SE116 – LAB#7 2021-2022 SPRING

Aim: Introduction to inheritance, superclass, subclass; understanding "is-a" relationship; the use of protected members.

TO DO @ LAB:

1. Implement an `Animal` hierarchy.

Implement the base class `Animal` to represent the animals in general. The class `Animal` will have the following `protected` data members: `String name`, `int age` and `int numberOfLegs`. Define appropriate set/get methods and non-parameterized/parameterized constructors of the class. Also, define a method named `printVoice` that takes no argument and returns no value. The method `printVoice` of class `Animal` will print out the following statement: `"In the future, the method printVoice will display the voice of a subclass of the class Animal"`. Actually, the objective of the method `printVoice` will be clear whenever it is overridden in a subclass of the class `Animal`. Additionally, define a method named `printAllData` that takes no argument and returns no value. The method `printAllData` will print out all information and will call the method `printVoice` as its last statement.

Implement one other class `Dog` as a subclass of the class `Animal`. The **"is-a" relationship** between the class `Animal` and the class `Dog` may be stated in the following manner: "A `Dog` **is an** `Animal`". A subclass inherits all the members from its superclass. The class `Dog` will have no extra data member. However, the subclass `Dog` will override the method `printVoice` (in other words, the method `printVoice` will be redefined in subclass `Dog`). In the method `printVoice`, the voice (bark sound) for a dog to be displayed will be `"WOOF!"`. Moreover, define a parameterized constructor for the subclass `Dog`. Keep in mind that the parameterized constructor of a subclass shall call the corresponding parameterized constructor of its superclass in its inheritance hierarchy.

Derive another subclass `Duck` from the superclass `Animal`. The subclass `Duck` will have no extra data member. However, the subclass `Duck` will override the method `printVoice`. In the method `printVoice`, the voice for a duck to be displayed will be `"QUACK!"`. Moreover, define a parameterized constructor for the subclass `Duck`.

Implement a class `Test` to contain the method `main`. In `main`, instantiate sample objects from each of the three classes mentioned above. Fill the information of these objects with the data to be entered by the user. Finally, call the method `printAllData` for each object.

2. Modify your project implemented in Question#**1** in the following manner:

Derive one more child class `Goat` from the parent class `Animal`. The class `Goat` contains the following extra data member: `int lengthOfBeard`. Do not forget to define extra set/get methods for this data member. Define a parameterized constructor that calls the corresponding parameterized constructor of the superclass to set the data members inherited and sets the data member `lengthOfBeard`. Moreover, the subclass `Goat` will override the method `printVoice`. In the method `printVoice`, the voice for a goat to be displayed will be `"BAAAAA!"`. Additionally, the subclass `Goat` will override the method `printAllData` since there is an extra data member to be displayed.

Modify the method `main` of class `Test` in order to use a `Goat` object as well.

TO DO @ HOME:

3. Modify your project implemented in Question#**2** in the following manner:

Suppose that you have a zoo to contain some animals. In your project, try to simulate the use of different types of animals in the zoo. In this manner, modify the method `main` of class `Test`. In `main`, use an instance of `ArrayList` to contain `Animal` references. Create 10 animals in random types and add their references into the `ArrayList`. You may choose a random number (i.e., a number from the set {1, 2, 3}) to decide to create a `Dog`, a `Duck` or a `Goat`. Then, print all information of each animal in `ArrayList`. Furthermore, print the information of the animal with the greatest age value. If there are duplicate age values for the animals in the list, the first one found may be used as the result of the search algorithm. Additionally, display the animals with 2 legs (consider also the random scenario that such animals do not exist in the list). Finally, calculate and print out the average age value of all animals in the list.

Hint: Remember that the class `SecureRandom` can be used to create random numbers.