NAME		GRADE	
		1ST - DAM / DAW	
SURNAMES		SUBJECT	ANNOUNCEMENT
		PROGRAMMING	2ND EVALUATION
DNI	DATE	GRADE	
	28/02/2023		

1. **(4 POINTS)**
   a. **(0.50p)** Implement an **enum class** called **Color** having following values: black, white, red, orange, brown.
   b. **(1, 50p)** Implement an **abstract class** called **Pet** having following elements:
      i. attributes: name (String), owner (String), age (int), alive (boolean) and colors (a set of Color)
      ii. constructor with all fields except alive which is always true.
      iii. an abstract method called speak having no parameters and not returning any value.
      iv. two methods: anniversary and die (both with no parameters and returning nothing) knowing that anniversary means increment the age by 1 and die means set alive to false.
      v. Implement getters for all the attributes
      vi. Implement toString method
      vii. Implement equals method knowing that a pet is equals to another pet if names, owners and ages are equals.
   c. **(0,5p)** Implement an **abstract class** called **Bird** which inherits from Pet having:
      i. An attribute called fly (boolean)
      ii. Constructor
      iii. Abstract methods fly and repose (both with no parameters and returning nothing)
      iv. Setters and getters for the attribute.
   d. **(0.75p)** Implement a **class** called **Canary** which inherits from Bird having:
      i. An attribute called singing (boolean)
      ii. Constructor
      iii. Implementing method speak (showing by console "PIO PIO"), method greets (showing by console "Hello, how are you?"), fly (setting fly to true) and repose (setting fly to false).
      iv. Setters and getters for the attribute.
   e. **(0.75p)** Implement a **class** called **Cat** which inherits from Pet having:
      i. An attribute called longHair (boolean)
      ii. Constructor
      iii. Implementing method speak (showing by console "MIAU MIAU").
      iv. Getter for the attribute.

NAME & SURNAMES

2. **(0,5 POINT)** In the future we will store animals in a database, but we do not know which one is going to be used, or if we are going to develop it by ourselves. So, you have to create an interface called **IPetRepository** with the following methods:
    a. Add an animal → void add(Pet pet)
    b. Get an animal → Pet get(int index)
    c. Remove an animal → Pet remove(int index)
    d. Get all animals → Set<Pet> getAll()

3. **(2 POINTS)** In order to simulate the behavior of the database, create a class allowing the animal storage. In order to do it, you have to:
    a. Create a dynamic structure with generic types (that is, a list with generic types). Hint: just create all operations you need to manipulate the previous interface.
    b. Create a class MyModelTAD implementing the IPetRepository interface and having as attribute a generic list defined in the previous point.

4. **(2 POINTS)** Given the great utility of the collections, we would like to implement another structure simulating the behavior of a future database. In order to do that, you have to:
    a. Create a class having a set of animals as attribute.
    b. This class will implement the IPetRepository interface as well as any other interfaces you think are needed.
    c. Override all methods you consider necessary for the correct performance of the class.
5. **(1.5 POINTS)** We want to get a series of lists, so you must create a method to print the following information on the screen.
    a. **(0.50p)** List of all animals sorted alphabetically.
    b. **(0.50p)** List of all alive animals sorted by age.
    c. **(0.50p)** List of all white cats.