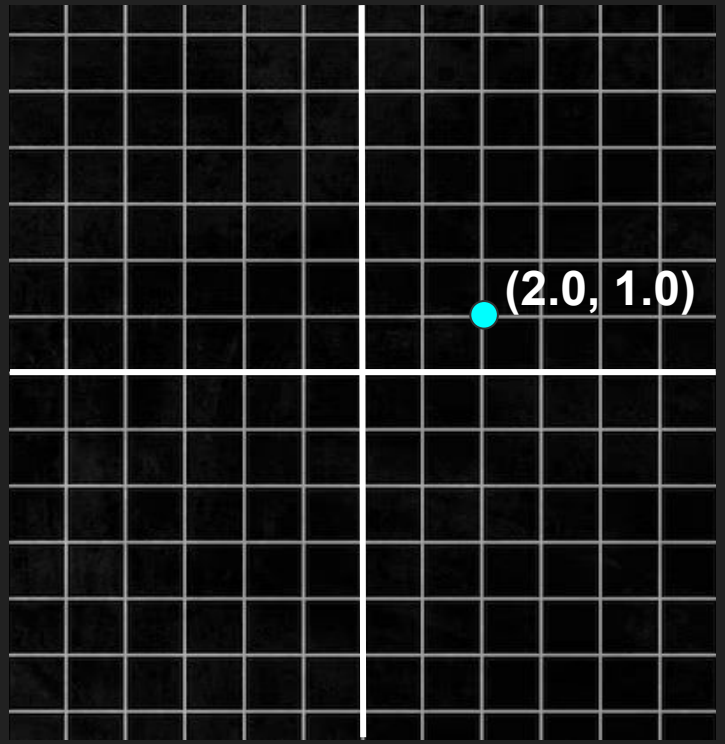
🪄 Magic Methods 🪄

Announcements

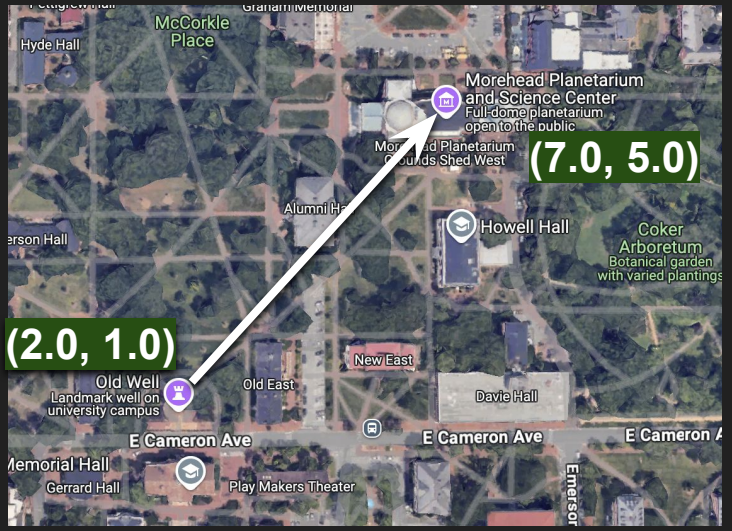
EX06 due Thursday at 11:59pm!

Recall this Point class

```
0  class Point:
1      x: float
2      y: float
3
4      def __init__(self, x: float, y: float):
5          self.x = x
6          self.y = y
7
8      def dist_from_origin(self) -> float:
9          return (self.x**2 + self.y**2) ** 0.5
10
11     def translate_x(self, dx: float) -> None:
12         self.x += dx
13
14     def translate_y(self, dy: float) -> None:
15         self.y += dy
16
17 pt: Point = Point(2.0, 1.0)
```



```
1-16 class Point: … # collapsed for space
  17
  18 class Line:
  19     start: Point
  20     end: Point
  21
  22     def __init__(self, start: Point, end: Point):
  23         self.start = start
  24         self.end = end
  25
  26     def get_length(self) -> float:
  27         x_diffs: float = self.end.x - self.start.x
  28         y_diffs: float = self.end.y - self.start.y
  29         return (x_diffs**2 + y_diffs**2) ** 0.5
  30
  31     def get_slope(self) -> float:
  32         x_diffs: float = self.end.x - self.start.x
  33         y_diffs: float = self.end.y - self.start.y
  34         return y_diffs / x_diffs
```



Create a Line object and find the distance from the Old Well to the Planetarium:

```
me: Point = Point(2.0, 1.0)
planet_y: float = 5.0
planet_loc: Point = Point(me.x + 5, planet_y) # Example of accessing an attribute's value
path: Line = Line(me, planet_loc)
print(path.get_length())
```

On your own: try printing a Point or Line object!

```
0 class Point:
1     x: float
2     y: float
3
4     def __init__(self, x: float, y: float):
5         self.x = x
6         self.y = y
7
8     def dist_from_origin(self) -> float:
9         return (self.x**2 + self.y**2) ** 0.5
10
11    def translate_x(self, dx: float) -> None:
12        self.x += dx
13
14    def translate_y(self, dy: float) -> None:
15        self.y += dy
16
17 pt: Point = Point(2.0, 1.0)
```

What happens?

On your own: try printing a Point or Line object!

```
0  class Point:
1      x: float
2      y: float
3
4      def __init__(self, x: float, y: float):
5          self.x = x
6          self.y = y
7
8      def dist_from_origin(self) -> float:
9          return (self.x**2 + self.y**2) ** 0.5
10
11     def translate_x(self, dx: float) -> None:
12         self.x += dx
13
14     def translate_y(self, dy: float) -> None:
15         self.y += dy
16
17 pt: Point = Point(2.0, 1.0)
```

What happens?

<__main__.Point object at 0xffff9506d9a0>

(or some other, seemingly random bunch of numbers and letters)

Let’s implement some magic in VS Code!