CS2006 C++ Lab Exercises
Week 16

1. Create the following classes:
   (a) Employee that stores name and employee number.
   (b) Student that stores school info and degree info.
   (c) Manager that inherits Student and Employee; stores title info (e.g. vice-president) and dues.
   (d) Scientist that also inherits Student and Employee; stores number of publications
   (e) Laborer that inherits Employee

   All the above classes should have a getdata() function to get its data from the user, and a putdata() function to display its data.

   Write a main program that tests all the classes by creating instances of them, asking the user to fill in their data with detdata(), and then displaying the data with putdata().

2. Design a class named Employee. The class should keep the following information in fields:

   Employee Name
   Employee number
   Hire date

   Write one or more constructors and the appropriate accessor and mutator methods for the class.

   Next, write a class named ProductionWorker that extends the Employee class. The ProductionWorker class should have fields to hold the following information:

   Shift (an Integer)
   Hourly pay rate (a double)

   The workday is divided into two shifts:day and night. The shift field will be an integer value representing the shift that the employee works. The day shift is shift 1 and the night shift is shift 2. Write one or more constructors and the appropriate accessor and mutator methods for the class. Demonstrate the classes by writing a program that uses a ProductionWorker object.

3. In a particular factory, a team leader is an hourly paid production worker that leads a small team. In addition to hourly pay, team leaders earn a fixed monthly bonus. Team leaders are required to attend a minimum number of hours of training per year. Design a TeamLeader class that extends the ProductionWorker class you designed in the previous Lab Exercise. The TeamLeader class should have fields for the monthly bonus amount, the required number of training hours, and the number of training hours that the team leader has attended. Write one or more constructors and the appropriate accessors and mutators for the class. Demonstrate the class by writing a program that uses a TeamLeader object.

4. Suppose we are developing a program that a car dealership can use to manage its inventory of used cars. The dealership's inventory includes three types of automobiles: cars, pickup trucks, and sport-utility vehicles (SUVs). Regardless of the type, the dealership keeps the following data about each automobile:
    - Make
    - Year model
    - Mileage
    - Price

    Each type of vehicle that is kept in inventory has these general characteristics, plus its own specialized characteristics. For cars, the dealership keeps the following additional data:
    - Number of doors (2 or 4)

    For pickup trucks, the dealership keeps the following additional data:
    - Drive type (two-wheel drive or four-wheel drive)

    And for SUVs the dealership keeps the following additional data:
    - Passenger capacity

    Design relevant classes and demonstrate in a simple program.

5. **Car Instrument Simulator:**
    Design a set of classes that work together to simulate a car's fuel guage and odometer. The classes you may include are:
    The FuelGauge class: The class will simulate a fuel guage. Its responsibilities are:
    - To know the car's current amount of fuel in gallons.
    - To report the car's current amount of fuel in gallons
    - To be able to increment the amount of fuel by 1 gallon. This simulates putting fuel in the car (The car can hold a maximum of 15 gallons).
    - To be able to decrement the amount of fuel by 1 gallon, if the amount of fuel is greater than 0 gallons. This simulates burning fuel as the car runs.

    The Odometer class: This class will stimulate the car's odometer. Its responsibilities are:
    - To know the car's current mileage
    - To report the car's current mileage
    - To be able to increment the current mileage by 1 mile. The maximum mileage the odometer can store is 999, 999 miles. When this amount is exceeded, the odometer resets the current mileage to 0.
    - To be able to work with a FuelGuage object. It should decrease the FuelGuage object's current amount by 1 gallon for every 24 miles travelled. (The car's fuel economy is 24 miles per gallon).

    Demonstrate the classes by creating instances of each. Siumulate filling the car up with fuel, and then run a loop that increments the odometer until the car runs out of fuel. During each loop iteration, print the car's current mileage and amount of fuel.