BooleanCollection

```
import java.util.ArrayList;
import java.util.List;

/**
* A collection of booleans that can answer individual and collective questions
* about the values in the collection.
*/
public class BooleanCollection {
   private List<Boolean> data;

   // EFFECTS: initialize the newly created collection
   public BooleanCollection() {
       data = new ArrayList<>();
   }

   // MODIFIES: this
   // EFFECTS: the item is added to the collection
   public void add(boolean item) {
       data.add(item);
   }

   // REQUIRES: index >= 0 and index < number of items in the collection
   // EFFECTS: returns the boolean at position index
   public boolean get(int index) {
       return data.get(index);
   }

   // EFFECTS: returns the number of items in the collection
   public int getNumberOfItems() {
       return data.size();
   }

   // EFFECTS: returns true if all the items in the collection are true,
   // false otherwise
   public boolean areAllTrue() {
       for (boolean item : data) {
           if (!item)
               return false;
       }
       return true;
   }

   // EFFECTS: returns the number of items in the collection that are true
   public int countTrue() {
       int count = 0;

       for (boolean item : data) {
           if (item)
               count++;
       }

       return count;
   }
}
```

Machine

```
package ca.ubc.cpsc210.machines.model;

// An interface for all machines giving their name and purpose

public interface Machine {
   String getName();
   String getPurpose();
}
```

Clock

```
package ca.ubc.cpsc210.machines.model;

// Represents a clock that displays the time in hours, minutes and seconds
public class Clock implements Machine {
  public static final int SECONDS_PER_MINUTE = 60;
  public static final int MINS_PER_HOUR = 60;
  public static final int HRS_PER_DAY = 24;

  private int hours;
  private int minutes;
  private int seconds;

  // EFFECTS: time has been set to 0:00:00
  public Clock() {
     hours = 0;
     minutes = 0;
     seconds = 0;
  }

  public int getHours() { return hours;  }

  public int getMinutes() { return minutes; }

  public int getSeconds() { return seconds; }

  // MODIFIES: this
  // EFFECTS: time on clock is advanced by 1 second
  public void tick() {
     seconds++;
     if (seconds >= SECONDS_PER_MINUTE) {
        seconds = 0;
        minutes++;

        if (minutes >= MINS_PER_HOUR) {
           minutes = 0;
           hours++;

           if (hours >= HRS_PER_DAY) {
              hours = 0;
           }
        }
     }
  }

   @Override
   public String getName() { return "Clock"; }

   @Override
   public String getPurpose() { return "Keep track of time"; }
}
```

AlarmClock

```
package ca.ubc.cpsc210.machines.model;

// represents an alarm clock
public class AlarmClock extends Clock {
   private int alarmHours;
   private int alarmMinutes;
   private boolean isEnabled;
   private boolean isSounding;

   // EFFECTS: time has been set to 00:00:00,
   // alarm time is 00:00, alarm is off and is not sounding
   public AlarmClock() {
       super();
       alarmHours = 0;
       alarmMinutes = 0;
       isEnabled = false;
       isSounding = false;
   }

   public int getAlarmHrs() {
       return alarmHours;
   }

   public int getAlarmMins() {
       return alarmMinutes;
   }

   // REQUIRES: 0 <= hrs < 24, 0 <= mins < 60; time is in 24hr format
   // MODIFIES: this
   // EFFECTS: alarm has been set to given time and alarm is on
   public void setAlarmTime(int hrs, int mins) {
       alarmMinutes = mins;
       alarmHours = hrs;
       isEnabled = true;
       adjustSounding();
   }

   // MODIFIES: this
   // EFFECTS: if isOn is true, alarm is on; otherwise alarm is off and alarm
   // is not sounding
   public void setAlarm(boolean isOn) {
       isEnabled = isOn;
       adjustSounding();
   }

   // EFFECTS: returns true if alarm is on, false otherwise
   public boolean isAlarmOn() {
       return isEnabled;
   }

   // EFFECTS: returns true if alarm is sounding, false otherwise
   public boolean isSounding() {
       return isSounding;
   }
```

```
    // MODIFIES: this
    // EFFECTS: turns sound on / off
    private void adjustSounding() {
        isSounding = isEnabled && alarmHours == getHours()
                && alarmMinutes == getMinutes();
    }

    // MODIFIES: this
    // EFFECTS: time on clock is advanced by 1 second, adjusts isSounding
    @Override
    public void tick() {
        super.tick();
        adjustSounding();
    }

    @Override
    public String getName() {
        return "Alarm clock";
    }

    @Override
    public String getPurpose() {
        return "Keep track of time and wake you up";
    }
}
```

Pedometer

```
package ca.ubc.cpsc210.machines.model;

// Represents a pedometer that counts the user’s steps and keeps track of
// distance traveled
public class Pedometer implements Machine {
   private static final double METRES_PER_KM = 1000.0;
   private int steps;
   private double stepLengthInMetres;

   // REQUIRES: length > 0
   // MODIFIES: this
   // EFFECTS: sets the user's step length
   public void setStepLength(double length) {
       stepLengthInMetres = length;
   }

   // EFFECTS: returns the distance travelled since last reset in km
   public double getDistance() {
       return steps * stepLengthInMetres / METRES_PER_KM;
   }

   // EFFECTS: returns the steps taken since last reset
   public int getSteps() {
       return steps;
   }

   // MODIFIES: this
   // EFFECTS: resets the step counter to 0
   public void reset() {
       steps = 0;
   }

   // MODIFIES: this
   // EFFECTS: records a step taken
   public void step() {
       steps++;
   }

   @Override
   public String getName() {
       return "Pedometer";
   }

   @Override
   public String getPurpose() {
       return "Count your steps";
   }
}
```

FitBit

```
package ca.ubc.cpsc210.machines.model;

// Represents a pedometer with a daily step goal and goal achievement history
public class FitBit extends Pedometer {
   private Map<LocalDate, Integer> history;
   private int stepGoal;

   // EFFECTS: fit bit has empty goal history
   public FitBit() {
       history = new HashMap<>();
   }

   @Override
   public String getName() {
       return "FitBit";
   }

   @Override
   public String getPurpose() {
       return "Count your steps towards a daily goal";
   }

   // REQUIRES: goal > 0
   // MODIFIES: this
   // EFFECTS: sets the daily step goal
   public void setStepGoal(int goal) {
       stepGoal = goal;
   }

   // EFFECTS: return the step goal
   public int getGoal() {
       return stepGoal;
   }

   // MODIFIES: this
   // EFFECTS: Record a step taken, print "Goal achieved!" on the console if the
   // user's step goal has just been met, print "Goal doubled!" on the console if
   // double the user's step goal has just been met
   @Override
   public void step() {
       super.step();
       if (getSteps() == getGoal()) {
           System.out.println("Goal achieved!");
           recordGoal();
       }
       if (getSteps() == 2 * getGoal()) {
           System.out.println("Goal doubled!");
       }
   }

   public boolean goalReached() {
       return getSteps() >= getGoal();
   }
```

```
    // EFFECTS: returns an unmodifiable history of dates on which step goal
    // was reached
    public Map<LocalDate, Integer> getHistory() {
        return Collections.unmodifiableMap(history);
    }

    // MODIFIES: this
    // EFFECTS: records that step goal was achieved on current date
    private void recordGoal() {
        history.put(getDate(), getSteps());
    }

    // EFFECTS: returns a local date (with no time and no time zone)
    private LocalDate getDate() {
        return LocalDate.now();
    }
}
```