LAB #5: INHERITANCE

1. Write the missing code in the following class definitions. Write a simple client for testing.

```java
//Inheritance: Exercise #1 Class_1
public class Class_1 {
   private int x;
   private int y;
   public Class_1() {
      x = 0;
      y = 0;
   }
   public Class_1(int x1, int y1) {
      x = x1;
      y = y1;
   }
   public void print() {
      System.out.print(x + " " + y + " ");
   }
   public String toString() {
      return x + " " + y + " ";
   }
   public void set(int x1, int y1) {
      x = x1;
      y = y1;
   }
}
//Inheritance: Exercise #1 Class_2
public class Class_2 extends Class_1 {
   private int z;
   //x = 0, y = 0, z = 0
   public Class_2() {
      ...
   }
   //x = x1, y = y1, z = z1
   public Class_2(int x1, int y1, int z1) {
      ...
   }
   //output x, y, z
   public void print() {
      ...
   }
   public String toString() {
      ...
   }
   //x = x1, y = y1, z = z1
   public void set(int x1, int y1, int z1) {
      ...
   }
}
```

2. Create a `Person` class. The class should contain 2 fields (both Strings called `firstName` and `lastName`) and the following methods:

- Default and alternate constructors.
- Two getters (accessors) to return the first and the last name
- A method named `setName` to set the fields to the parameters passed
- 2 methods to print:
    1. A method named `printLastFirst` (in this order, use “,” as a separator)
    2. A method called `print` (should print in order first name and last name)
- A method named `toString()`
- A method named `equals` (pass an object of the `Object` class)
- 2 methods named `copy` and `getCopy` to make a copy of a `Person` object into another `Person` object

The class `Person` should serve as the superclass (base class) for a class called `Employee`. This subclass (derived class) should contain 3 fields (`payRate, hoursWorked,` and `department`).
Continue the implementation for class `Employee` from here and complete the missing code:

```
//Class Employee: subclass of Person
public class Employee extends Person {
    private double payRate;
    private double hoursWorked;
    private String department;

    public final int HOURS = 40;
    public final double OVERTIME = 1.5;

     //default constructor
     public Employee() {
      ...
     }

     //add an alternate constructor with parameters


     public String toString() {
         //should return a String like this:
         //The wages for xxxx from the xxxx department are: $xxxxx.xx"

         ...
     }

     public void print() {
       //Should print output like this (same line):
       //The employee xxxx from the xxxx department worked xx hours
       //with a pay rate of $xxx.xx. The wages for this employee are $xxxxx.xx
       ...
     }


     public double calculatePay() {
        //Method to calculate and return the wages
        //handle both regular and overtime pay

        ...
     }
     public void setAll(String first, String last, double rate, double hours,
String dep){
         ...
     }

     public double getPayRate() {
         ...
     }


     public double getHoursWorked() {
         ...
     }


     public String getDepartment() {
         ...
     }

     public boolean equals(Object o) {
       ...
     }

     public Employee getCopy() {
       ...
     }

     public void copy(Employee e) {
       ...
     }

}
```

Write a simple client for testing. Continue the implementation for class `ClientEmployee` from here:

```
//Client program for Person/Employee
import java.util.Scanner;
public class ClientEmployee {
    public static void main(String[] arg) {
        Scanner input = new Scanner(System.in);
        String last, first, dept;
        double pay_rate;
        int hours;
        Employee prof = new Employee("John", "Doe", 25.50, 50, "COSC");
//subclass alternate constructor invoked
        Employee newEmp = new Employee(); //subclass default constructor
invoked
        ...
        ...
    }
}
```

SAMPLE OUTPUT:

```
Enter employee last name: Bond
Enter employee first name: James
Enter department: THEATRE
Enter employee pay rate: 35
Enter employee hours worked: 47
--- Record for both employees with overridden .toString from subclass ---
The wages for Doe, John from the COSC department are: $1402.50
The wages for Bond, James from the THEATRE department are: $1767.50
--- Output with calls to overridden method print from subclass ---
The employee John Doe from the COSC department worked 50.0 hours with a pay rate of
$25.50. The wages for John Doe are $1402.50
The employee James Bond from the THEATRE department worked 47.0 hours with a pay
rate of $35.00. The wages for James Bond are $1767.50
--- Output with calls to getters from the superclass ---
The wages for James Bond from the THEATRE department are $1767.50
--- Call to overridden equals/subclass for 2 Employee objects---
Couldn't find an employee with same record.
```