Model 1 Loud Toys

```
public class ToySheep {
    private int volume;

    public ToySheep() {
        this.volume = 3;
    }

    public int getVolume() {
        return volume;
    }

    public void setVolume(int volume) {
        this.volume = volume;
        makeNoise();
    }

    public void makeNoise() {
        System.out.println("Baaa");
    }
}
```

```
public class ToyRobot {
    private int chargeLevel;
    private int volume;

    public ToyRobot() {
        this.chargeLevel = 5;
        this.volume = 10;
    }

    public void recharge() {
        chargeLevel = 10;
    }

    public int getVolume() {
        return volume;
    }

    public void setVolume(int volume) {
        this.volume = volume;
        makeNoise();
    }

    public void makeNoise() {
        System.out.println("Beep Beep!");
    }
}
```

Questions (15 min) Start time:

1. Identify ***similarities*** in the code:

a) What attributes do the classes have in common?

volume

b) What methods do the classes have in common?

getVolume, setVolume, makeNoise

2. Summarize ***differences*** between the constructors and the `makeNoise` methods.

The volume is initialized to 3 in ToySheep, but it's 10 in ToyRobot. The ToySheep says "Baaa", but the ToyRobot says "Beep Beep!".

3. Design a new class named `LoudToy` that contains the code that `ToySheep` and `ToyRobots` have in common. The constructor of `LoudToy` should take `volume` as a parameter. The `makeNoise` method should have an empty body.

```
public class LoudToy {
    private int volume;

    public LoudToy(int volume) {
        this.volume = volume;
    }

    public int getVolume() {
        return volume;
    }

    public void setVolume(int volume) {
        this.volume = volume;
        makeNoise();
    }

    public void makeNoise() {
        // will be overridden in subclass
    }


}
```

4. Redesign `ToySheep` so that it extends `LoudToy`. The constructor of `ToySheep` should call the constructor of `LoudToy`. Remove the code from `ToySheep` that is no longer necessary.

```
public class ToySheep extends LoudToy {
    public ToySheep() {
        super(3);
    }

    public void makeNoise() {
        System.out.println("Baaa");
    }


}
```

5. Redesign `ToyRobot` so that it extends `LoudToy`. Remove the code from `ToyRobot` that is no longer necessary.

```
public class ToyRobot extends LoudToy {

    private int chargeLevel;

    public ToyRobot() {
        super(10);
        chargeLevel = 5;
    }

    public void recharge() {
        chargeLevel = 10;
    }

    public void makeNoise() {
        System.out.println("Beep Beep!");
    }


}
```

6. What is the output of the following examples?

a)
```
LoudToy toy1 = new LoudToy(1);
toy1.makeNoise();
```
(no output)

b)
```
LoudToy toy2 = new ToySheep();
toy2.makeNoise();
```
Baaa

c)
```
LoudToy toy3 = new ToyRobot();
toy3.makeNoise();
```
Beep Beep!

7. In the previous question, did the variable's type or the object's type determine the version of `makeNoise` that was called?

The object's type – notice that the variable type is the same in all three instances.