Week 8:Inheritance

Learning Materials: Chapter 9
Topics:

- Derive class,
- Override member functions,
- Use Base class function in Overridden function,
- Which function will be called (base or derived.)?
- Type of Inheritance (public, private, protected)
- Multiple inheritance
- Multi-Level Inheritance
- Aggregation and Composition

Task 0:

Imagine you are developing software for a logistics company that manages and optimizes the process of moving goods from one location to another.

Write the class definitions required for the following scenarios using the concepts of OOP:

The company has different types of vehicles for transportation. Each vehicle has a license plate, manufacturer, and carriage size limit in kg. The vehicles can be categorized into two types based on fuel consumption: gasoline vehicle uses fossil fuel and electric vehicle uses electricity. Gasoline vehicles are distinguished by their fuel tank capacity in liters and fuel type (petrol, octane, etc.). On the other hand, electric vehicles are distinguished by their battery capacity in KWh and charging time in minutes.

The company has three types of gasoline vehicles: motorcycles, cars, and trucks. For the cars, the company keeps track of the number of passengers each of them can hold. The trucks' cargo capacity is also kept track of. Additionally, the company has some hybrid vehicles. They use both fossil fuels and electricity. This type of vehicle uses a regenerative braking system that converts the kinetic energy of a moving vehicle into

electrical energy to recharge the battery. The specific attribute of a hybrid vehicle is energy regeneration efficiency. The logistics company has many branches. Each branch maintains 10 motorcycles, 10 cars, 10 trucks, 10 electric vehicles, and 10 hybrid vehicles.

There are two types of employees present in each branch: the manager and the driver. During the recruitment, the company asks the employees to provide their names. They are also assigned an ID. Each branch has 10 managers and 30 drivers.

Upon request, each branch needs to introduce its employees. The branch introduces all the managers by printing the names with "Mr. " as the prefix. The drivers are introduced by printing their names only.

At some point, a branch may need to perform some maintenance on its vehicles. Different categories of vehicles require different types of maintenance. Trucks might need cargo checks, while cars might need passenger checks. Mention the changes you will require to achieve this constraint.

For simplicity, instead of the specific checks for each vehicle, you can print "<vehicle type> - maintenance" when maintenance is performed on any vehicle.

Task 1:

You are asked to create a base class called **Animal** and subclasses that demonstrate different inheritance types and function overriding. The structure and requirements are as follows:

Class Animal

Define the base class Animal with the following criteria:

Private Members:

- nameOfAnimal: string
- habitat_area: string
- sound: string
- weight: integer

height: integer

Public Members:

A parmeterized constructor to assign default values to all the private members.

displayInformation(): A method that prints a summary of the animal's details.

changeWeight(int _weight): A method to modify the weight of the animal.

make_sound(): Prints the sound attribute.

Class Cow (USE Public Inheritance):

Private member: milkProductionInLiters: double.

Sets the sound attribute to "moo."

make_sound(): display "The cow says: " before calling the base class make_sound() method.

Implements getMilkProduction(): Returns a value of daily milk production in liters.

Class Chicken (USE Protected Inheritance):

Private member: dailyEggCount: int.

Sets the sound attribute to "buck buck."

make_sound(): to display "The chicken says: " before calling the base class make_sound() method.

Implements getEggCount(): Returns a daily egg count.

Class Cat (Private Inheritance):

Sets the sound attribute to "meow."

make_sound() to display "The cat says: " before calling the base class make_sound() method.

Class PetCat:

Private member: petName: string.

make_sound() to include the pet’s name in the output.