Exercise 3

Building a switch

In a file called `ex3.py`, build a class `LightSwitch` with the following properties:

- When I create a switch, I should be able to set its 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 `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 on a switch when it's already on), nothing should happen.

One other thing (hopefully this enforces you to read instructions first before you start 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 `ex3.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.