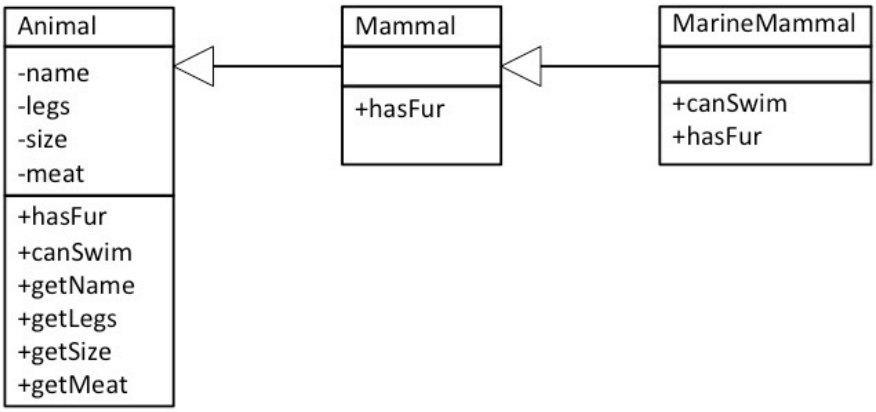
Object Oriented Programming In Python

Answers for the Activity Sheet. Week 4 The 20Q Example

1 Task 1: Review the Provided Code

1.1 Classes and Class Diagram

The classes are Animal, Mammal, and MarineMammal. The class diagram is:



1.2 Overriding and Inheritance

Method	Classes		
	Animal	Mammal	MarineMammal
hasFur	Implemented	Overridden	Overridden
canSwim	Implemented	Inherited	Overridden
getSize	Implemented	Inherited	Inherited
getLegs	Implemented	Inherited	Inherited

1.3 Comparing Legs and Swimming

The different Animals must all differ. There are two ways that are used to do this in the code:

- *The animals belong to different subclasses of Animal.*
- *The class constructors have parameters used to initialise the attributes of the class.*

Explain the way that:

i. The function canSwim() returns a different value for a Tiger and a Dolphin

ii. The function getSize() returns a different value for a Tiger and Badger.

The method canSwim() returns a different value in Tiger and Dolphin because Tiger is a Mammal wheras Dolphin is a MarineMammal. The canSwim() method is overridden in the MarineMammal class to return true.

The method getSize() accesses the value of the size attribute which is set in the constructor of the Animal (and all its subclasses). The getSize() method is not overridden: the implementation in the Animal class is inherited by all the subclasses. Instead, the two Mammals have different size values given in the call to the constructor.

2 Task 2: getDescription Function

When the game ends, it would be good to print out a full description of the secret object. For example, it might print:

```
The Tiger has the following characteristics:
  It cannot swim; it does not have fur
  It has 4 legs; it is Large; it eats meat
```

Implement this feature in the following steps:

2.1 Step 1: Which Class

The mthod is implmentd in th nimal calss and inhritd elsewhre.

2.2 Step 2: Implementation

```
def getDescription(self):
  s = "The " + self.name + " has the following characteristics:\n"
  s = s + ("  It can swim" if self.canSwim() else "  It cannot swim")
  s = s + ("; it has" if self.hasFur() else "; it does not have") + " fur\n"
  s = s + "  It has " + str(self.legs) + " legs"
  s = s + "; it is " + self.size
  s = s + ("; it eats meat" if self.meat else "; does not eat meat")
  return s
```

Note that the version of the canSwim() and hasFur() methods used depends on the (sub-) class of object (i.e. animal) on which the method is called. For example, suppose the animal is a Dolphin. Then

- getDescription is inherited from the Animal class. It calls:
- hasFur and canSwim, both overridden in the MarineMammal class.

The solution uses the _ if _ else _ expression: it can easier be written without this but it is more concise.

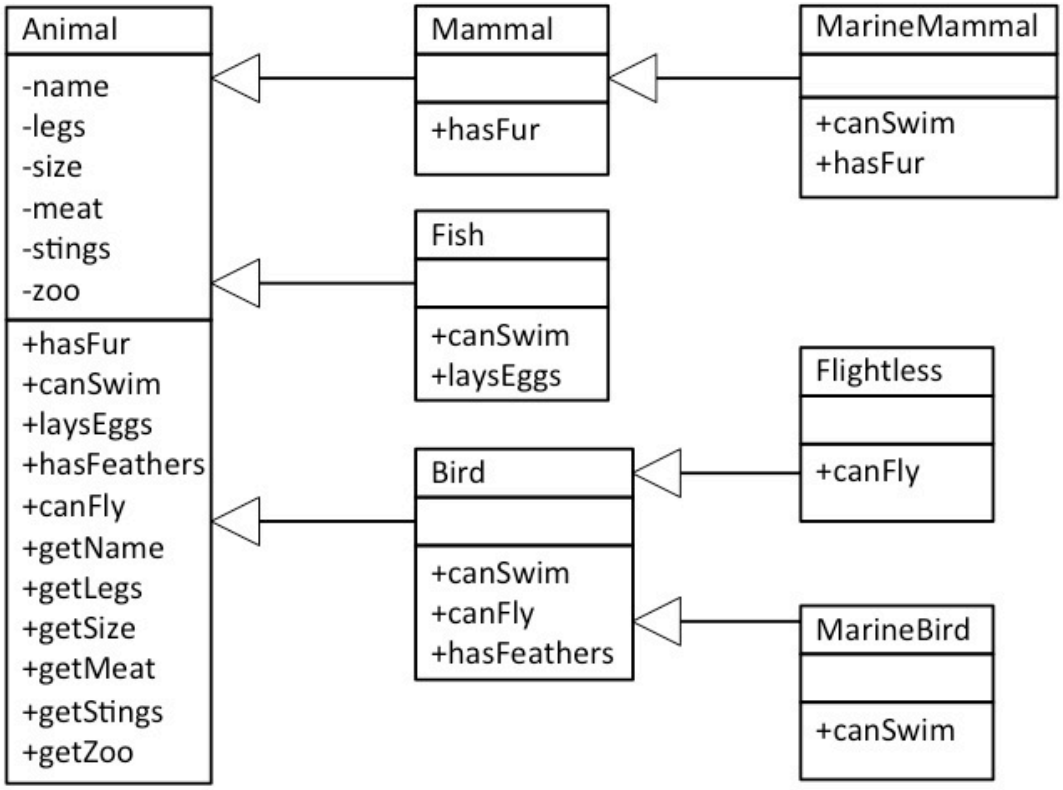
3 Task 3: Adding More Animals (or other objects)

3.1 Step 1: List the new Animals

The following shows some possible classes and animals. Some of these are used in the example solution.

Bird: Duck, Eagle, Emu, Penguin CreepyCrawly: Grasshopper, Wasp, Spider, … Fish: Shark, Minnow, Trout Reptile: Snake, Newt, Alligator

The class diagram of the example solution is:



4 Task 4 (Advanced): More Objects

Class/File	Description	Imports
Animal.py	The Animal class	n/a
Mammal.py	The Mammal class	Animal
MarineMammal.py	The MarineMammal class	Mammal
Question	The Question class: a single question.	n/a
Topic	The Topic class: a list of things and questions to make up a 20Q topic	Question
AnimalTopic	The AnimalTopic class: the animals and questions for animals	Topic, Animal, Mammal, MaineMammal
20Q.py	The main program. The program is run from here.	AnimalTopic