```java
import bmod.wattagecalculator.Plugin;

/**
 * Always leave a good comment here about what the class does, it will be saved
 * when you generate documentation, and be placed there.
 *
 * Always include your name, and email address, so future onlookers know who
 * created the class for copyright reasons.
 *
 * If you put your name after an @author tag, then it will be specially
 * formatted in the documentation.
 *
 * Class names should be formatted in CamelCase, no underscores.
 *
 * @author Joseph Lewis <joehms22@gmail.com>
 */
public class Elevator extends Plugin
{
	// Declare all variables up here, outside the main method.
	// if you know a variable is not going to change, precede it with "final"
	// this makes sure you don't accidentally change it later.
	// final variables should be named in ALL_CAPS_SEPARATED_BY_UNDERSCORES
	private final double PI = 3.14;
	private final int MAX_NUMBER_OF_FLOORS = 15;
	private final double PERCENT_TAKING_ELEVATOR = .01;
	private final String ELEVATOR_DEVICE_NAME = "Elevator";


	/**
	 * Methods should be commented with what they do, but remember to always
	 * use as descriptive of a name as possible, so it takes less time to
	 * learn for newcomers to your code. Methods should always start with a
	 * lower case, and should be camelCase afterwards.
	 */
	@Override
	public void getWatts()
	{
		// Get the number of watts the elevator uses. We won't make this a final
		// up above, because the user may change the number of watts the
		// elevator uses between runs.
		double elevator_watts = m_mech.readDeviceType(ELEVATOR_DEVICE_NAME)
		                              .getWatts();


		// Each "floor" is a zone, floor "0" is ground level.
		for(int i = 1; i <= MAX_NUMBER_OF_FLOORS; i++)
		{
			String floor = getElevatorZoneForFloor(i);

			// We'll get the number of people in the zone, in the given building
			// from start time to end time.
```

```
                        int zone_pop = getZonePopulation(floor, startTime, endTime, building);

                        // For demo purposes only, we'll say 1% of people take the elevator
                        // on each floor, fancier things could be done though, i.e. higher
                        // floors take the elevator more often.
                        double watts = zone_pop * PERCENT_TAKING_ELEVATOR * elevator_watts;

                        // Here we call a method that is from the class we "extended"
                        // earlier, the documentation for that class says it will add the
                        // number of watts to the total, and
                        addWatts(watts, floor);
                }
        }


        /**
         * If you are ever going to do the same calculation more than three times,
         * or if the calculation is more than three lines and done more than once
         * create a method for it.
         *
         * The params starting with the at symbol here are fairly self explanatory.
         *
         * @param floorNumber - The floor number the elevator will stop at.
         * @return The name of the zone representing the floor.
         */
        public String getElevatorZoneForFloor(int floorNumber)
        {
                // When doing calculations, try to leave one blank space around
                // operators.
                return "Floor" + floorNumber;

                // As a bad example, the string "Floor" here is hard-coded, these hard
                // codings that aren't defined up above in the class are called "magic
                // numbers" and should be avoided, because it makes later maintenance
                // much more difficult.
        }
}


/**
 * END NOTE: The best coders try to leave "self documenting" code, meaning it
 * has few, if any, comments, except for method and class headers. This means
 * all variables should be named well, including units, if applicable i.e.
 * degreesC, all methods describe what they do, and their params are clear.
 * This is because as code gets updated, comments tend not to, so contradictions
 * are common.
 */
```