o segredo da reutilização e confiabilidade.

Encapsulamento



Se o estado de um objeto foi modificado sem uma chamada de método desse objeto, então o encapsulamento foi quebrado

Encapsulamento e Abstração

- São conceitos complementares
- Abstração foca sobre o comportamento observável de um objeto, enquanto encapsulamento se concentra na execução que dá origem a esse comportamento

Herança

- A abstração ajuda a diminuir a complexidade.
- Encapsulamento ajuda a gerenciar essa complexidade, ocultando a visão dentro de nossa abstrações.
- A modularidade também ajuda, dando-nos uma maneira de agrupar logicamente abstrações relacionadas.
- Um conjunto de abstrações, muitas vezes forma uma hierarquia, e identificando essas hierarquias no nosso projeto, simplifica grandemente o nossa compreensão do problema.

Herança

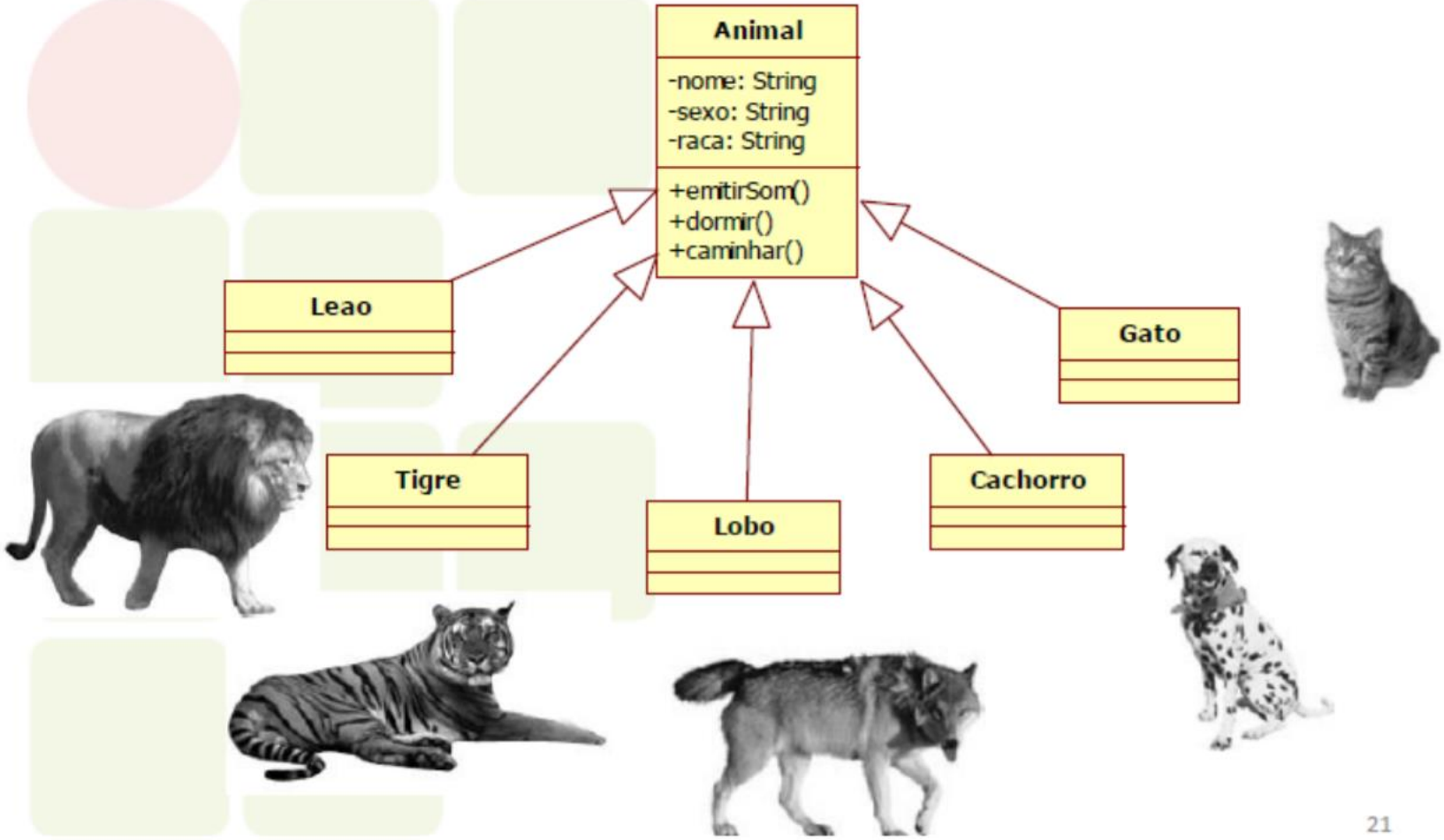
- Herança é o mecanismo para expressar a similaridade entre Classes, simplificando a definição de classes iguais que já foram definidas.
- O que um leão, um tigre, um gato, um lobo e um dálmatas têm em comum?

- Como eles são relacionados?





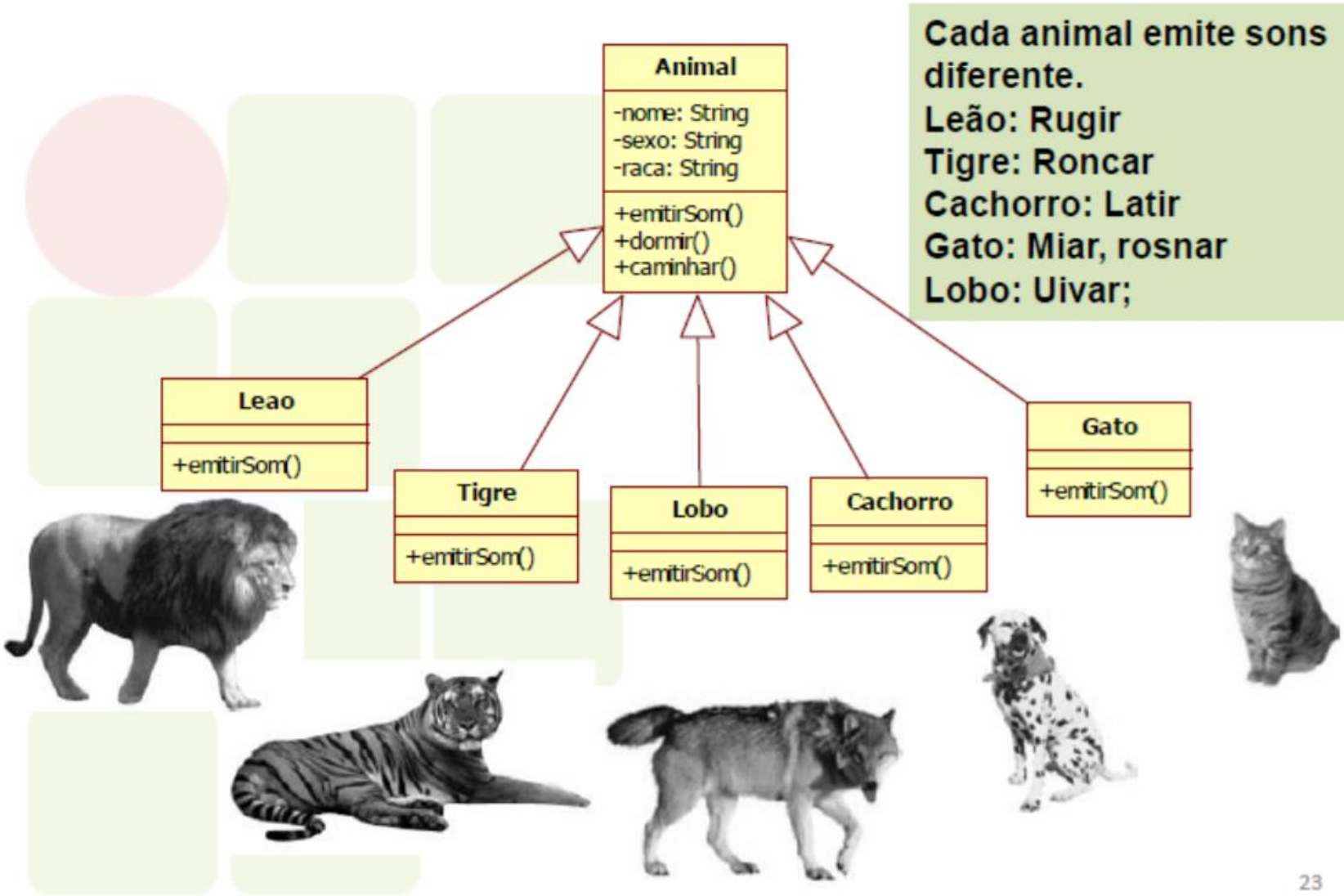
Herança


Animal
-nome: String
-sexo: String
-raca: String
+emitirSom()
+dormir()
+caminhar()
Leao
Tigre
Lobo
Cachorro
Gato

Polimorfismo

- Polimorfimos
  - **Poli** -> varias; **Morfos** -> formas;
- Significa que um objeto pode assumir diferentes formas;
- O conceito de polimorfismo está associado a Herança;
- É caracterizado como o fato de uma operação poder ser implementada de diferentes maneiras pelas classes na hierarquia.

Polimorfismo



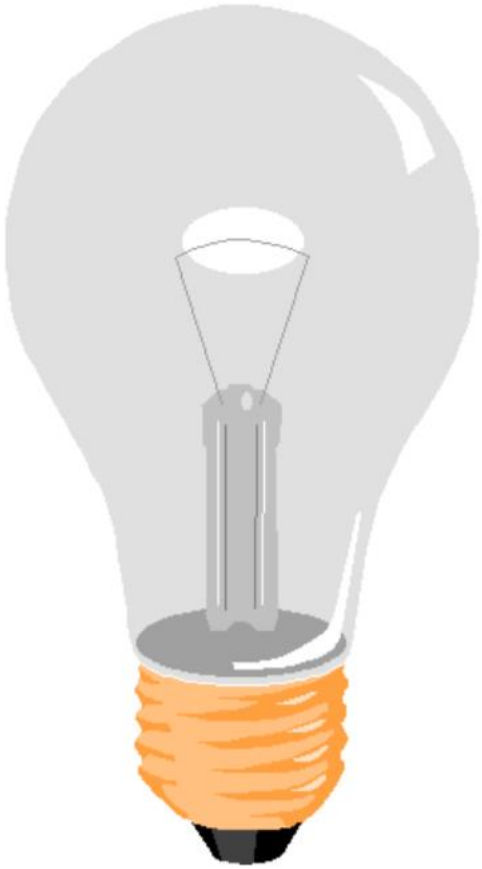
Visibilidade

- Private
  - O nível de acesso se restringe apenas a classe;
  - Não é passado por herança;
- Public
  - O nível de acesso é irrestrito;
  - Por padrão, é a visibilidade definida para métodos e atributos em uma classe
- Protected
  - É visível em toda a classe;
  - É passado por herança (mesmo em pacotes diferentes);

Visibilidade

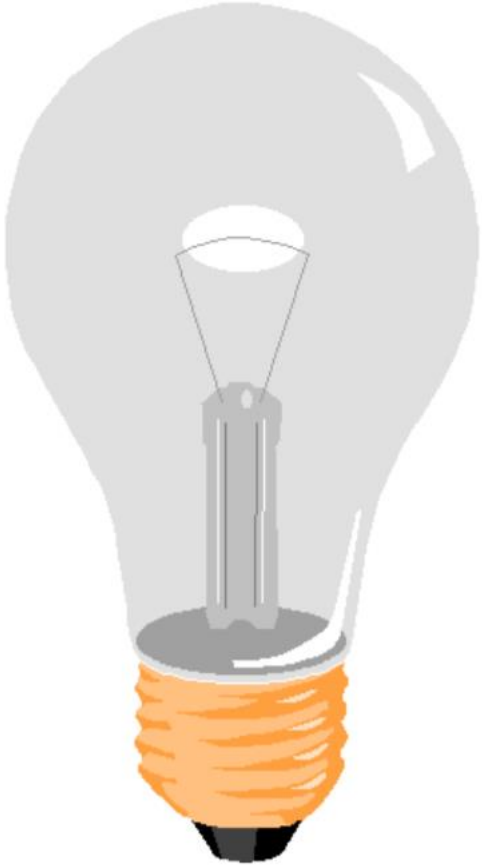
- Internal
  - Com este modificador, o acesso é limitado apenas ao assembly atual.
- Protected Internal
  - Com este modificador, o acesso é limitado ao assembly atual e aos tipos derivados da classe que contém o modificador.

Classes



- Classe Lampada
  - Atributos
    - potencia, ligada
  - métodos
    - ligar, desligar, estaLigada

Classes



Nome da classe

Atributos

métodos