CS 1110:
Introduction to Computing Using Python

Lecture 24

An Extended Example That Reviews Much of CS1110

[Andersen, Gries, Lee, Marschner, Van Loan, White]

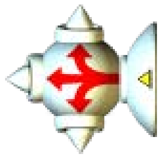
Announcements

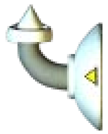
- Final Exam:
  - May 18th, 9am-11:30am
  - **Location**: Barton Hall Central and East
- A5 is out; due midnight Wednesday 5/10
  - some announcements went out by email
  - check for important updates
- Today is the final lecture – Tuesday's class will be Professor Lee open office hours
- Labs next week are TA office hours

Goals for today

- Discuss a *real-world* engineering challenge
  (that is particularly meaningful to me)
- Break down this large challenge into smaller components
- Convince you that we have learned enough Python to build these components
- Utilize many different parts of CS1110
- Try to review as much as possible

Objects: How to organize?

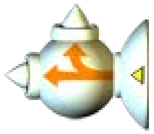

Splitter
(1 input, 3 outputs)


Bender
(1 input, 1 output)

Asteroid
(0 inputs, 0 outputs)

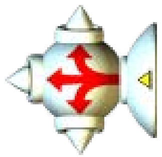
Key consideration: what attributes are ***shared***?

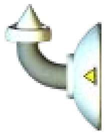
Laser
(0 inputs, 1 output)


Splitter
(1 input, 2 outputs)

Target
(1 input, 2 outputs)

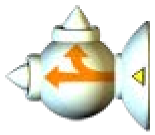
What attributes are shared?


Splitter
(1 input, 3 outputs)


Bender
(1 input, 1 output)

Asteroid
(0 inputs, 0 outputs)

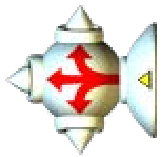
Laser
(0 inputs, 1 output)

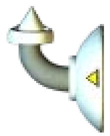

Splitter
(1 input, 2 outputs)

Target
(1 input, 2 outputs)

- input directions (possibly 0)
- output directions (possibly 0)
- flow (possibly nothing)

What attributes are shared?

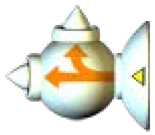

Splitter
(1 input, 3 outputs)


Bender
(1 input, 1 output)

Asteroid

These three are basically the same thing.

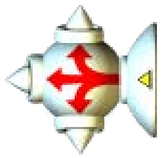
Laser
(0 inputs, 1 output)

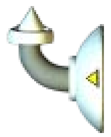

Splitter
(1 input, 2 outputs)

Target
(1 input, 2 outputs)

- input directions (possibly 0)
- output directions (possibly 0)
- flow (possibly nothing)

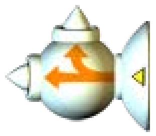
What attributes are shared?


Splitter
(1 input, 3 outputs)


Bender
(1 input, 1 output)

Asteroid
(0 inputs, 0 outputs)

Laser
(0 inputs, 1 output)

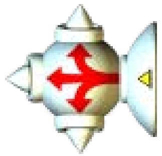

Splitter
(1 input, 2 outputs)

Target
(1 input, 2 outputs)

These ***four*** are basically the same thing.

- input directions (possibly 0)
- output directions (possibly 0)
- flow (possibly nothing)

What attributes are shared?


Splitter/Bender
(1 input, 1-3 outputs)

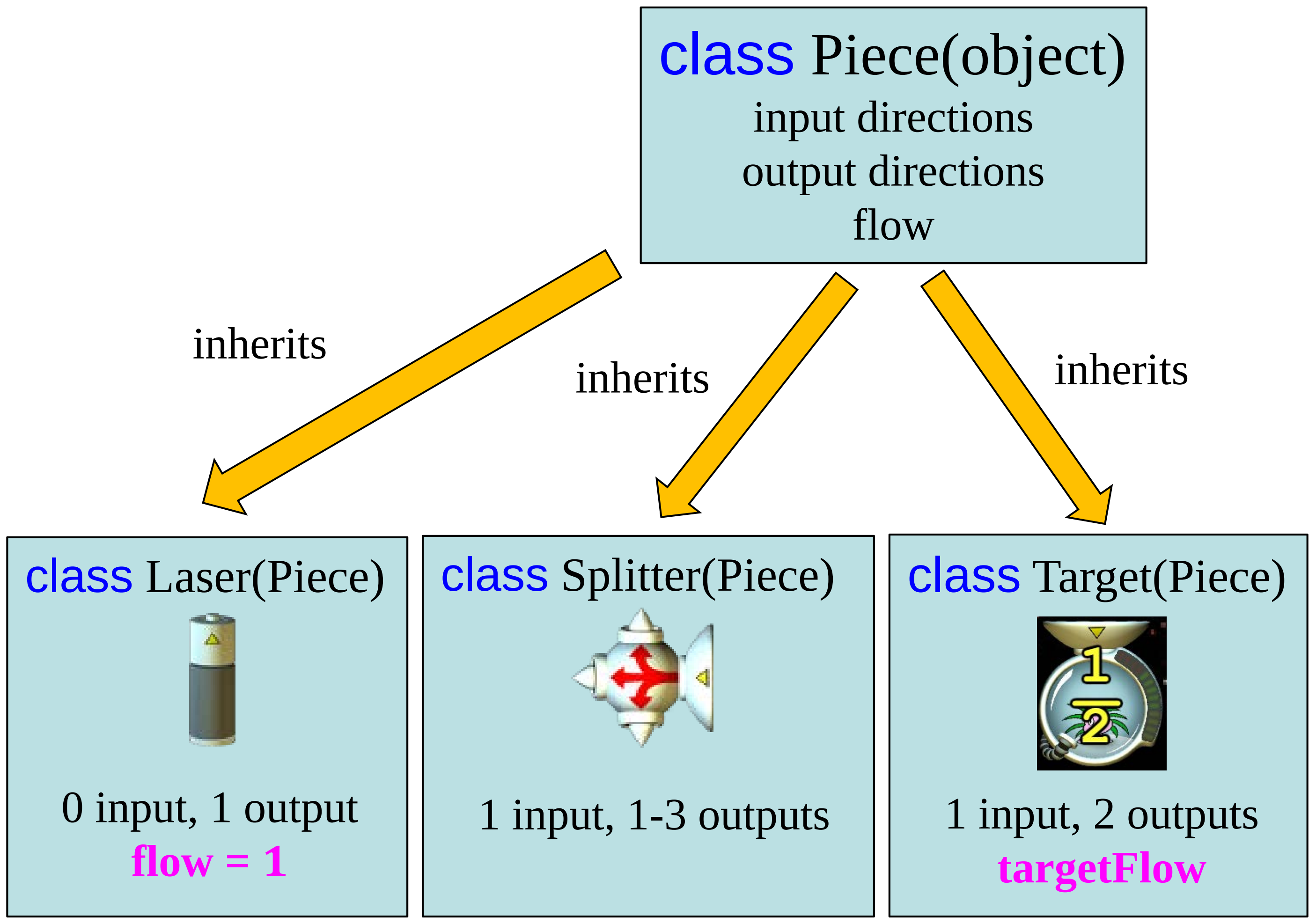
Asteroid
(0 inputs, 0 outputs)

Laser
(0 inputs, 1 output)

Target
(1 input, 2 outputs)

- input directions (possibly 0)
- output directions (possibly 0)
- flow (possibly nothing)

Solution: Inheritance



Initialization: Piece

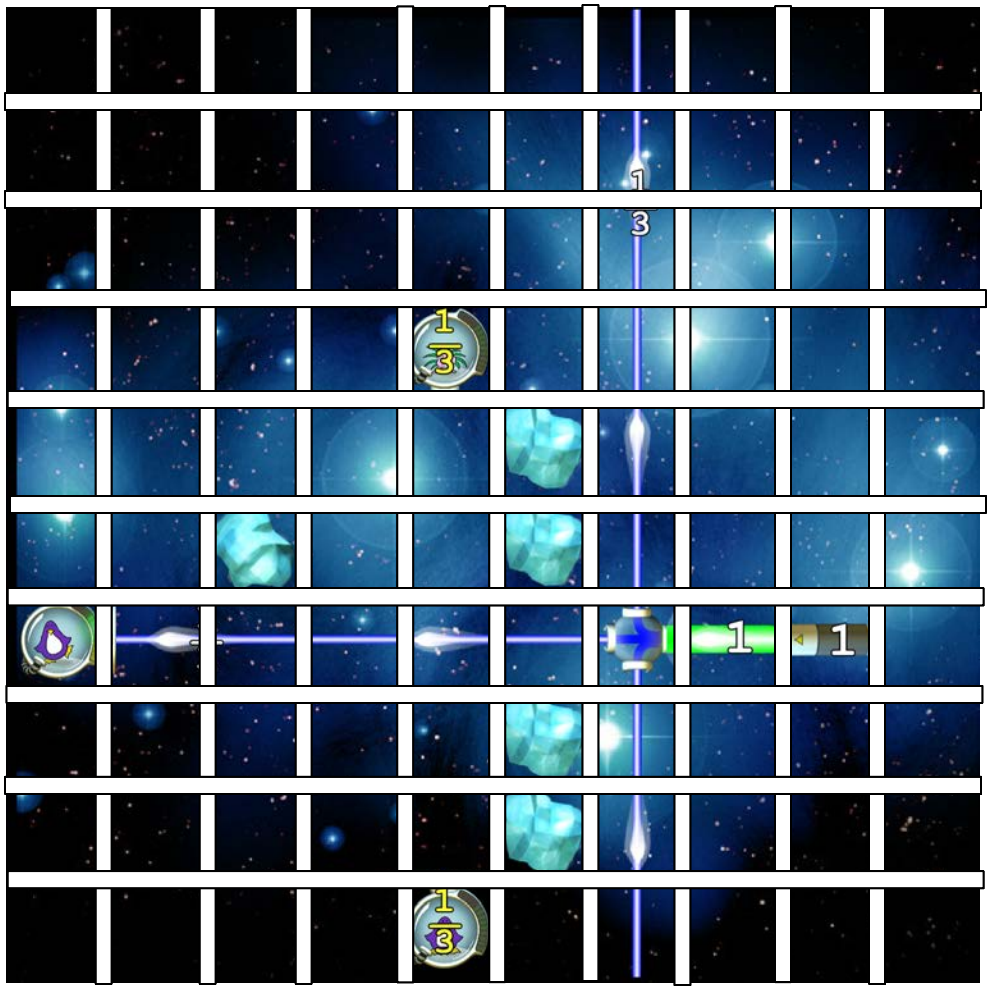
```
class Piece(object):
    def __init__(self):
        self.inputDirections = []
        self.outputDirections = []
        self.flow = Fraction(0, 1)
```

Initialization: Laser

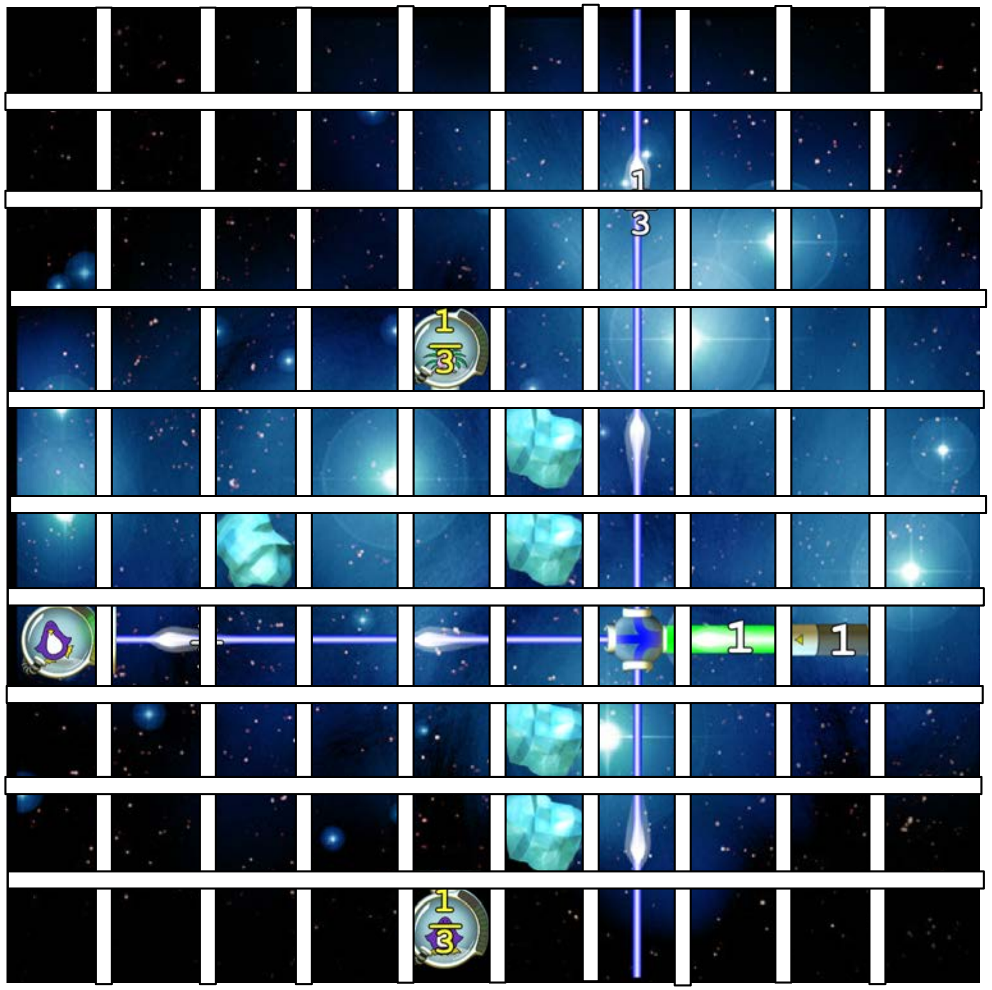
```
class Piece(object):
    def __init__(self):
        self.inputDirections = []
        self.outputDirections = []
        self.flow = Fraction(0, 1)

class Laser(Piece):
    def __init__(self, outputDirection):
        # want to end up with the above and:
        # outputDirections = [outputDirection]
        # flow = Fraction(1, 1)
```

OK, now we need a grid.



What comes to our rescue?

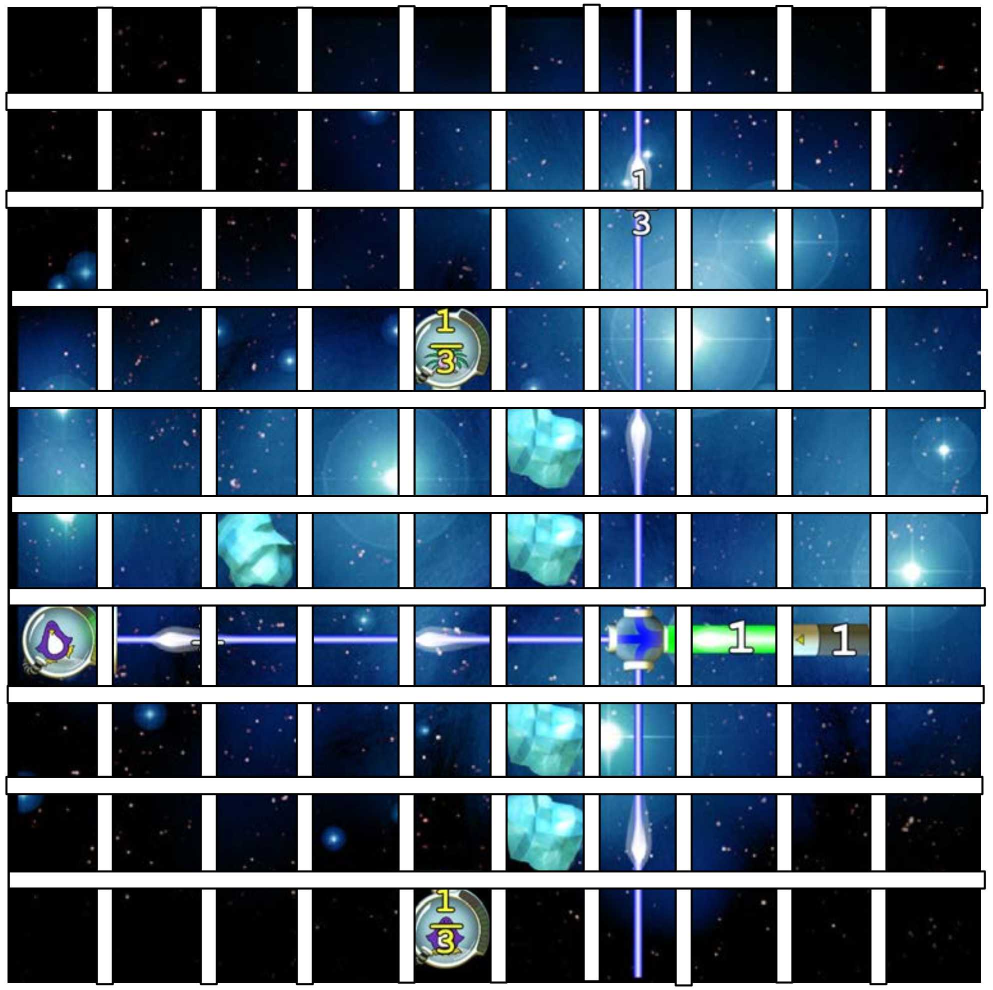


A: List

B: Nested list

C: Dictionary

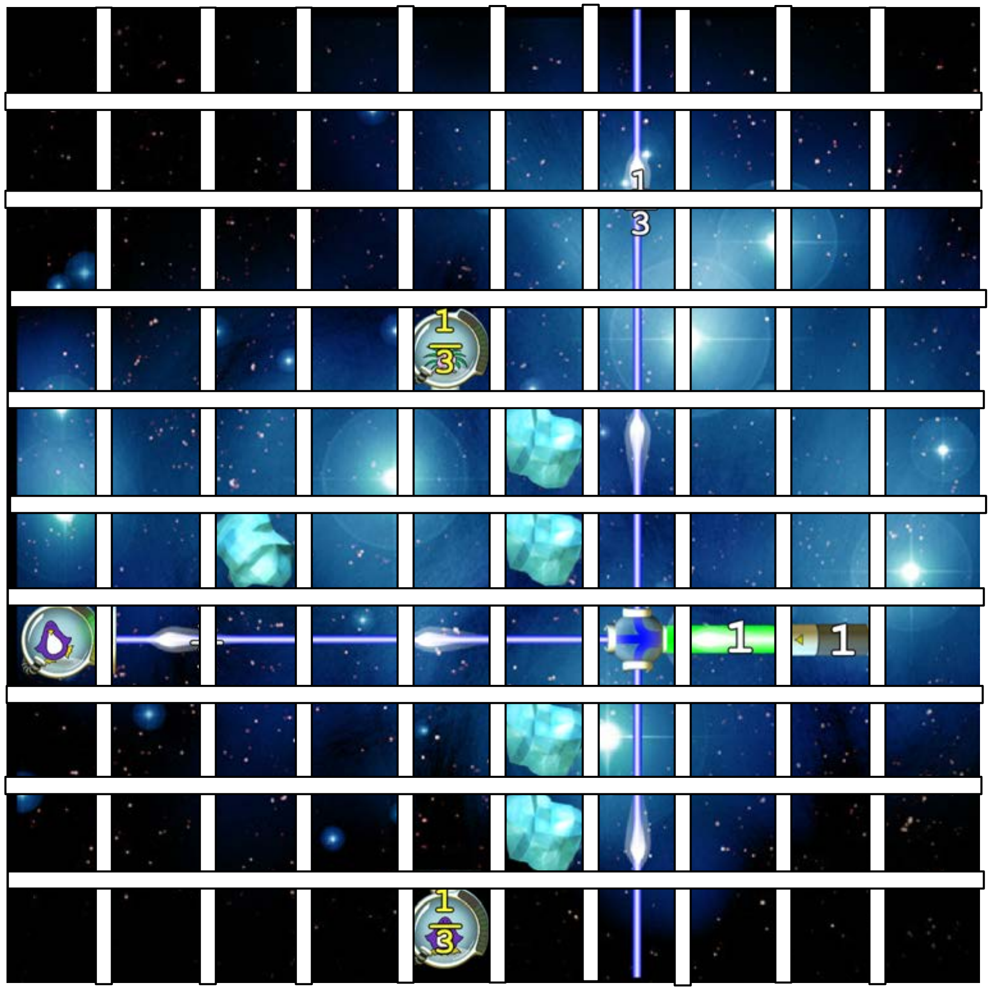
Solution: Two-dimensional lists



```
class Grid(object):
    WIDTH = 10
    cells[][]
```

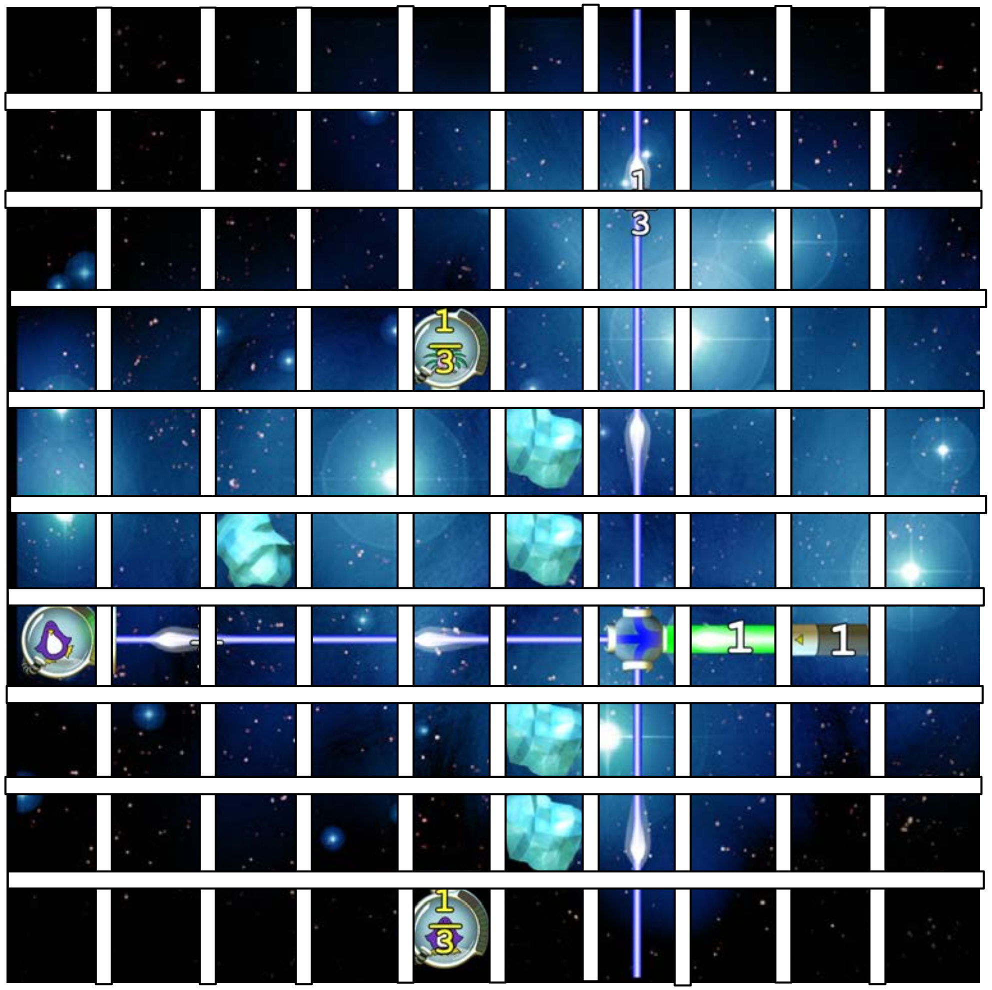
- Holds either a Piece or None
- Row-major order:
  - cells[<row>][<column>] gets piece at <row>, <column>

Need to initialize all cells to None



```
class Grid(object):
    WIDTH = 10
    cells[][]
```

Solution: Nested for-loops

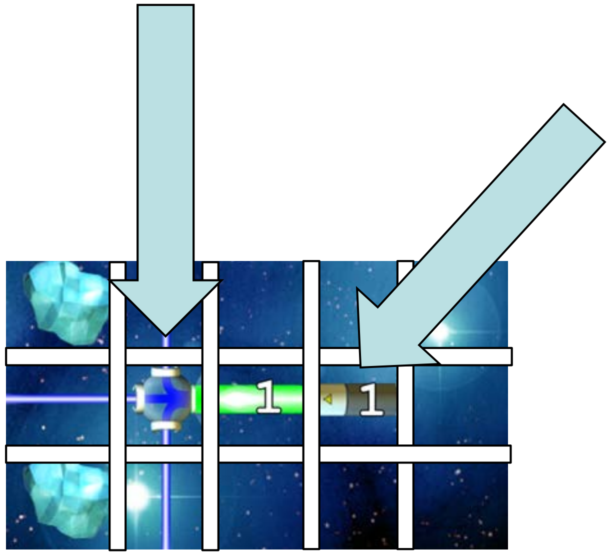


```
class Grid(object):
    WIDTH = 10
    cells[][]
```

Laser Propagation

Keep going left until we hit a piece

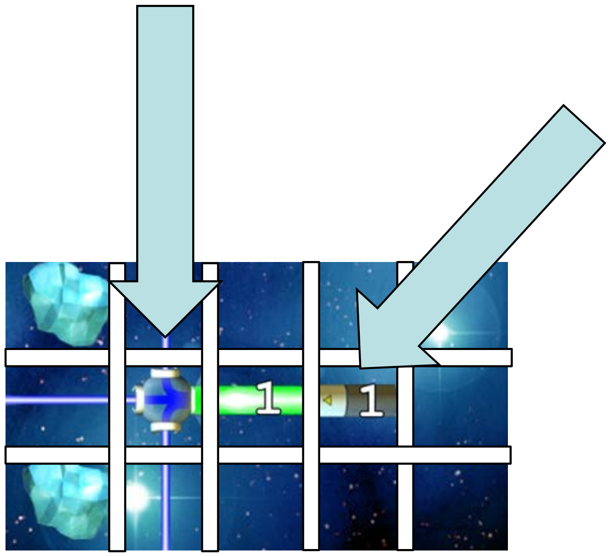
Start from Laser.
outputDirection = LEFT



Solution: while-loop

Keep going left until we hit a piece

Start from Laser.
outputDirection = LEFT



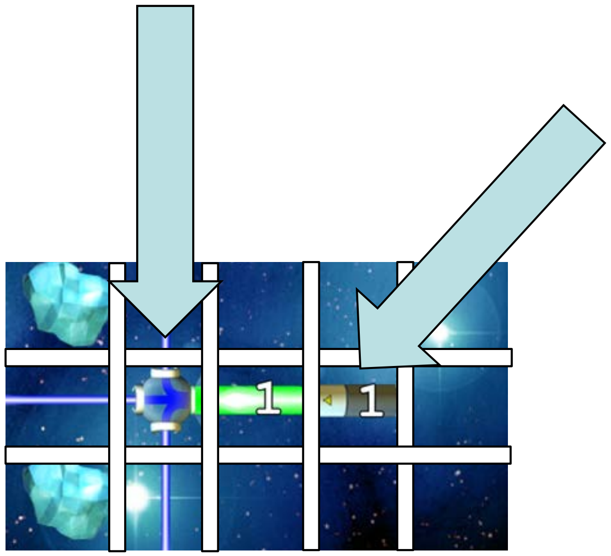
Step 1: Find all of the lasers

```
def sendLasers(self):
    for row in range(Grid.WIDTH):
        for column in range(Grid.WIDTH):
            piece = self.cells[row][column]
            if isinstance(piece, Laser):
                # push laser from piece
```

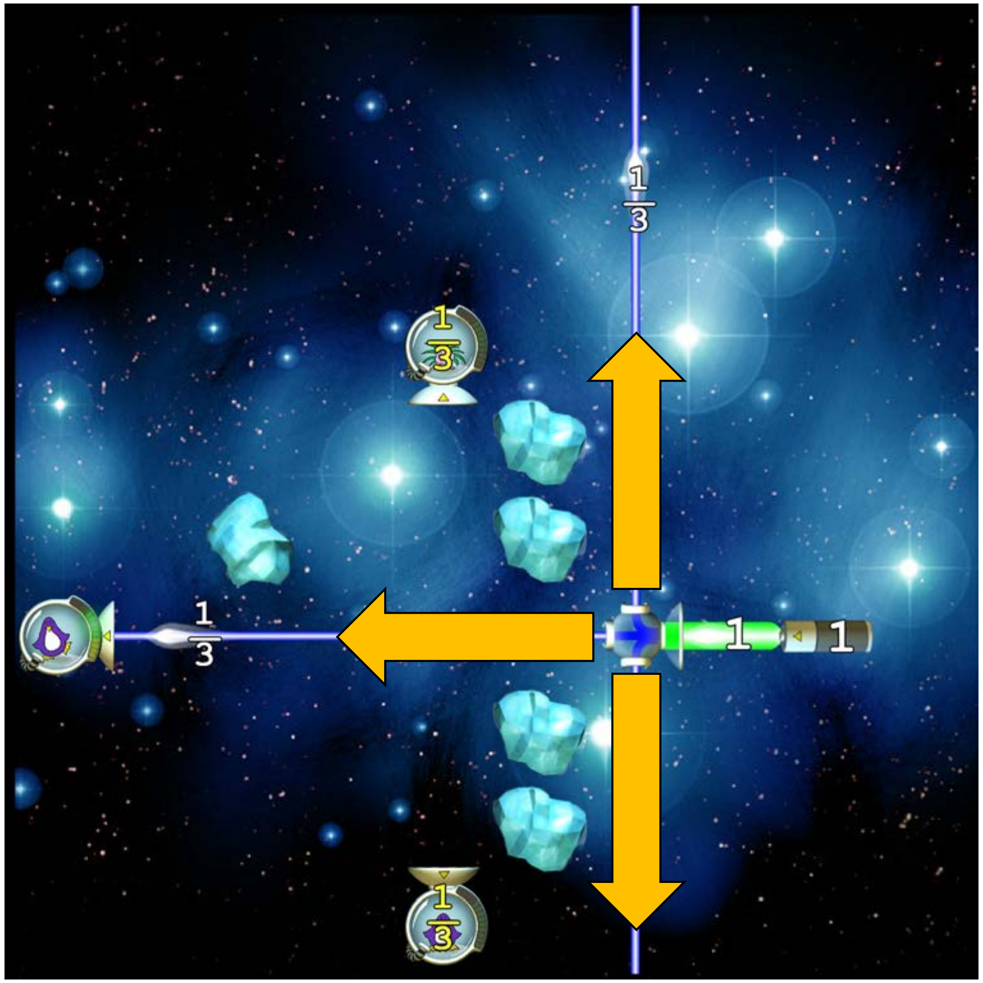
Step 2: Keep stepping

Keep going left until we hit a piece or the edge

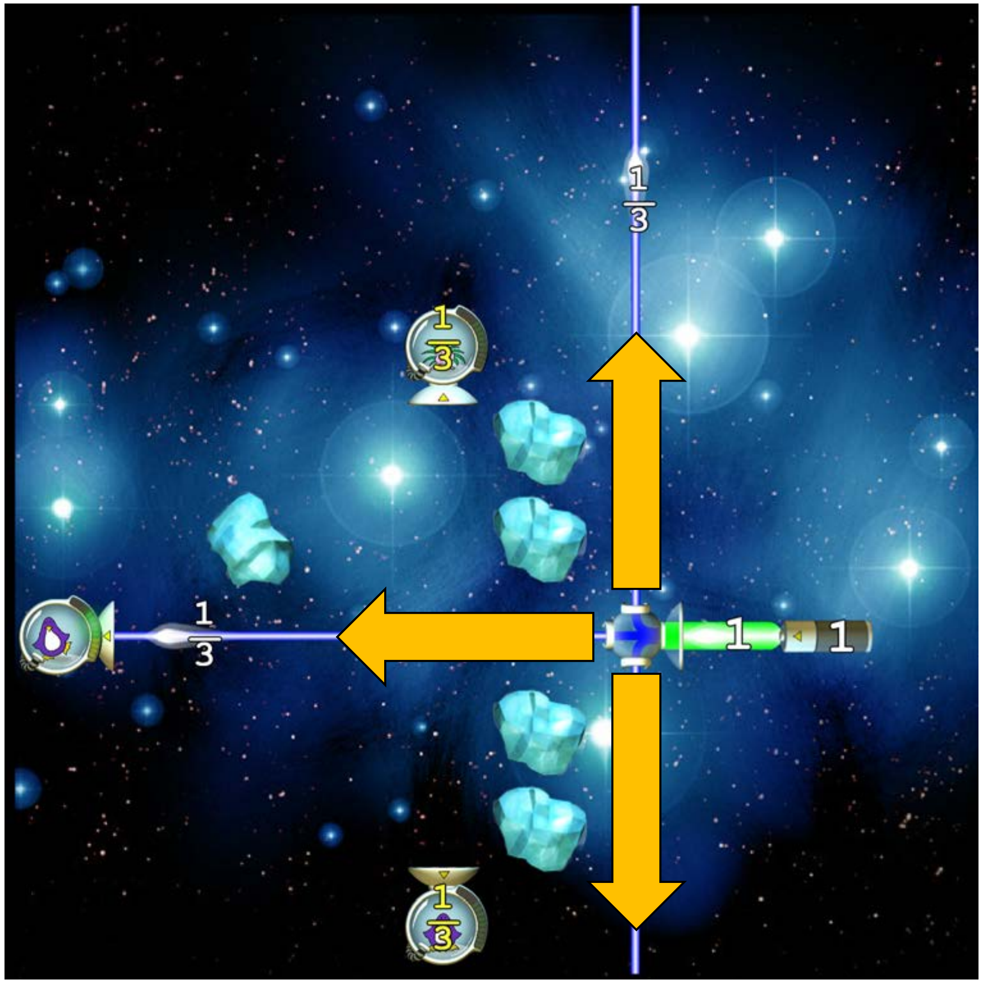
Start from Laser.
outputDirection = LEFT



Now we need to go in the output directions



Recursion to the rescue!



Did I Win?

- Loop through all the cells
- Find the targets
- Check if target is powered