Purpose

This game provides an analogy of the logic behind Object Oriented Programming (OOP). For people who are learning this process, this game may be is a useful resource to understand major concepts involved in OOP.

Rationale

What is OOP?

In **programming**, we abstract **objects** from **real life**. It means, the programmer uses real life's **logic** and turns it into a **code**.

We're surrounded by "objects" (living and non-living things) that belong to a specific general category (or class) because they share specific **attributes** and/or **behaviors**.

Example:

Object (general class)	**Attributes**
Mammals	They are warm-blooded. They have backbones. They all produce milk to feed their young.
Car	It has wheels It has doors

This is the logic that programmers use when working on **Object Orienting Programming**. The **general class** is the category that includes objects that share specific **attributes and/or behaviors**.

Why an analogy?

The logic behind OPP may be complex for a beginner programmer. So, I decided to create an **analogy** that helps us understand this logic. The user will design a pet, and with each selection, they will see the specific concept represented and the code generated by it.

Analogies are high-order processing strategies that connect familiar and unfamiliar ideas:

"The capacity for making and learning analogies is clearly at the heart of advanced human cognitive capabilities and of creativity (. . .) Analogy simply means sameness or resemblance of two objects or processes at some level of abstraction. Analogy-making is being able to see familiar things in a different manner than usually, thus enabling some unfamiliar things to look similar to some familiar ones. This process depends on the ability to combine, split and rearrange existing concepts, and on the ability to reason about relationships."

Jani, N. G., & Levine, D. S. (2000). A neural network theory of proportional analogy-making. *Neural Networks, 13*(2), 149-183.

Connection Analogy – OOP

This are the major concepts and analogies represented in the game.

Concept From OOP	**Definition**	**Analogy Game Connection**		
Super class	General category that includes all objects selected	Pet shape	Animal	Features
Sub class	The attributes that define a class.		Dog Cat Goose	
Over loading	When you have the option to add more behaviors into the class.	My Family	1.By default: the pet itself. 2. (open for adding any option)	Behavior
Over writing	When the method has an attribute that can be erased and changed for a new one.	Speak	1.By default: bark, meow, (it can be replaced) 2.We can add a new way to speak	

Instructions

The user will create his own pet by following these steps:

Step order	Concept From OOP	Action	Section
1	Super Class	[Click on] **Begin shape mode** See Code at the bottom	**-**
2	Attribute	**Pet** [select one] Dog Goose Cat See Code at the bottom	
3	Attribute	**Name** [Enter your selection] See Code at the bottom	**Choose your pet**
4	Attribute	**Eyes color** [Select from palette color] See Code at the bottom	
5	Attribute	**Hair Color** [Select from palette color] See Code at the bottom	
6	Attribute	**Size** [select one] Small Medium Large	

		See Code at the bottom	
7	**-**	[Click on] **Create pet** See Code at the bottom	**-**
8	**-**	[Click on] **Default Features** See Code at the bottom	**-**
7	Over loading	**My Family** [Enter your selection after default option "itself"] See Code at the bottom	**Features**
8	Over writing	**How I move** [Enter your selection, erase default option if prefer] See Code at the bottom	
9	Over writing	**My language** [Enter your selection, erase default option if prefer] See Code at the bottom	
10	**-**	[Click on] **Save Features** See Code at the bottom	

Play Here: https://anderama100.github.io/DupeMyPet/