**Programming with C#**

**Lab 1 (week 1, 2)**

=====================================================================

Your C# program consists of the following classes that represent various types of employees that are employed at a company.

|  |  |
| --- | --- |
| Classes | Purpose |
| Program | This is the executable class which has the Main() method. It creates a Staff of employees and invokes the payday method to pay them all.  The program output includes information about each employee and how much each is paid. |
| Staff | The Staff class maintains an array of objects that represent individual employees of various kinds (staffList). The array is declared to hold StaffMember references and filled with objects created from several other classes. These classes are all inherit from StaffMember class.  The staffList array is filled with polymorphic references. |
| StaffMember | This abstract class represents a generic staff member. It does not represent a particular type of employee and is not intended to be instantiated.  It serves as the ancestor of all employee classes and contains information that applies to all employees. Each employee has a name, address and phone number, so variables to store these values are declared in the StaffMember class and are inherited by all subclasses. |
| Volunteer | This class represents a staff member that works as a volunteer.  A volunteer is not compensated monetarily for his or her work. |
| Employee | This class represents an employee that gets paid at a particular rate each period. |
| Executive | This class represents an executive staff member, who can earn a bonus in addition to his or her own normal pay rate. |
| Hourly | This class represents an employee that gets paid by the hour. |

The payday method of the Staff class scans through the list of employees, printing their information and invoking their pay methods to determine how much each employee should be paid. The invocation of the pay method in some of the classes described above is polymorphic because each class has its own version of the pay method.

The following output shows the execution results of the program.

Name: Sam

Address: 123 Main Line

Phone: 555-0469

Social Security Number: 123-45-6789

Paid: 2923.07

-----------------------------------

Name: Carla

Address: 456 Off Line

Phone: 555-0101

Social Security Number: 987-65-4321

Paid: 1246.15

-----------------------------------

Name: Woody

Address: 789 Off Rocker

Phone: 555-0000

Social Security Number: 010-20-3040

Current hours: 40

Paid: 422.0

-----------------------------------

Your program will be shown as below. You must find out the correct answers to be inserted in the blanks  in the program.

[Program]

public class Program {

public static void Main(String[] args) {

Staff personnel = new Staff();

 ;

Console.ReadKey();

}

}

public class Staff {

private StaffMember[] staffList;

public Staff() {

staffList = new StaffMember[3];

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 Hourly("Woody", "789 Off Rocker",

"555-0000", "010-20-3040", 10.55);

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

((  )staffList[2]).addHours(40);

}

public void payday() {

double amount;

for (int count=0; count < staffList.Count(); count++)

{

Console.WriteLine(staffList[count].display());

amount =  ; // polymorphic

if (amount == 0.0)

Console.WriteLine("Thanks!");

else

Console.WriteLine("Paid: " + amount);

Console.WriteLine("-----------------------------------");

}

}

}

abstract public class StaffMember {

private String name;

private String address;

protected String phone;

public String Name

{ get {return name;}

set {name = value;}

}

public String Address

{ get {return address;}

set {name = value;}

}

public StaffMember(String eName,

String eAddress, String ePhone) {

name = eName;

address = eAddress;

phone = ePhone;

}

 public String display() {

String result = "Name: " + name + "\n";

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

result += "Phone: " + phone;

return result;

}

public abstract double pay();

}

public class Volunteer : StaffMember {

public Volunteer(String eName, String eAddress, String ePhone)

: base (eName, eAddress, ePhone)

{

}

override public double pay() {

return 0.0;

}

}

public class Employee : StaffMember {

protected String socialSecurityNumber;

protected double payRate;

public Employee(String eName, String eAddress, String ePhone,

String socSecNumber, double rate) :

base(eName, eAddress, ePhone)

{

socialSecurityNumber = socSecNumber;

payRate = rate;

}

override public String display() {

String result =  + "\nSocial Security Number: " +

socialSecurityNumber;

return result;

}

override public double pay() {

return payRate;

}

}

public class Executive :  {

private double bonus;

public Executive(String eName, String eAddress, String ePhone,

String socSecNumber, double rate):

base(eName, eAddress, ePhone, socSecNumber, rate)

{

bonus = 0;

}

public void awardBonus(double execBonus) {

bonus = execBonus;

}

override public double pay() {

double payment = base.pay() + bonus;

bonus = 0;

return payment;

}

}

public class Hourly : Employee {

private int hoursWorked;

public Hourly(String eName, String eAddress, String ePhone,

String socSecNumber, double rate) : 

{

hoursWorked = 0;

}

public void addHours(int moreHours) {

hoursWorked += moreHours;

}

override public double pay() {

double payment = payRate \* hoursWorked;

hoursWorked = 0;

return payment;

}

override public String display() {

String result = base.display();

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

return result;

}

}