# WORKSHEET #12

## Assumes: Ch1, Ch2, Ch3, Ch5, Ch6, Ch7, Ch8, Ch9, Ch10

1. Using Object Oriented methodology, design a class named **Person** and its two subclasses named **Student** and **Employee**. Make **Faculty** and **Staff** subclasses of **Employee**. A person has a name, address, phone number, and email address. A student has a class status (freshman, sophomore, junior, or senior). Define the status as constant. An employee has an office number, salary, and date hired. Use the **SimpleDate** class as defined in Chapter3 to create an object for date hired. A faculty member has office hours and a rank. A staff member has a title. Override the **toString()** method in each class to display the class name and the person’s name.

Draw the UML diagram for the classes and implement them. Write a client that creates a Person, Student, Employee, Faculty, and Staff objects and invokes their toString methods.

1. Using Object Oriented methodology, design an abstract class encapsulating a part, with two attributes: the part number and a budget cost for it. This class has two non-abstract subclasses: one encapsulating a self-manufactured part, and the other encapsulating an outsourced part. A self-manufactured part has a cost and a drawing number; it also has a method returning whether it is over budget or under budget. An outsourced part has set of suppliers, each with a price for the part. It also has a method to retrieve the lowest-cost supplier for a part and the corresponding cost.

Draw the UML diagram for the classes and implement them. Write a client that tests these classes.

1. Using Object Oriented methodology, design a class encapsulating some food; it has two attributes: description and the number of calories per serving. It also has an abstract method taking the number of servings as a parameter and returning the number of calories. This class has two non-abstract subclasses: one encapsulating a liquid food (such as a drink, for example), and the other encapsulating a fruit. A liquid food has an additional attribute: its viscosity. A fruit has an additional attribute: its season.

Draw the UML diagram for the classes and implement them. Write a client that creates an ArrayList of food items, displays the items to the user, prompts the user for the selected items and the number of servings per item, and calculates the number of calories per meal utilizing polymorphism.

1. ☺