Apu Animal Park

1 Objectives

In this assignment, you are to use three principles of OOP, encapsulation, inheritance as well as dynamic binding as a little step towards the huge world of polymorphism.

While aggregation (or composition) between objects is based on a "has-a" relation, object inheritance helps to reuse code by creating an ”is a” relation between objects of same type. In this assignment, we will be using both of these basic concepts, leaving polymorphism for the next module

Notes:

- This assignment will be used again in the next module, so make sure to do a good job and keep your files for reuse. Don't forget to comment your code well, and save your work quite often as your are going on writing code.

- This assignment is intended to be done using Windows Forms and controls. However, you may try Windows Presentation Foundation (WPF, which will work with in a later module) if you are an experienced Windows Forms programmer. It is also allowed to create a MAUI project, if you would like to test the platform.

- You don't have to follow the instructions given here step by step. You may change the GUI, and implement your own solution, provided that you meet the requirements, keep a good programming level, and follow the OOP rules.

2 Description

Apu is working in the city zoo where all types of animals are welcome to have a home. He wishes that you write a program for him for managing the registration of the animals hosted in the Park. Animals are categorized according to their main species classification as Mammals, Birds, and Reptiles. The categories along with some species are outlined below.

2.1 Category:

- **Bird**, examples dove, eagle, pelican
- **Insect**, as butterfly, bee, ant
- **Mammal,** as dog, wolf, deer, killer whales
- **Fish**, for example goldfish, shark
- **Reptile,** as frog, lizard, snake

2.2 Gender: Animals can be

- **Male,**
- **Female,**
- **Unknown.**

Each category has some characteristics, which are common for the animal objects down the hierarchy. Number of teeth, color of hair or fur for mammals, and for birds flying speed and length of wings could be examples of such characteristics. Similarly, each type of species may have its own specific characteristics (for example breed for Dog objects).

Our development plan is to divide the work in different phases (different assignments) and in each phase produce a new version. In this version, Version 1, we try to identify all the classes we might need to write and then determine the proper associations (aggregation – "has a" and inheritance – "is a") between the classes.

Our goals are to:

a. Convert the problem into as many classes as possible where each class to have its own area of responsibility.

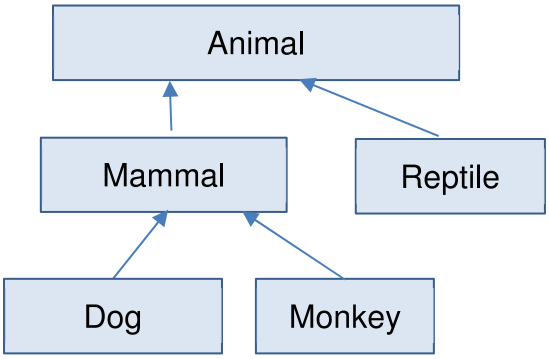
b. Assign each task in a class to a separate method.

c. Document your code by writing comments explaining briefly each class, each method and inside code wherever a line of code needs more clarification. Assume always that your code will be checked by someone else, and it should not be a problem to follow and understand your thoughts.

3 Specifications and Requirements for a Pass grade (C)

3.1 The application must have a graphical user interface (GUI). It should be possible to create an object of a specific animal, for instance, an object of Dog or Monkey. The GUI should be adjusted so all necessary input data can be saved in the object. When a certain category is chosen by the user, the GUI should show which input is to be read for the selected category, in addition to the general input.

3.2 The GUI should also show which input the chosen animal is to be given.



3.3 **Classes**: Design and implement a class hierarchy using inheritance, with an **Animal** class as the base class. Moreover, your application should include classes for the following classes.

- At least **two** categories (e.g. Mammal and Reptiles, or Mammal and Bird)
- Each category should include at least **two** animals (e.g. Dog and Monkey for Mammal).

3.4 **Attributes**: Determine attributes (fields, instance variables) for the classes:

- At least 5 attributes for the Animal class:

- **id**: to be determined by the application (use any value,"0000", etc. in this version.
- **name**, **age**, **gender** and one more (pick up yourself), user input

- At least 1 attribute for each animal, user input.

The attributes should be types that are unique for the object. For instances, the **Mammal** class should have attributes that are common for all mammal animals. The **Dog** class should have attributes that are specific for dogs, not for other animals.

3.5 When the user presses the button Add (next figure), an object of the selected animal (e.g. Dog, Butterfly) is to be created in the application. The user input should fetched from the GUI controls and saved into the object, at the three levels:

- Name, ID, age and other common values are stored in the class Animal. The ID can an empty string or any default value like "M001" (Mammals) or "B001" (Birds), in this version.
- Category-related values (e.g. number of teeth for mammals) are stored in the related base class (Mammal, Bird, etc).
- The values related to the selected animal (breed) are saved in the concrete object (Dog)

3.6 The output from this part is to display the values saved for the selected animal as the run example suggests.

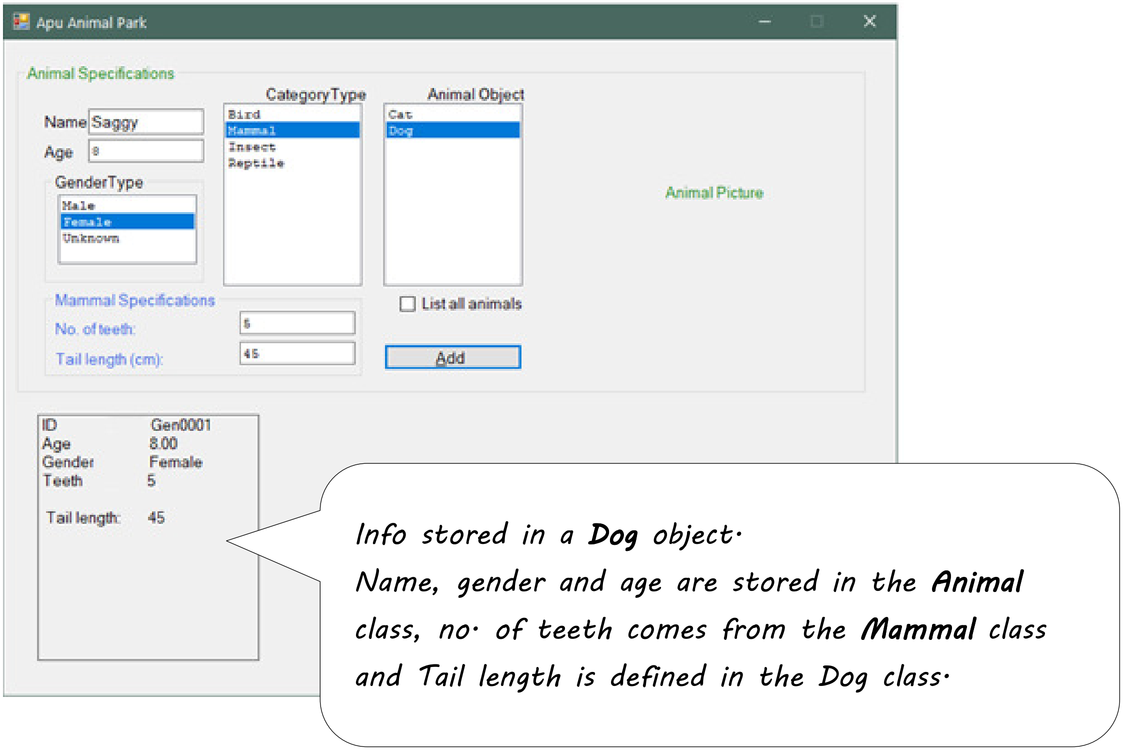
3.7 Use encapsulation, inheritance and dynamic binding to make your application work. Dynamic binding is an important goal of this assignment so make sure you understand how it is implemented for a class hierarchy.

3.7.1 Declare an object (reference variable) of Animal in the MainForm and then let the reference variable hold the object of the animal that you are creating.

```
private Animal animal = null;
...
animal = new Dog(...
```

3.8 No input/output operation is to be included in any class other than the GUI class (**MainForm**). It is this class, which only and fully is responsible for all interactions with the user. No other logics should be placed in this class.

3.9 The program should not crash for any input given by the user.



Apu Animal Park

Animal Specifications

Name Saggy

Age 8

GenderType: Male, **Female**, Unknown

CategoryType: Bird, **Mammal**, Insect, Reptile

Animal Object: Cat, **Dog**

Mammal Specifications

No. of teeth: 5

Tail length (cm): 45

☐ List all animals

Add

Animal Picture

ID Gen0001
Age 8.00
Gender Female
Teeth 5

Tail length: 45

Info stored in a **Dog** object.
Name, gender and age are stored in the **Animal** class, no. of teeth comes from the **Mammal** class and Tail length is defined in the Dog class.

3.10 The project should be well organized into subfolders in which you save a group of related classes.

4 Specifications and Requirements for a higher grade (A, B)

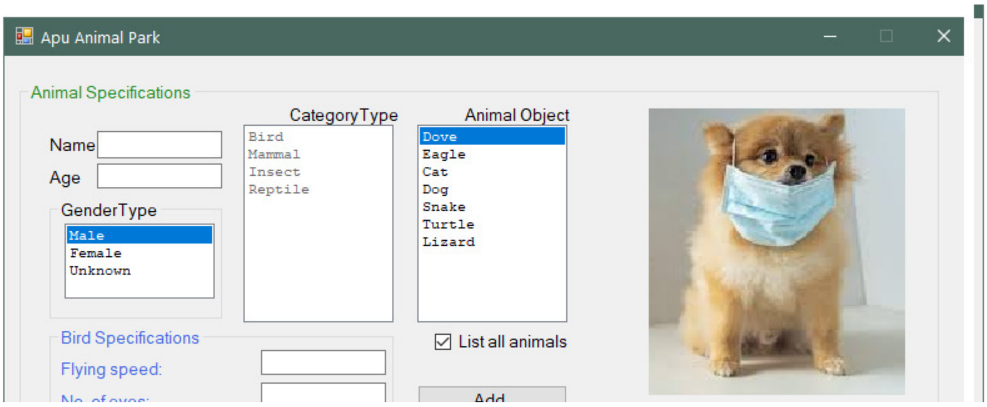
In addition to the above requirements, the following items are also to be implemented:

For Grade B:

4.1 When the user checks the option "List all animals", all animals are to be listed and categories are to be grayed. -When unchecking this option, the GUI should go back to normal inputting status.

For Grade A:

In addition to above for Grade B, the user should be given the possibility of loading an image for the selected animal. However, it is not necessary in this version to save the image in the program. Let the user browse to an image file and load the image.



Apu Animal Park

Animal Specifications

Name

Age

GenderType: **Male**, Female, Unknown

CategoryType: Bird, Mammal, Insect, Reptile

Animal Object: **Dove**, Eagle, Cat, Dog, Snake, Turtle, Lizard

☑ List all animals

Bird Specifications

Flying speed:

No. of eyes:

Add

5 Grading and submission

The assignment is submitted through Canvas. Compress all the files, folders and subfolders into a **zip** or **rar** file, and then upload it via the Assignment page Make sure that you submit the correct version of your project and that you have compiled and tested your project before handing in. Be careful not to use any hard-coded file paths (for example path to an image file on your C-drive) in your source code. It will not work on other computers. Projects that do not compile and run correctly will be directly returned for completion and resubmission.

Good Luck!

Farid Naisan,

Course Responsible and Instructor