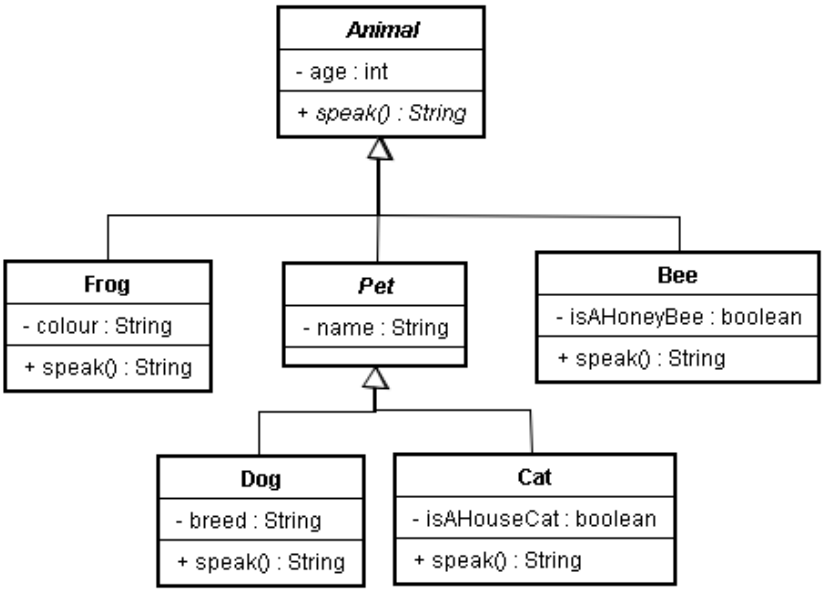
Programming Exercises - PRO1 - Session 21

Exercise 21.01

Based on the example I showed in the slides, add a `toString` method and an `equals` method to each of the 3 classes (`TwoDimensionalShape`, `Circle` and `Rectangle`). Then create a test class to try out the functionality. Make sure you experiment with polymorphism, by creating some reference variables of type `TwoDimensionalShape`, and assign some `Circle` and `Rectangle` objects to them.

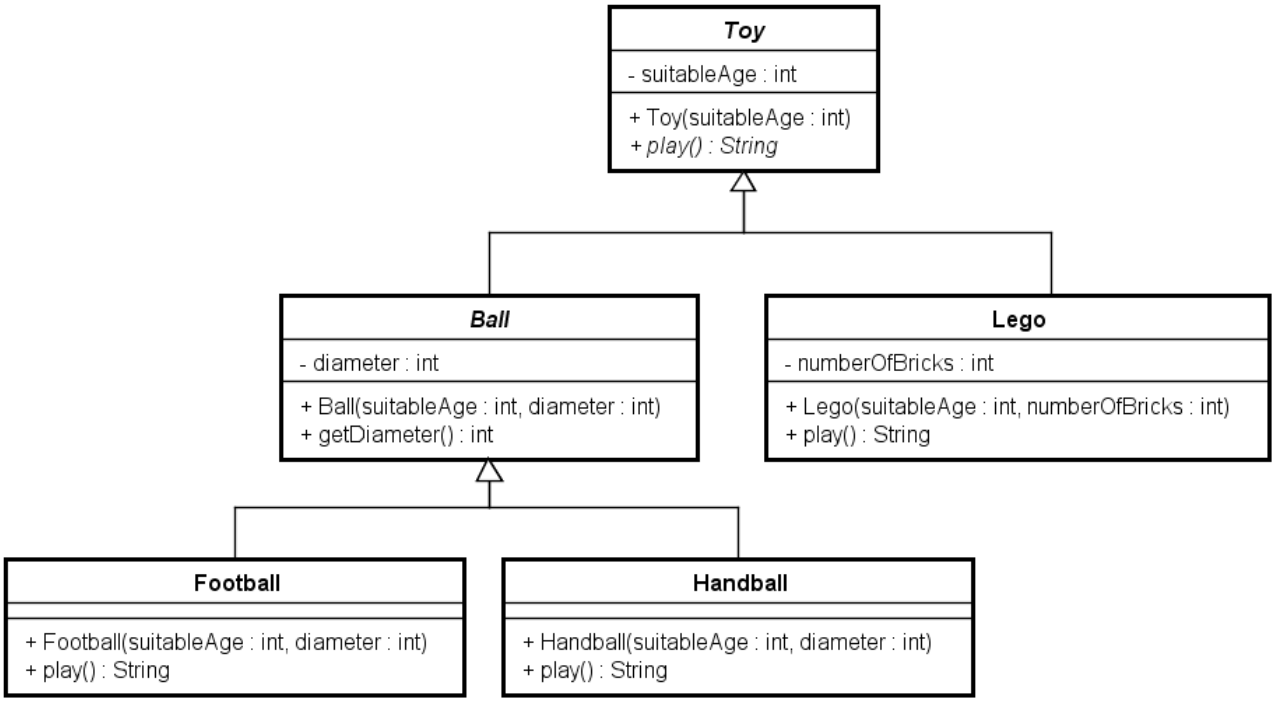
Exercise 21.02

The classes shown in the UML class diagram below represents an Animal hierarchy. Class `Animal` is abstract defining the abstract method `speak()` which will return the sound of the animal (A frog says "`Ribbit`", a bee "`Bzzz`" a dog "`Woof`" and a cat "`Meow`")



a) Implement class `Animal`, an abstract class with the abstract method `speak()`
b) Implement class `Pet`, an abstract sub class to `Animal` (not implementing the method `speak()`)
c) Implement classes `Frog`, `Bee`, `Dog` and `Cat`
d) Add the necessary constructors to all classes, even though they are not in the diagram
e) Create a test class with a main method in which you create a variable of type `Animal[]` and let each array position reference one of the sub classes to `Animal`. In a loop call `speak()` for all elements/animals
f) Add get and set methods, a `toString` method, and an `equals` method to all the classes. Then test these methods in the test class as well

Exercise 21.03



a) Implement the class hierarchy shown in the UML class diagram above, consisting of the abstract classes *`Toy`* and *`Ball`* and the classes `Lego`, `Football` and `Handball`
   - The abstract method *`play`* in class *`Toy`* should be implemented in the classes so that it returns a string indicating how to play with each toy:
     - In class `Lego`: "`Build`"
     - In class `Football`: "`Kick`"
     - In class `Handball`: "`Throw`"

b) Then implement a test class, with a main method, in which you use polymorphism by creating an array of type `Toy`, filling it with different types of toys, and running through the array calling the `play` method for each toy

c) Add `toString` methods and the relevant get and set methods to the classes. Then test these too

Exercise 21.04

[Gaddis] Algorithm Workbench 3, p. 695

Exercise 21.05

Which of the following statements regarding inheritance and classes are true?

a) The direct superclass of a subclass (specified by the keyword extends in the first line of a class declaration) is the superclass from which the subclass inherits. An indirect superclass of a subclass is two or more levels up the class hierarchy from that subclass.
b) In single inheritance, a class is derived from one direct superclass. In multiple inheritance, a class is derived from more than one direct superclass. Java does not support multiple inheritance.
c) A subclass is more specific than its superclass and represents a smaller group of objects.
d) Every object of a subclass is also an object of that class's superclass. However, a superclass object is not an object of its class's subclasses.
e) An "is-a" relationship represents inheritance. In an "is-a" relationship, an object of a subclass can also be treated as an object of its superclass.
f) A "has-a" relationship represents associations. In a "has-a" relationship, a class object should contain references to objects of other classes.
g) A subclass cannot access or inherit the private members of its superclass. Allowing this would violate the encapsulation of the superclass. A subclass can, however, inherit the non-private members of its superclass.
h) A superclass method can be overridden in a subclass to declare an appropriate implementation for the subclass.
i) A superclass's public members are accessible wherever the program has a reference to an object of that superclass or one of its subclasses.
j) A superclass's private members are accessible only within the superclass.
k) A superclass's protected members have an intermediate level of protection between public and private access. They can be accessed by members of the superclass, by members of its subclasses and by members of other classes in the same package.
l) The first task of any subclass constructor is to call its direct superclass's constructor, either explicitly or implicitly, to ensure that the instance variables inherited from the superclass are initialized properly.
m) A subclass can explicitly invoke a constructor of its superclass by using keyword super, followed by a set of parentheses containing the superclass constructor arguments.
n) When a subclass method overrides a superclass method, the superclass method can be accessed from the subclass if the superclass method name is preceded by the keyword super and a dot (.) separator.
o) Method toString takes no arguments and returns a String. The Object class's toString method is normally overridden by a subclass.
p) When an object is output using the System.out.println method or an object is added to a string using the operator +, the object's toString method is called implicitly to obtain its string representation.