Weekly Lab 5 – **Another Type of Employee**

Maximum Points = 10

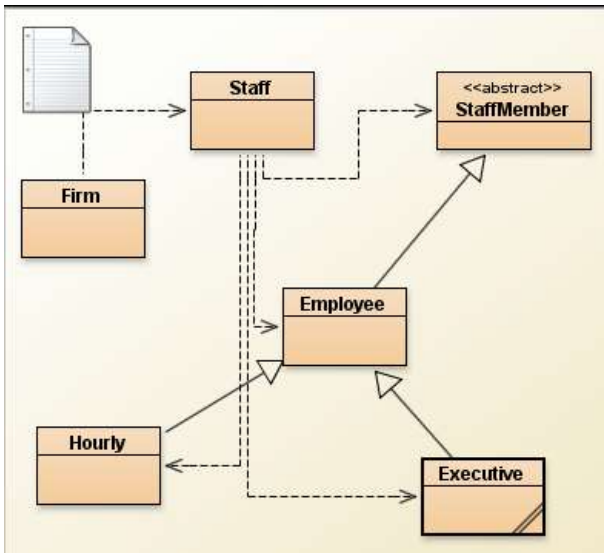
The files *Firm.java, Staff.java, StaffMember.java, Volunteer.java, Employee.java, Executive.java,* and *Hourly.java* are listed below. The program illustrates inheritance and polymorphism. In this exercise you will add one more employee type to the class hierarchy(see below). The employee will be one that is an hourly employee but also earns a commission on sales. Hence the class, which we'll name *Commission*, will be derived from the *Hourly* class.

Write a class named *Commission* with the following features:

- It extends the *Hourly* class.
- It has two instance variables (in addition to those inherited):
  - one is the total sales the employee has made (type double) and
  - the second is the commission rate for the employee (the commission rate will be type double and will represent the percent (in decimal form) commission the employee earns on sales (so .2 would mean the employee earns 20% commission on sales)).
- The constructor takes 6 parameters: the first 5 are the same as for *Hourly* (name, address, phone number, social security number, hourly pay rate) and the 6th is the commission rate for the employee. The constructor should call the constructor of the parent class with the first 5 parameters then use the 6th to set the commission rate.
- One additional new method is needed: *public void addSales (double totalSales)* that adds the parameter to the instance variable representing total sales.
- The *pay* method must call the pay method of the parent class to compute the pay for hours worked then add to that the pay from commission on sales. (See the pay method in the Executive class.) The total sales should be set back to 0 (note: you don't need to set the hoursWorked back to 0—why not?).
- The *toString* method needs to call the *toString* method of the parent class then add the total sales to that.

To test your class, update Staff.java as follows:

- Increase the size of the array to 6.
- Add two commissioned employees to the *staffList*—make up your own names, addresses, phone numbers and social security numbers.
  - Have one of the employees earn $6.25 per hour and 20% commission and
  - the other one earn $9.75 per hour and 15% commission.
- For the first additional employee you added, put the hours worked at 35 and the total sales $400;
- for the second, put the hours at 40 and the sales at $950.



```
//********************************************************************
//  Firm.java       Author: Lewis/Loftus
//
//  Demonstrates polymorphism via inheritance.
//********************************************************************

public class Firm
{
   //-----------------------------------------------------------------
   //  Creates a staff of employees for a firm and pays them.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      Staff personnel = new Staff();

      personnel.payday();
   }
}


//********************************************************************
//  Staff.java       Author: Lewis/Loftus
//
//  Represents the personnel staff of a particular business.
//********************************************************************

public class Staff
{
   private StaffMember[] staffList;

   //-----------------------------------------------------------------
   //  Constructor: Sets up the list of staff members.
   //-----------------------------------------------------------------
   public Staff ()
   {
      staffList = new StaffMember[4];

      staffList[0] = new Executive ("Sam", "123 Main Line",
         "555-0469", "123-45-6789", 2423.07);

      staffList[1] = new Employee ("Carla", "456 Off Line",
         "555-0101", "987-65-4321", 1246.15);
      staffList[2] = new Employee ("Woody", "789 Off Rocker",
         "555-0000", "010-20-3040", 1169.23);

      staffList[3] = new Hourly ("Diane", "678 Fifth Ave.",
         "555-0690", "958-47-3625", 10.55);

      ((Executive)staffList[0]).awardBonus (500.00);

      ((Hourly)staffList[3]).addHours (40);
   }

   //-----------------------------------------------------------------
   //  Pays all staff members.
   //-----------------------------------------------------------------
```

```
  public void payday ()
  {
    double amount;

    for (int count=0; count < staffList.length; count++)
    {
      System.out.println (staffList[count]);

      amount = staffList[count].pay();  // polymorphic

      if (amount == 0.0)
        System.out.println ("Thanks!");
      else
        System.out.println ("Paid: " + amount);

      System.out.println ("-----------------------------------");
    }
  }
}


//*******************************************************************
//  StaffMember.java       Author: Lewis/Loftus
//
//  Represents a generic staff member.
//*******************************************************************

abstract public class StaffMember
{
  protected String name;
  protected String address;
  protected String phone;

  //-----------------------------------------------------------------
  //  Constructor: Sets up this staff member using the specified
  //  information.
  //-----------------------------------------------------------------
  public StaffMember (String eName, String eAddress, String ePhone)
  {
    name = eName;
    address = eAddress;
    phone = ePhone;
  }

  //-----------------------------------------------------------------
  //  Returns a string including the basic employee information.
  //-----------------------------------------------------------------
  public String toString()
  {
    String result = "Name: " + name + "\n";

    result += "Address: " + address + "\n";
    result += "Phone: " + phone;

    return result;
  }
```

```
  //-----------------------------------------------------------------
  //  Derived classes must define the pay method for each type of
  //  employee.
  //-----------------------------------------------------------------
  public abstract double pay();
}

//*******************************************************************
//  Employee.java       Author: Lewis/Loftus
//
//  Represents a general paid employee.
//*******************************************************************

public class Employee extends StaffMember
{
  protected String socialSecurityNumber;
  protected double payRate;

  //-----------------------------------------------------------------
  //  Constructor: Sets up this employee with the specified
  //  information.
  //-----------------------------------------------------------------
  public Employee (String eName, String eAddress, String ePhone,
                   String socSecNumber, double rate)
  {
    super (eName, eAddress, ePhone);

    socialSecurityNumber = socSecNumber;
    payRate = rate;
  }

  //-----------------------------------------------------------------
  //  Returns information about an employee as a string.
  //-----------------------------------------------------------------
  public String toString()
  {
    String result = super.toString();
    result += "\nSocial Security Number: " + socialSecurityNumber;
    return result;
  }

  //-----------------------------------------------------------------
  //  Returns the pay rate for this employee.
  //-----------------------------------------------------------------
  public double pay()
  {
    return payRate;
  }
}

//*******************************************************************
//  Hourly.java       Author: Lewis/Loftus
//
//  Represents an employee that gets paid by the hour.
//*******************************************************************
```

```
public class Hourly extends Employee
{
   private int hoursWorked;

   //-----------------------------------------------------------------
   //  Constructor: Sets up this hourly employee using the specified
   //  information.
   //-----------------------------------------------------------------
   public Hourly (String eName, String eAddress, String ePhone,
                  String socSecNumber, double rate)
   {
      super (eName, eAddress, ePhone, socSecNumber, rate);

      hoursWorked = 0;
   }

   //-----------------------------------------------------------------
   //  Adds the specified number of hours to this employee's
   //  accumulated hours.
   //-----------------------------------------------------------------
   public void addHours (int moreHours)
   {
      hoursWorked += moreHours;
   }

   //-----------------------------------------------------------------
   //  Computes and returns the pay for this hourly employee.
   //-----------------------------------------------------------------
   public double pay()
   {
      double payment = payRate * hoursWorked;
      hoursWorked = 0;
      return payment;
   }

   //-----------------------------------------------------------------
   //  Returns information about this hourly employee as a string.
   //-----------------------------------------------------------------
   public String toString()
   {
      String result = super.toString();

      result += "\nCurrent hours: " + hoursWorked;

      return result;
   }
}

//********************************************************************
//  Executive.java       Author: Lewis/Loftus
//
//  Represents an executive staff member, who can earn a bonus.
//********************************************************************

public class Executive extends Employee
```

```
{
  private double bonus;

  //-----------------------------------------------------------------
  //  Constructor: Sets up this executive with the specified
  //  information.
  //-----------------------------------------------------------------
  public Executive (String eName, String eAddress, String ePhone,
                    String socSecNumber, double rate)
  {
    super (eName, eAddress, ePhone, socSecNumber, rate);

    bonus = 0;  // bonus has yet to be awarded
  }

  //-----------------------------------------------------------------
  //  Awards the specified bonus to this executive.
  //-----------------------------------------------------------------
  public void awardBonus (double execBonus)
  {
    bonus = execBonus;
  }

  //-----------------------------------------------------------------
  //  Computes and returns the pay for an executive, which is the
  //  regular employee payment plus a one-time bonus.
  //-----------------------------------------------------------------
  public double pay()
  {
    double payment = super.pay() + bonus;

    bonus = 0;

    return payment;
  }
}
```

Compile and run the program. Make sure it is working properly.

(Due before end of the day on Friday, February 11, 2011) Submit your .java files containing your program to the dropbox in WebCT.

Grades are determined using the following scale:

- Runs correctly……:___/3
- Correct output……..……:___/2
- Design of output..……:___/1
- Design of logic……:___/2
- Standards……:___/1
- Documentation……...:___/1

Grading Rubric (Word document)