Object Oriented Programming
Lab 14 Practice Questions

May 8, 2023

1 Lab Goal

The purpose of this lab is to test what you have learned throughout the course and apply it to various questions to check your programming prowess.

2 Problems

Read the problems carefully and come up with their solution. You must use **UMLet** to create their class diagram first before solving them.

2.1 Problem 1 - Animals are noisy

Create a class named Animal that has two instance variables: name and sound.

The class should have a constructor that takes the name and sound as parameters and assigns them to the instance variables. The class should also have a method named makeSound that prints the name and sound of the animal.

Create two subclasses of Animal: Dog and Cat, and override the makeSound method to print a different message for each subclass. Create an object of each subclass and call the makeSound method on them.

2.2 Problem 2 - Employees are at it again

Create a class named Employee that has three instance variables: name, salary, and department. The class should have a constructor that takes the name, salary, and department as parameters and assigns them to the instance variables. The class should also have a method toString that prints the name, salary, and department of the employee.

Create two subclasses of Employee: Manager and Developer, and add an instance variable named bonus to the Manager class and an instance variable named language to the Developer class. The subclasses should have constructors that call the super constructor with the appropriate parameters and also assign values to their own instance variables.

The subclasses should also override the toString method to print their own information along with the super information. Create an object of each subclass and call the toString method on them.

2.3 Problem 3 - Fly you fools!

Create an interface named Vehicle that has two abstract methods: start and stop. The start method should take no parameters and return nothing, while the stop method should take an integer parameter named distance and return a boolean value indicating whether the vehicle can stop within the given distance or not.

Create three classes that implement Vehicle: Car, Bike and Helicopter, and provide different implementations for the start and stop methods. The Car class should have an instance variable named speed, while the Bike class should have an instance variable named gear. The helicopter should have an instance variable of rotations. The classes should have constructors that take the appropriate parameters and assign them to the instance variables.

The start method should print a message indicating that the vehicle has started, while the stop method should use some logic to determine whether the vehicle can stop within the given distance or not based on its speed,

gear or rotations. Create an object of each class and call the start and stop methods on them with different distances.

There should be an Operator interface that has the operate function. Three classes Driver, Rider and Pilot should implement this interface. Every vehicle must have an operator.

There should be a transport company that has a random number of Cars, Bikes and Helicopters available. It should receive Requests that contain the number of passengers that want to make the trip. If there is only one passenger, then a bike is used. If there are 2 to 4 passengers, a car is used and if there are more than 4 passengers a helicopter is used.

Before the trip an operator is assigned to a vehicle who will then operate the vehicle. Any operator can be assigned to any vehicle but only the pilot will operate the Helicopter. Hence, the pilot implements the Flyer interface that contains the flies function.