Object-Oriented Programming

CE-1103 Algorithms and Data Structures I

Disclaimer / Descargo de Responsabilidad

Esta presentación corresponde a una guía usada por el profesor durante las clases. La misma ha sido modificada para ser utilizado en el modelo de cursos asistidos por tecnología. No es una versión final, por lo que la misma podría requerir todavía hacer algunos ajustes. Para aspectos de evaluación esta presentación es solo una guía, por lo que el estudiante debe profundizar con el material de lectura asignado y lo discutido en clases para aspectos de evaluación.

This presentation corresponds to a guide material used by the professor during classes. It has been modified to be used in the model of technology-assisted courses. It is not a final version, so it may still require some adjustments. For evaluation aspects, this presentation is only a guide, so the student should delve with the assigned reading material and what has been discussed in class.

What is an Object?

Objects

A general definition

➔ Everywhere you look in the **real world**, you see things or **objects**

- Plants
- Animals
- Cars
- Buildings
- Houses
- And the list can go on for hours or days!

Objects

A general definition

➔ We think in terms of objects

➔ Let’s see how natural this is for humans

Think about a car...

Now describe it!

Objects

A general definition

➔ The car **is**:

- Black
- Has doors that open upward
- Lamborghini
- Made of metal
- Heavy
- Expensive
- Not a Hyundai…
- The same I have in my garage!

Objects

A general definition

➔ The car **is**:

- Black
- Has doors that open upward
- Lamborghini
- Made of metal
- Heavy
- Expensive
- Not a Hyundai…
- The same I have in my garage!

Objects

A general definition

➔ The car **is**:

- Black
- Has doors that open upward
- Lamborghini
- Made of metal
- Heavy
- Expensive
- Not a Hyundai…
- The same I have in my garage!

Only mentions the attributes of the object

Objects

A general definition

➔ The car **can**:

- Accelerate
- Break
- Turn
- Play music
- Connect to Internet

Objects

A general definition

➔ The car **can**:

- Accelerate
- Break
- Turn
- Play music
- Connect to Internet

This includes the behavior of the object

Objects

A general definition

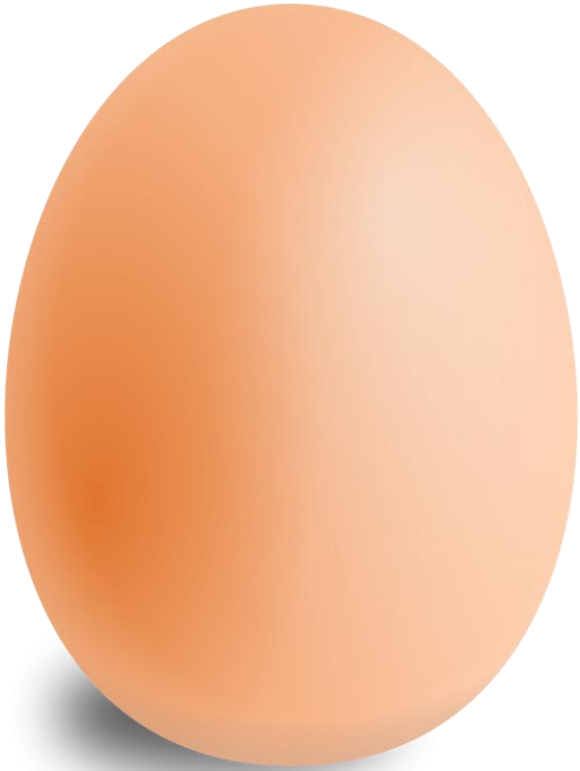
➔ Remember these two keywords: attributes and behavior

➔ Every object in real life has both **attributes** and **behavior** and can be described with them

➔ We learn about the objects by observing the attributes and the behavior of each one

Let’s bake a cake

What objects are involved?

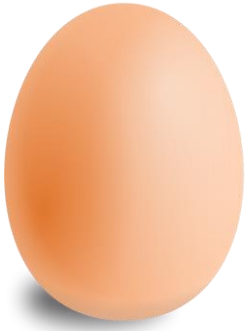



Pillsbury
Moist Supreme
AS ALWAYS, PUDDING IN THE MIX
Makes 24 Cupcakes*
Pillsbury
Moist Supreme®
PREMIUM CAKE MIX
Classic Yellow
NET WT 15.25 OZ (432g)
SERVING SUGGESTION
TRY OUR OTHER PRODUCTS!


LE CREUSET


TAYLOR
cups
4
3/4
1/2
1/4
3
3/4
1/2
1/4
2
3/4
1/2
1/4
1
3/4
1/2
1/4
4
2/3
1/3
3
2/3
1/3
2
2/3
1/3
1
2/3
1/3
TAYLOR
CUPS

Now define the relationship between the objects

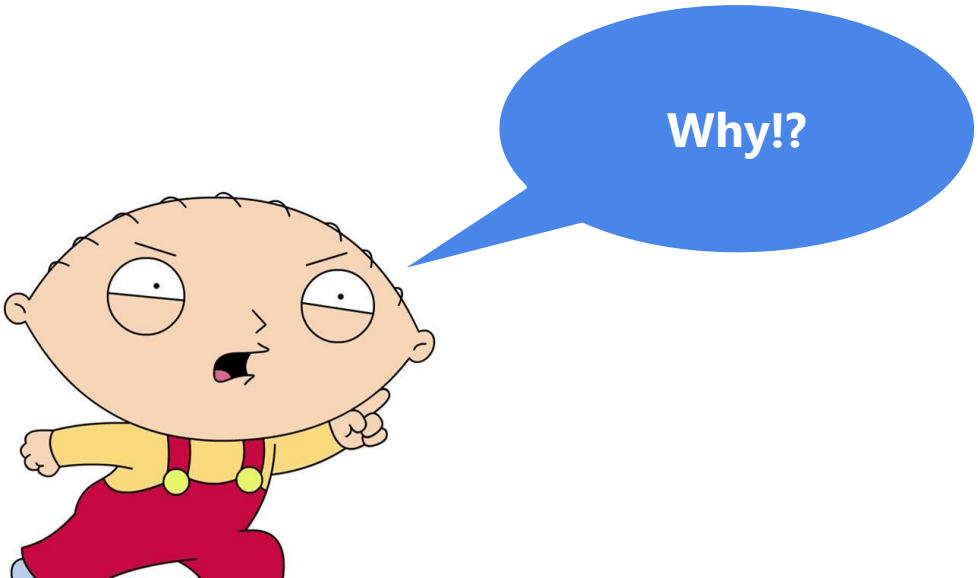



Pillsbury
AS ALWAYS, PUDDING IN THE MIX
Moist Supreme
PREMIUM CAKE MIX
Classic Yellow

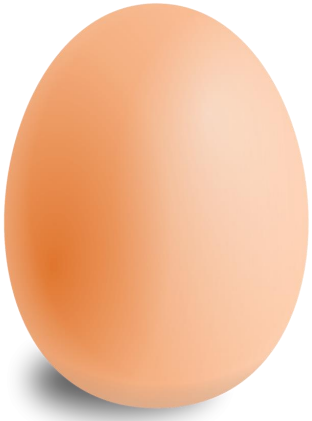

TAYLOR
cups

We have to define the attributes of each one

We have to define the attributes of each one

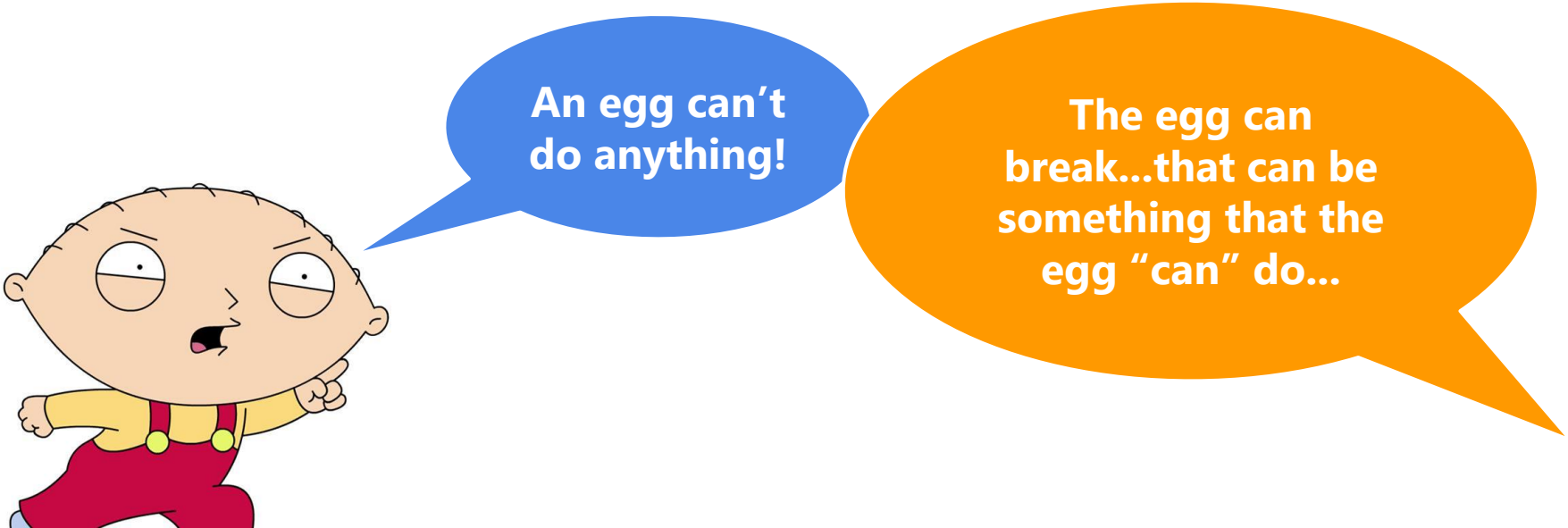


Ok..you need an egg, but what kind of egg?



Now define the behavior...

Now define the behavior…



About the example

➔ This was a very simple example of how we usually solve a problem in real

➔ In real life we don’t pause to think on the attributes and behavior, that is something implicit most of the time

➔ But that doesn’t mean the objects we used don’t have attributes and behavior

About the example

➔Keep in mind what we learned

➔Programming using OO is just as simple. **You already know how to do it!**

What is a paradigm?

par·a·digm

/ˈperəˌdīm/

noun

noun: **paradigm**; plural noun: **paradigms**

1. *technical*
   a typical example or pattern of something; a model.
   "there is a new paradigm for public art in this country"
   synonyms: model, pattern, example, exemplar, template, standard, prototype, archetype
   "why should your sets of values be the paradigm for the rest of us?"
   - a worldview underlying the theories and methodology of a particular scientific subject.
     "the discovery of universal gravitation became the paradigm of successful science"

par·a·digm

/ˈperəˌdīm/

noun

noun: **paradigm**; plural noun: **paradigms**

1. *technical*
   a typical example or patt
   "there is a new paradig
   synonyms: model, patt tandard, prototype, archetype
   "why sho adigm for the rest of us?"

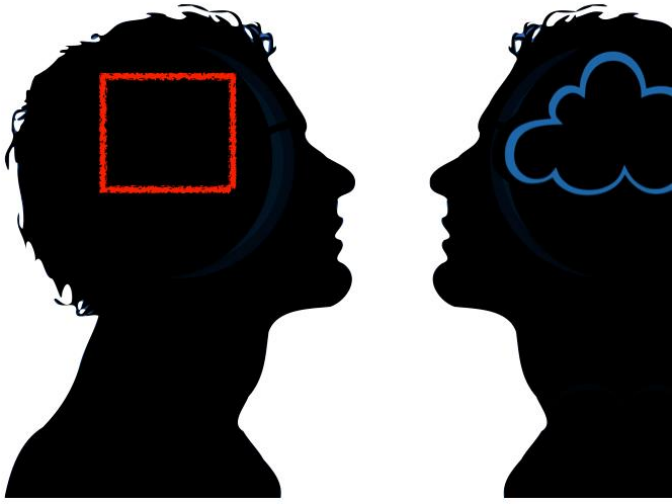
   - a worldview underlying the theories and methodology of a particular scientific subject.
     "the discovery of universal gravitation became the paradigm of successful science"

The overall perspective from which one sees and interprets the world

In the context of computer science...

Programming paradigm

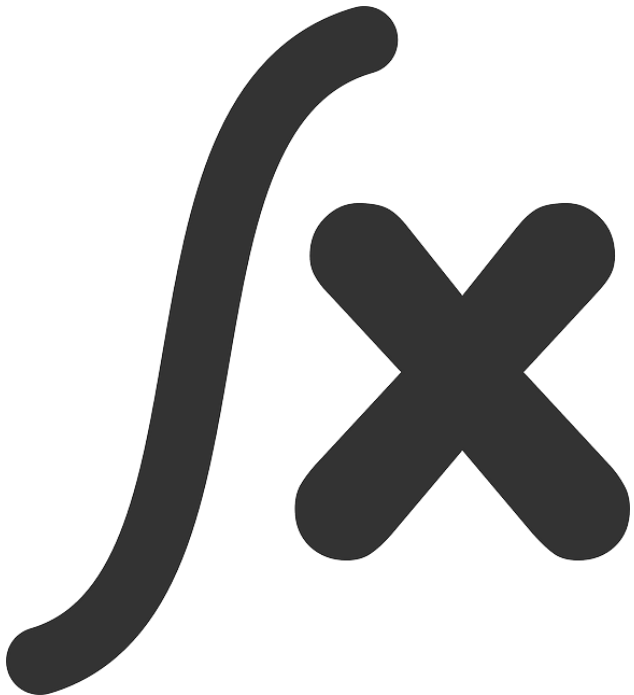
- ➔ Is a **fundamental style** of computer programming
- ➔ Serves as a way of building the structure and elements of computer programs
- ➔ Is how the language “sees” the world



Programming paradigm

➔ In the **functional paradigm** the real world is modeled as mathematics and functions theory

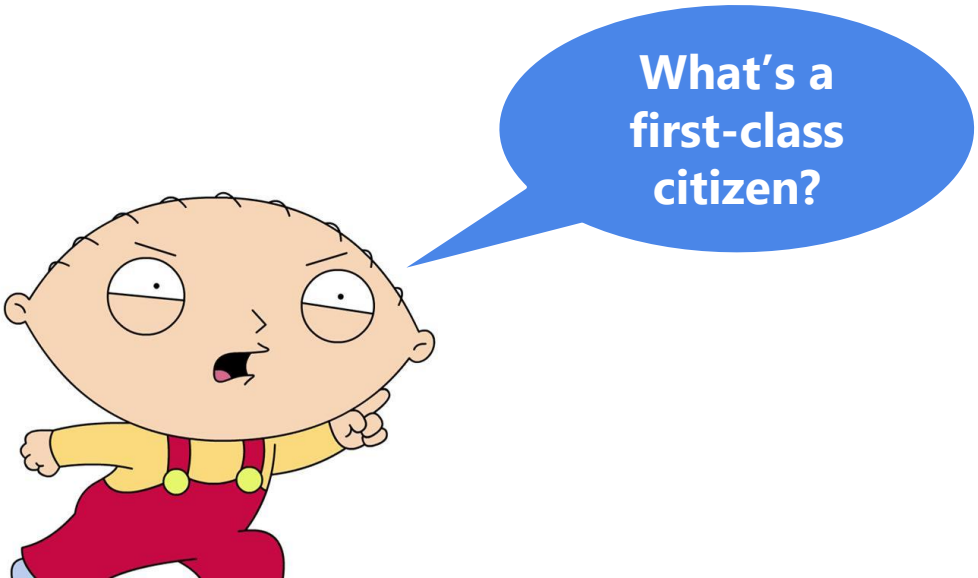
- Functions are first-class citizens
- Scheme, Lisp, F#

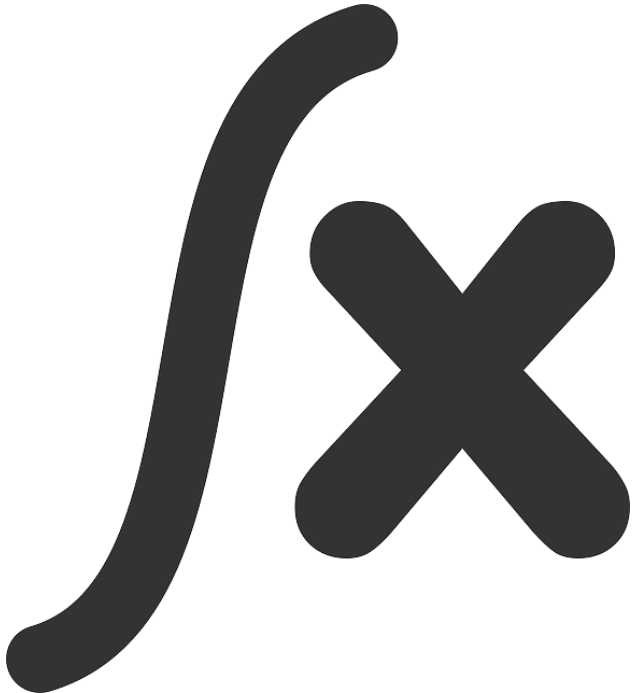


Programming paradigm

➔ In the **functional paradigm** the real world is modeled as mathematics and functions theory

- Functions are first-class citizens
- Scheme, Lisp, F#

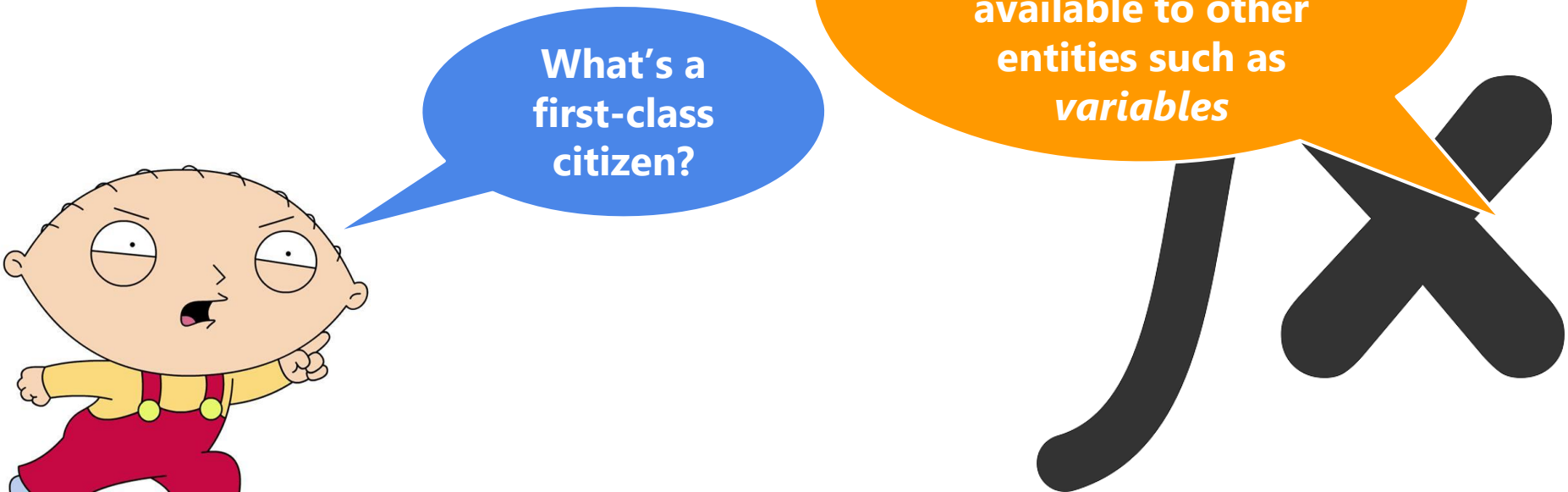




Programming paradigm

➔ In the **functional paradigm** the real world is modeled as mathematics and functions theory

- Functions are first-class citi
- Scheme, Lisp, F#



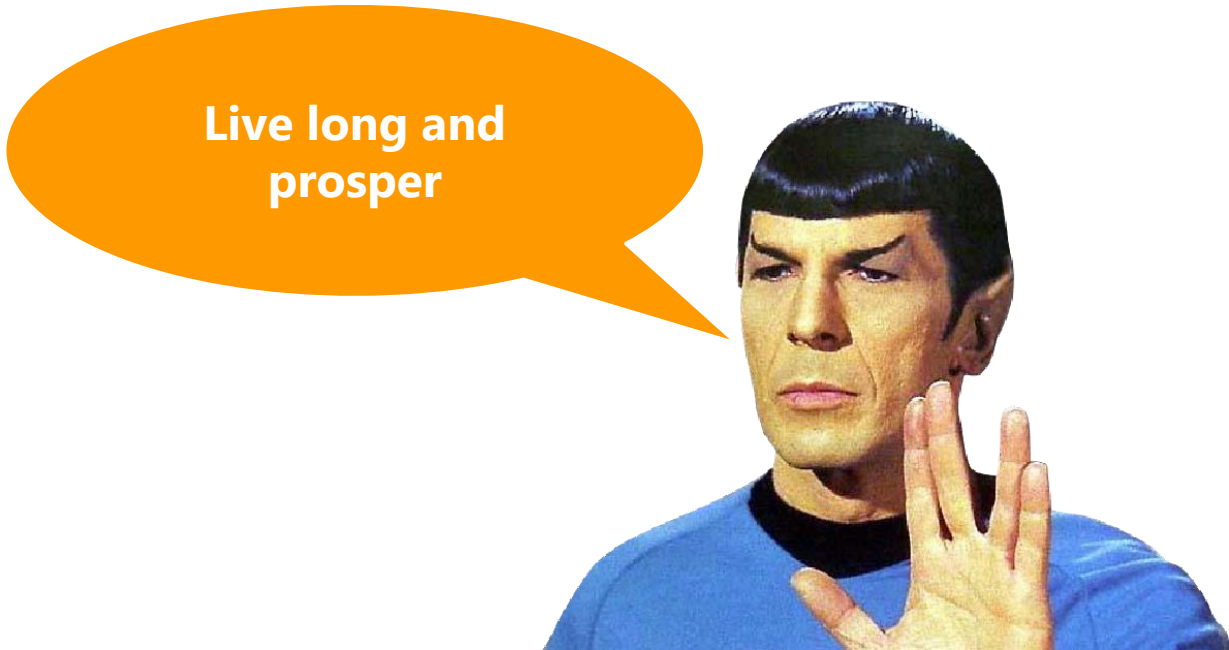
Programming paradigm

- ➔ In the **imperative paradigm** the real world is modeled as steps, commands and structures
  - ◆ Structures are first-class citizens
  - ◆ C, C++, PHP

Programming paradigm

➔ In the **logic paradigm** the real world is modeled as axioms, inference rules and queries.

- Rules are first-class citizens
- Prolog...



Programming paradigm

➔ In the **object-oriented paradigm** the real world is modeled as objects.

- Objects are first-class citizens
- Java, C++, C#, SmallTalk
- Why has the OO paradigm being so successful?

What is OOP?

Object-Oriented Programming

- ➔ Design and build programs under the object-oriented paradigm
- ➔ Models software in terms similar to those that people use to describe real-world objects
- ➔ Defining the **attributes** and **behavior** of a set of objects interacting between each others in order to solve a specific problem

Object-Oriented Programming

➔ There are some important concepts to understand OOP:

- Abstraction
- Encapsulation
- Classes
- Attribute
- Method
- Instance
- Inheritance
- Polymorphism

What is Abstraction?

The first thing you might think of is art...


Kixhome Oil Painting

Why is called abstract art?

Abstraction

- ➔ Abstraction is a non-accurate representation of the reality
- ➔ This means that the artist draw that he/she thinks of the reality without being close to the real thing.
- ➔ For example: **draw a house**

What is that?

Is that a house? Or a representation of a house?

Where are the electric wires, the pipes, and all the necessary things that a house needs?

Abstraction

- ➔ Abstraction is the capacity of the humans to create representations of the reality
- ➔ These representations omit details to allow us to handle the complexity of the problems
- ➔ We need to exclude non-relevant details in order to focus on what is really important

Object-Oriented Programming

Abstraction

➔ How does this applies to OOP?

➔ In OOP, the abstraction allows us to define objects that represent **abstract concepts**, perform a specific task, change its state and communicate with other objects

➔ Coding in OOP is like creating an abstract painting

Object-Oriented Programming

Encapsulation

- An OOP wraps attributes and behavior (operations, methods) into objects
- A single entity contains both attributes and behavior
- Attributes and objects can be thought as a single block of memory

Encapsulation

➔ Objects have also the property of **information hiding**

➔ Objects hide the implementation details from other objects

➔ Interaction between objects is done through a well defined interface

Let’s think about encapsulation when you use a microwave oven

What is a class?

Object-Oriented Programming

Classes

- A class is like a mold, for example a cupcake mold
- The mold defines how the resulting cupcake will look like, but does not define the cupcake itself
- With the same mold, you can bake many types of cupcakes

Classes

➔ You cannot eat the mold

➔ The mold will only help you create new cupcakes that you can eat

Object-Oriented Programming

Classes

- ➔ In OOP classes define the attributes and methods (behavior) that an object will have
- ➔ A class is a blueprint to **instantiate** new objects based on it
- ➔ So, how can we create a new class?

Classes

- You'll need to define the attributes and methods that the objects based on that class will have

- An **attribute** is a data element of the class. For example, a Car class has attributes such as Top Speed or Max Fuel Capacity.

- Both of these are integers and can hold exactly one value at any given moment

Classes

- ➔ Think of an attribute as a variable that is linked to the an object. For example, we can have two cars with different top speed
- ➔ Both are cars, but their attribute TopSpeed is different for each of them.
- ➔ All humans have different physical attributes, but we all are human beings...

Classes

- **Methods** are operations that make changes to the attributes of the objects
- Methods define the interface of an object, for example, the pedals of a car is part of its interface
- Thankfully you never have to interact with the engine to make the car go faster or slow down

Object-Oriented Programming

Classes

➔ The car hides the complexity of the engine, and you only interact with its interface

➔ Methods are **messages** sent to an specific object

Instance

- ➔ After you have defined the “mold” now comes the time to bake the cupcakes
- ➔ Are you going to do chocolate cupcakes or vanilla?
- ➔ **Instantiating** is creating an object using the mold or the class defined before

Instance

- ➔ After you have defined the “mold” now comes the time to bake the cupcakes
- ➔ Are you going to do chocolate cupcakes or vanilla?
- ➔ **Instantiating** is creating an object using the mold (class) defined before

Instance

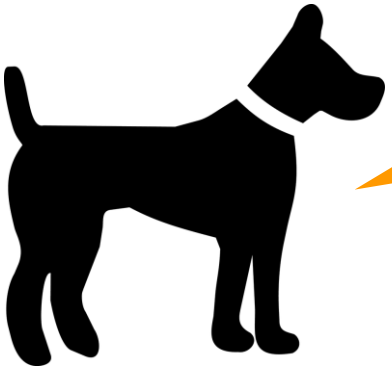
- The term instance refers to an object created using a class

- Not all instances are the same just like a chocolate cupcake is the same as a vanilla cupcake but both can be considered as cupcakes

Instance

- ➔ Each instance is a separate entity
- ➔ An change on an instance does not affect other instances of the same class
- ➔ Methods are executed on an specific instance

Attributes:
Name, Age, Weight,
Aggressiveness



Methods:
Bark, Greet, Roll,
Attack

Dog class

Spike, 1, 2 kg,
Middle

Spot, 2, 12 kg,
Low

Devil, 4, 20 kg,
Very High

Time for a quick review…

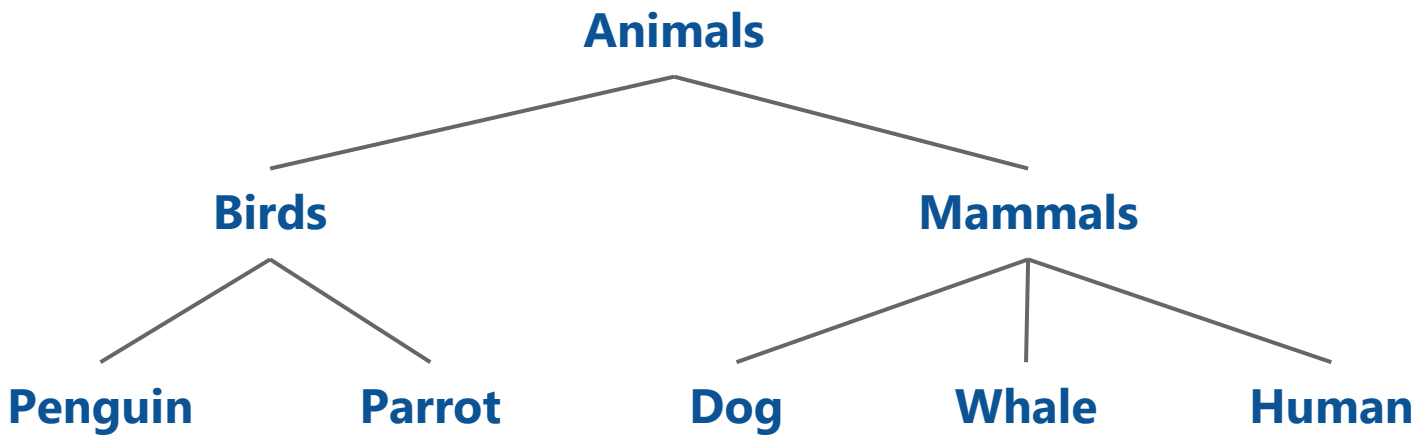
➔ Are we clear on:

- Abstraction
- Encapsulation
- Classes
- Attribute
- Method
- Instance

Object-Oriented Programming

Inheritance

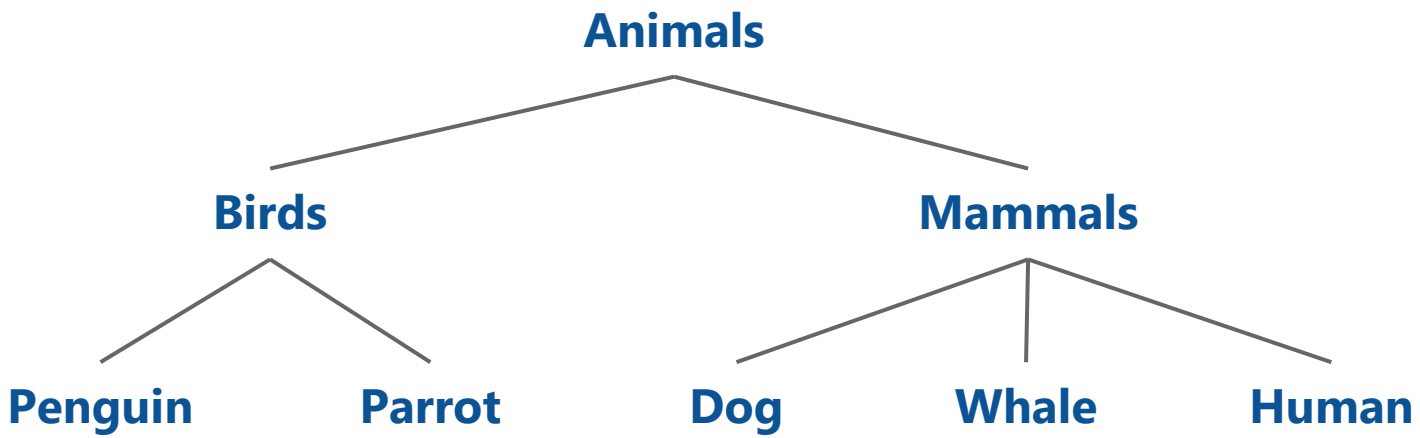
➔ In real life objects form hierarchies. For example:



Object-Oriented Programming

Inheritance

➔ Each animal family share attributes and behavior



Inheritance

- In OOP classes can also share attributes and methods using inheritance
- This means that a class can be related with other class in a parent-child relationship
- Parent class is called base class or superclass.

Inheritance

- The child class is called derived class or subclass
- Subclasses inherit attributes and methods of its parent, but it also can add new ones
- Let's go back to the animal hierarchy...

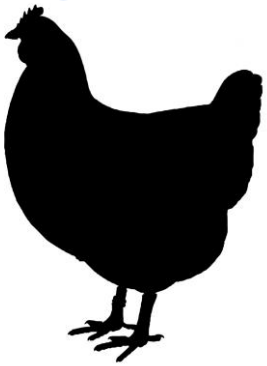
Inheritance

- The child class is called derived class or subclass
- Subclasses inherit attributes and methods of its parent, but it also can add new ones
- Let's go back to the animal hierarchy...

Object-Oriented Programming

Inheritance

What attributes and behavior share chicken and eagles and with other birds?



What attributes and behavior are different between each other and with other birds?

Object-Oriented Programming

Inheritance

- ➔ Inheritance is a good way of **reusing code**. At first it may be hard for you, but eventually it will be more natural
- ➔ Use it when is necessary…the right tool for the right problem
- ➔ Inheritance = **“is a”** type of relationship

Object-Oriented Programming

Polymorphism

➔ Polymorphism is tightly related to inheritance

➔ It allows to treat different objects as they were the same

➔ A subclass can modify the methods inherited from this parent. This is called **overriding**

Polymorphism - Overriding

- For example, Bird class has the method putEgg
- All subclasses will inherit the same method with the same code. But the way a penguin puts an egg can be different of a parrot's
- So Penguin class and Parrot class can change the inherited putEgg method to **adjust it to their particular nature**

Polymorphism

- So through the use of inheritance and overriding, you can accomplish polymorphism

- For example, you may have many Penguins object and many Parrot objects. You can treat them all as Birds and call its method putEgg

Polymorphism

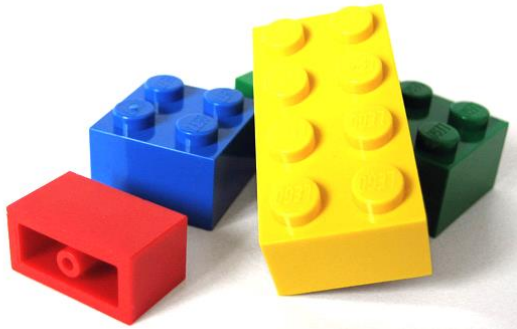
➔ Which method will be called? The one from Bird Class or the method of each of the subclasses?

Modularity

➔ Modularity allows to group logically related abstractions

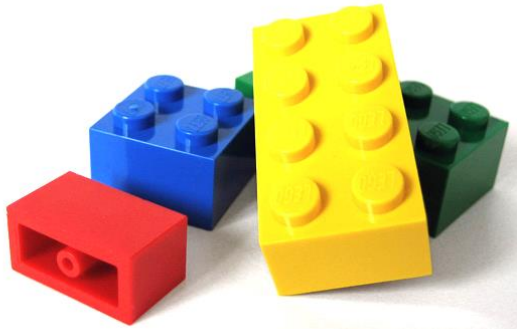
➔ Each group is a **reusable module** that can be used from different modules and classes

➔ This avoids using copy-paste of code that you need to reuse



Modularity

- ➔ Divide and conquer
- ➔ It is a good practice to pack logic in different modules
- ➔ Different languages have different terms to refer to modularity



Time for a another review…

➔ Are we clear on:

- Inheritance
- Polymorphism
- Modularity

Object-Oriented Programming

CE-1103 Algorithms and Data Structures I

Geek Corner

"Fun" facts for geeks

Today's fact:

Some Java concepts

➔ **Java**

When we talk about Java we are referring to two things: a programming language and the JVM

Java is an OO programming language initiated in 1991 by Sun Microsystems

Java is one of the most complete OO languages in the market

Geek Corner

Today's fact:

Some Java concepts

One the main features of Java is its portability. Meaning the same program can run in different platforms

➔ **JVM**

The term Java also refers to the Java Virtual Machine. What is a virtual machine? Let's discuss about it

The Virtual Machine runs on top of the hardware. That is what allows the portability of a Java program

Geek Corner

Today's fact:

Some Java concepts

Java programs don't talk to the hardware. They talk to the JVM which works as a middle man between them

➔ **ByteCode**

Java programs are not compiled. When you build a Java program instead of generating native code, it generates what is called ByteCode. This is "native code" for the JVM. The JVM translates in run time the ByteCode to machine code

Geek Corner

Today's fact:

Some Java concepts

➔ **Java API**

What is an API? The Java API is composed of a lot of packages containing useful functionalities. Each time you write an import statement, you are using the Java API

➔ **JRE**

The Java Runtine Environment is the JVM plus the class library. This is what you need to install to run Java programs. There is a JRE for each platform

Geek Corner

Today's fact:

Some Java concepts

➔ **JDK**

Java Development Kit. It is a group of tools that the developers can use to create Java programs. JDK includes the API and tools like compiler, debugger, profiler, among others.

➔ **IDE**

The Integrated Development Environment. Is the main tool to code programs. Is a tool that integrates the compiler and debugger to help programmers. Some common IDE are Eclipse, Netbeans,

Geek Corner

"Fun" facts for geeks

Today's fact:

Some Java concepts

- ➔ Java
- ➔ JVM
- ➔ ByteCode
- ➔ Java API
- ➔ JDK
- ➔ IDE