CSCA08 Exercise 8
Due: November 18, 2016. 5:00pm

Light Switch Problem

This week's exercise centres around a classic logic puzzle:

A lot of switches

```
I have a board of light switches, numbered 0,1,2,...,1023.
Each light switch can be either on or off. All switches are initially off.

Step 1:
I flip all of the switches starting at 0.
At this point, all of the light switches are on.

Step 2:
I flip every second switch, starting at 0.
At this point, lights 0,2,4,6,8,... are off.
Lights 1,3,5,7,9,... are still on.

Step 3:
I flip every third switch, starting at 0.
So I flip switches 0,3,6,9,12,...
that is, if a switch is on, I flip it to off.
If a switch is off, I flip it to on.

...

Step 1023:
I flip every 1023'rd switch, starting at 0.
So I flip 0 and 1023.

Question: At this point, which switches are on and which are off?
```

Solve the Problem

Your first task is to try to solve the puzzle using nothing but your own brain (and possibly a pen & paper). Can you figure out the pattern? (there's a way of finding the answer without actually walking through 1023 steps).

No... really... Solve the Problem

I'm sure half of you just jumped straight over the previous section to get to the "real" exercise. But really, c'mon... give it a try.

Building a switch

In a file called `ex8.py`, build a class `LightSwitch` with the following properties:

- When I create a switch, I should be able to set it's default state by inputting a string "on" or "off".

- I should be able to use a switch's `turn_on` method to turn it on.

- `turn_off` should work the same way, but in the opposite direction.

- I should also be able to call a `flip` method to flip its state (if it's on, it turns off; if it's off, it turns on).

- If I print a switch, it should print either `I am on` or `I am off` (whichever is currently true).

- If I try to perform an illegal operation on a switch (e.g., turn a switch on when it's already on), nothing should happen.

One other thing (hope you read the instructions before you started coding). The switch shouldn't hold its state in a `string`. There's a better data type that a switch can use to keep track of its state.

Building a switch board

In your `ex8.py` file, build a class `SwitchBoard` with the following properties:

- When I create a switchboard, I should be able to set the number of switches it contains.

- All switches should start in the `"off"` position.

- If I print a switchboard, it should print something along the lines of: `"The following switches are on:  0 2 4 6 8"`.

- The `which_switch` method should return a list of integers representing the switches that are on, in order(e.g., `[1,3,5,7,9]`).

- If I call `flip(n)` with `n` as an integer, it should flip the state of the n'th lightswitch.

- If I call `flip_every(n)` with `n` as an integer, it should flip the state of every n'th lightswitch, starting at 0. So `flip_every(2)` would flip switches 0, 2, 4, 6, etc.

- The method `reset()`, should turn all switches off.

- If I ask the switchboard to flip a switch which doesn't exist, nothing should happen (it shouldn't crash).

You can add any extra methods you need to make the code work as described above.

Now check your solution

Now, write a piece of global code that solves the problem as stated above. Your code should not run when the automarker imports your file. Did you get it right? Can you see the pattern now? Do you understand why this pattern exists?