**Classes**

Python’s core data structure:

* Lists
* Tuples
* Dictionaries
* Etc.
* **Classes** are part of a programming paradigm called **object-oriented programming (OOP).**
* **OOP** focuses on building **reusable blocks of code** called **classes**.
* When you want to **use a class in one of your programs**, you make an **object from that class**, which is where the phrase "object-oriented" comes from.
* A **class** is a body of code that **defines the attributes and behaviors** required to accurately model something you need for your program. You can model something from the real world, such as a rocket ship or a guitar string, or you can model something from a virtual world such as a rocket in a game, or a set of physical laws for a game engine.
* An **attribute** is a **piece of information**. In code, an attribute is just **a variable that is part of a class**.
* A **behavior** is an **action that is defined within a class**. These are **made up of methods**, which are just **functions** that are **defined for the class.**
* An **object is a particular instance of a class**. An object has a certain set of values for all of the attributes (variables) in the class. You can have as many objects as you want for any one class.
* **Class**
* **Object**
* **Attribute**
* **Behavior**
* **Method**

Once you have a **class**, you can define an **object** and use its **methods**.

**1. Class:**

Names of classes are following the PascalCase formatting convention.

More on comments in Classes [here](https://peps.python.org/pep-0257/#handling-docstring-indentation).

A screenshot of a computer program

Description automatically generated with low confidence

**2. Object** (created from the Class):Using this example: it creates a rocket (a variable called my\_rocket). An object *Rocket* is being created from the class.



**1.2 Method 1 & 2** (within the Class):

A method is a function that is part of a class. It sets the values for any parameters that need to be defined when an **object** is first created.

Function names that start and end with two underscores are special built-in functions. The \_\_init\_\_() is called automatically when you create an object from your class.

All methods in a class need the self object as their first argument, so they can access any attribute that is part of the class.

In this case the method 1 initializes the x and y values of the Rocket to 0.

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer program

Description automatically generated with low confidence

* Access **object’s** **variables** or **methods** by using **object’s name** and **value name after self.**: my\_rocket.x or my\_rocket.y
* Use a **method** on an **object** you write **object’s name** and **method’s name**: my\_rocket.move\_up()