CSE142, Spring 2009, Inheritance-Examples Handout

1 Employees

Assume these classes have been defined:

```
public class Employee {
    public int getHours() {
        return 40;           // works 40 hours / week
    }

    public double getSalary() {
        return 40000.0;      // $40,000.00 / year
    }

    public int getVacationDays() {
        return 10;           // 2 weeks' paid vacation
    }

    public String getVacationForm() {
        return "yellow";     // use the yellow form
    }
}

public class TechnicalWriter extends Employee {
    public void writeManual(String app) {
        System.out.println("Writing a manual about: " + app);
    }
}

public class Lawyer extends Employee {
    public String getVacationForm() {
        return "pink";
    }

    public int getVacationDays() {
        return 15;           // 3 weeks vacation
    }

    public void sue() {
        System.out.println("I'll see you in court!");
    }
}

public class Marketer extends Employee {
    public double getSalary() {
        return 50000.0;      // $50,000.00 / year
    }

    public void advertise() {
        System.out.println("Act now for additional savings.");
    }
}
```

What output does this program produce?

```
public class EmployeeMain {
    public static void main(String[] args) {
        Lawyer leslie = new Lawyer();
        TechnicalWriter toby = new TechnicalWriter();
        printInfo(leslie);
        printInfo(toby);
    }

    public static void printInfo(Employee empl) {
        System.out.println("salary = " + empl.getSalary());
        System.out.println("days = " + empl.getVacationDays());
        System.out.println("form = " + empl.getVacationForm());
        System.out.println();
    }
}
```

What output does this program produce?

```
public class EmployeeMain2 {
    public static void main(String[] args) {
        Employee[] e = { new Lawyer(),   new TechnicalWriter(),
                         new Marketer(), new Lawyer() };

        for (int i = 0; i < e.length; i++) {
            System.out.println("salary: " + e[i].getSalary());
            System.out.println("v.days: " + e[i].getVacationDays());
            System.out.println();
        }
    }
}
```

2 Foo and Friends

```
public class Foo {
    public void method1() {
        System.out.println("foo 1");
    }

    public void method2() {
        System.out.println("foo 2");
    }

    public String toString() {
        return "foo";
    }
}

public class Bar extends Foo {
    public void method2() {
        System.out.println("bar 2");
    }
}

public class Baz extends Foo {
    public void method1() {
        System.out.println("baz 1");
    }

    public String toString() {
        return "baz";
    }
}

public class Mumble extends Baz {
    public void method2() {
        System.out.println("mumble 2");
    }
}
```

What output does this code fragment produce?

```
Foo[] pity = {new Baz(), new Bar(), new Mumble(), new Foo()};
for (int i = 0; i < pity.length; i++) {
   System.out.println(pity[i]);
   pity[i].method1();
   pity[i].method2();
   System.out.println();
}
```

3 AmFoods (tricky)

```
public class Lamb extends Ham {
    public void b() {
        System.out.print("Lamb b   ");
    }
}

public class Ham {
    public void a() {
        System.out.print("Ham a   ");
        b();
    }

    public void b() {
        System.out.print("Ham b   ");
    }

    public String toString() {
        return "Ham";
    }
}

public class Spam extends Yam {
    public void b() {
        System.out.print("Spam b   ");
    }
}

public class Yam extends Lamb {
    public void a() {
        System.out.print("Yam a   ");
    }

    public String toString() {
        return "Yam";
    }
}
```

What output does this code fragment produce?

```
    Ham[] food = {new Lamb(), new Ham(), new Spam(), new Yam()};
    for (int i = 0; i < food.length; i++) {
       System.out.println(food[i]);
       food[i].a();
       System.out.println();     // to end the line of output
       food[i].b();
       System.out.println();     // to end the line of output
       System.out.println();
    }
```