15-112
Lecture 2

Week 9 Thu
Recursion

Instructor: Pat Virtue

Announcements

Hack 112!

OH this weekend

- Heads up! Staff will be split between HW9 and Hack 112

Term Project

- Ideation meetings (required)
- Mini-Lectures this week (must attend at least one)
- Instructions (posted soon) (will be part of pre-reading checkpoint)

Announcements

Hack 112!

OH this weekend

- Heads up! Staff will be split between HW9 and Hack 112

Term Project

- Ideation meetings (required)
- Mini-Lectures this week (must attend at least one)
- Instructions (posted soon) (will be part of pre-reading checkpoint)

General Recursive Form

```
def recursiveFunction():
    if (this is the base case):
        do something non-recursive
    else:
        do something recursive
```

recursiveFunction

Recursion Example

- Recursive case
- Base case
- Recursion errors
- Call Stack
- Visualizing recursion
- Debugging recursion

Poll 1

Which is the best base case

[✓ A.]
```
if n == 0
    return 0
```

✗ B.
```
if n == 0
    return 1
```

✗ C.
```
if n == 1
    return 0
```

[✓ D.]
```
if n == 1
    return 1
```

~ E.
```
if n == 2
    return 3
```

Debugging!

Notes: Recursive Debugging

Hazards!

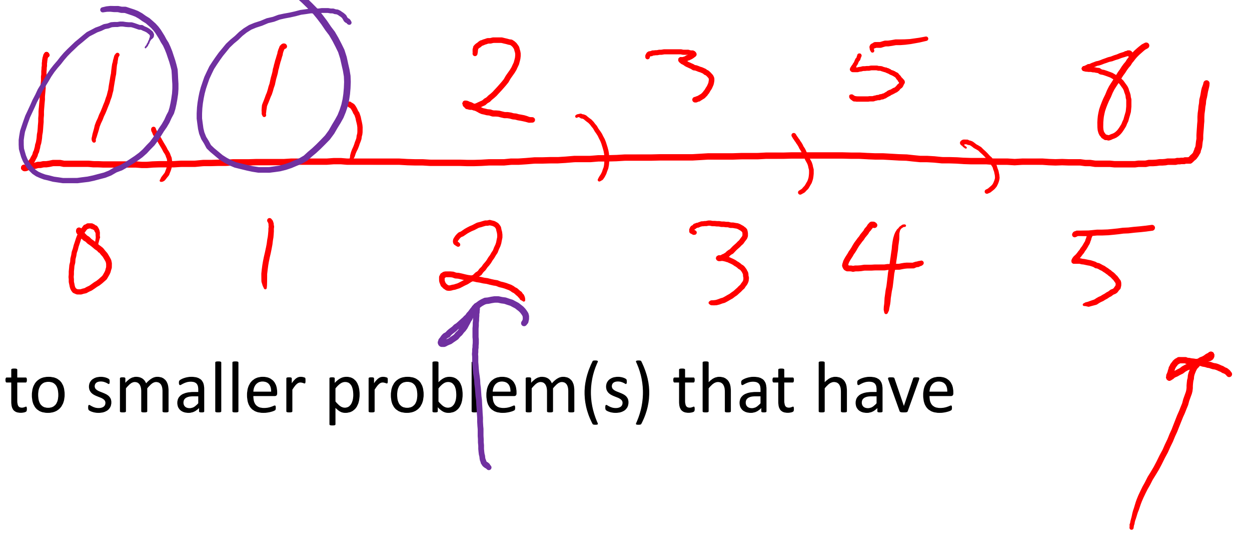
Notes: Hazard Extra Recursive Calls

Recursive thinking

Suggestion: start with the recursive case

- How can you reduce the problem into smaller problem(s) that have the same structure as the original?
- Assume (magically) that next recursive cases will work

Multiple recursive cases



Example Fibonacci

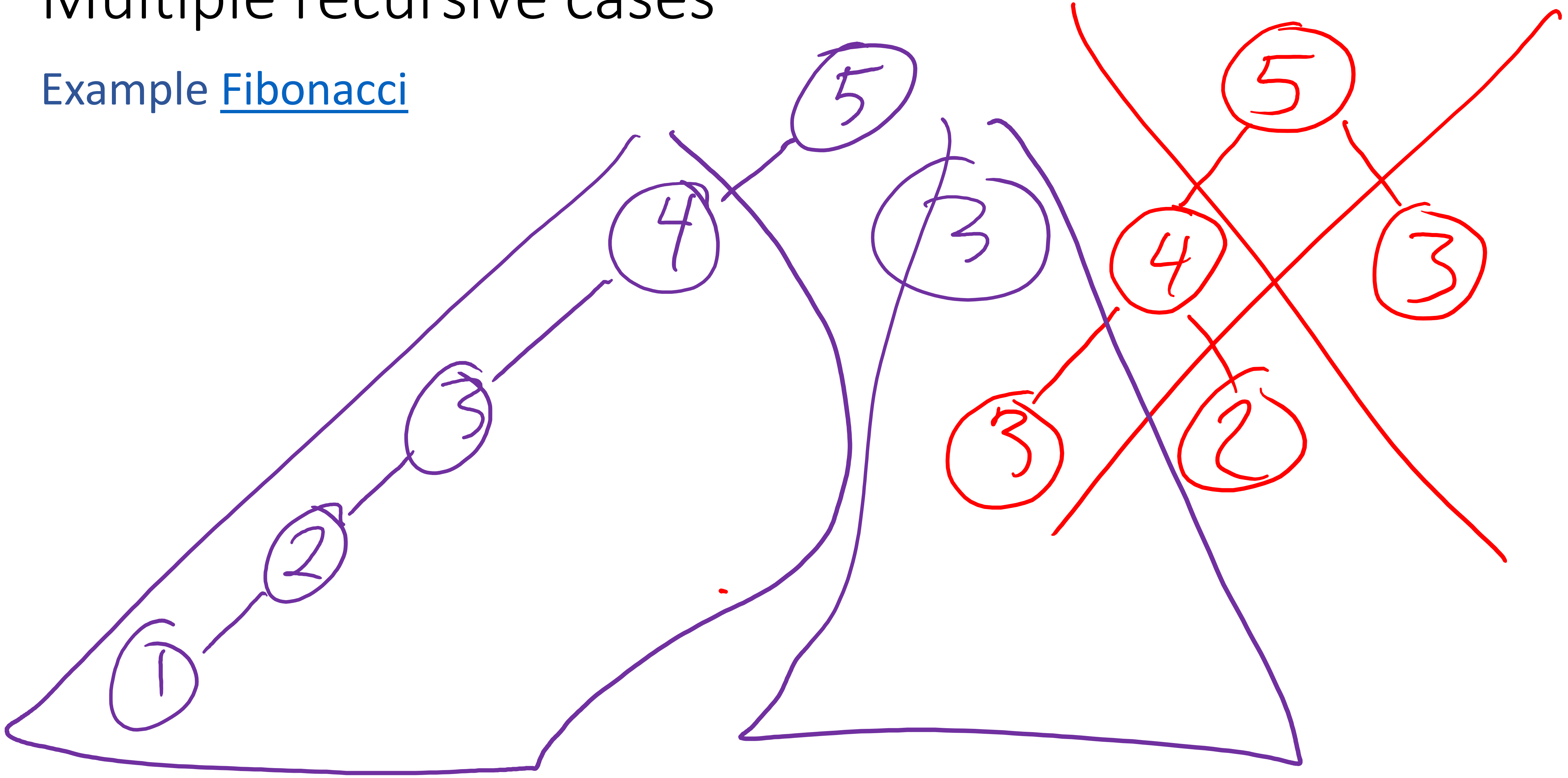
- How can you reduce the problem into smaller problem(s) that have the same structure as the original?
- Assume (magically) that next recursive cases will work

```
def fib6()
    result = fib4() + fib5()
    return result
```

fib5
fib4

Multiple recursive cases

Example Fibonacci



Towers of Hanoi

Goal: Move stack to a different peg

Restrictions

- One piece at a time
- Can't put bigger piece on top of smaller

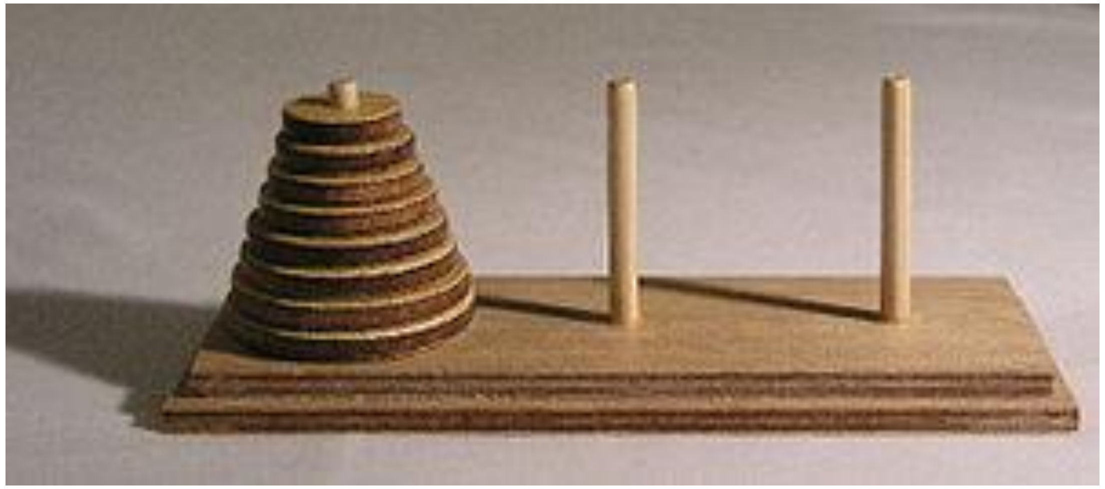


Image (left): https://commons.wikimedia.org/wiki/File:Tower_of_Hanoi.jpeg

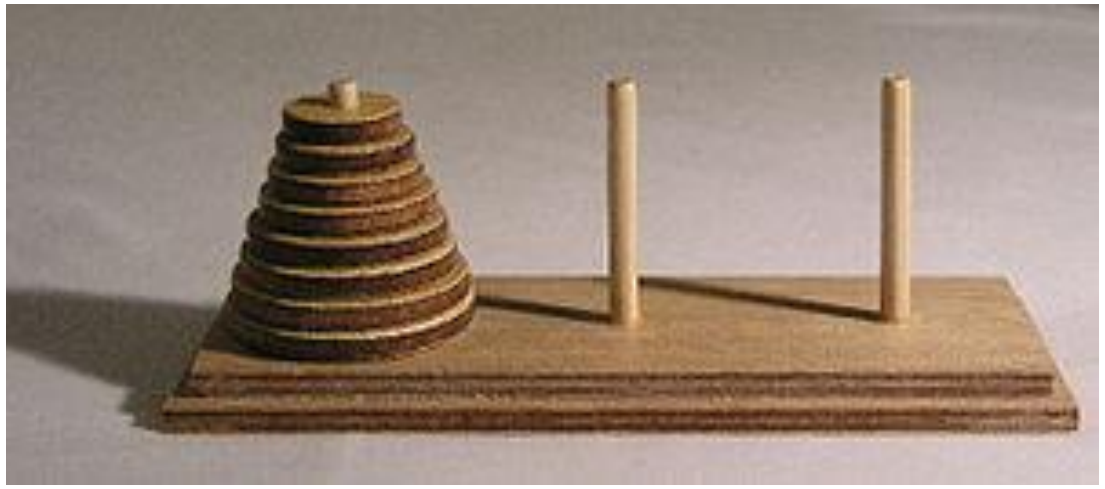
Reminder General Recursive Form

```
def recursiveFunction():
    if (this is the base case):
        do something non-recursive
    else:
        do something recursive
```

Towers of Hanoi

Recursive case

- Let’s start with magic!



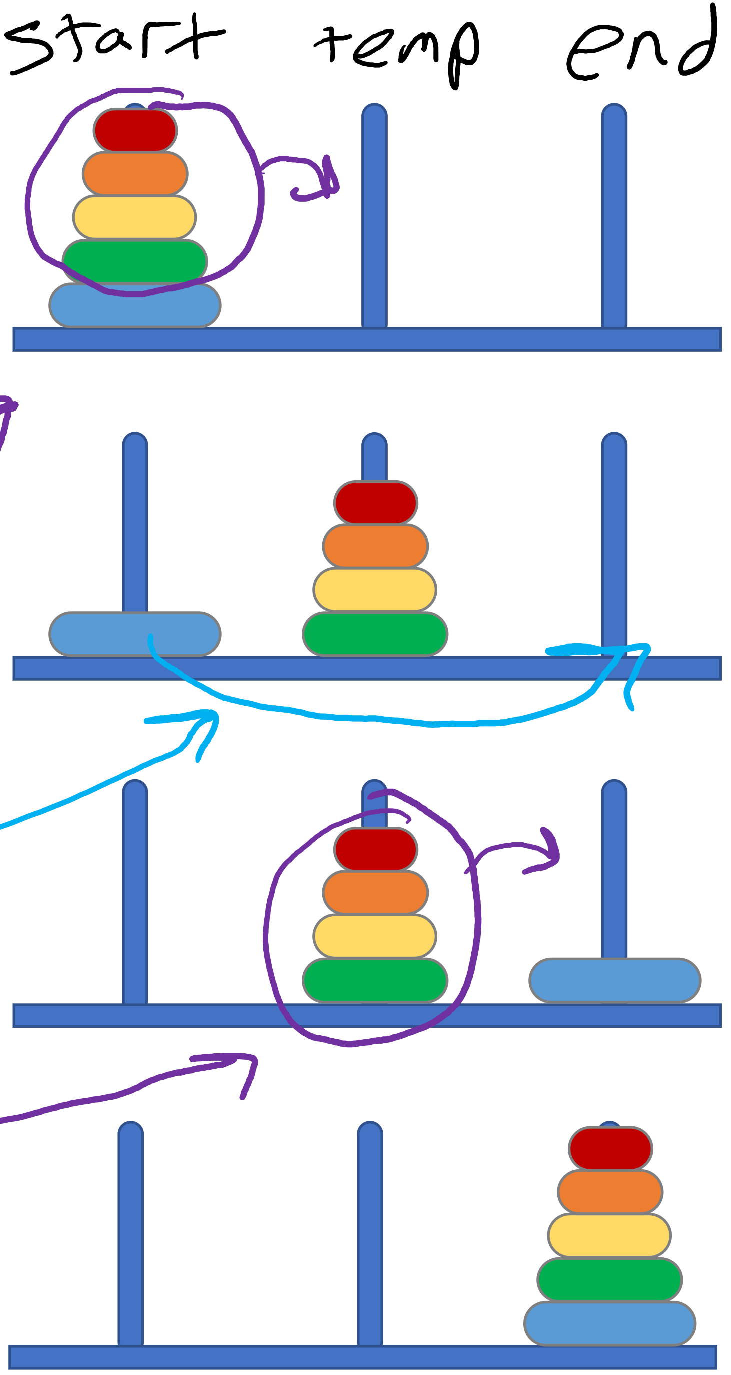
Towers of Hanoi

Recursive case

- Let’s start with magic!

```
import magic # For now :)

def move5(start, end, temp):
    # Move 5 pieces from start to end
    magic.move4(start, temp, end)
    print(f"Move piece from {start} to {end}")
    magic.move4(temp, end, start)
```



Revisit Merge Sort

Merge sort: $O(N \log N)$

Merge concept:

Assume you had two piles that were already independently sorted.

Could you shuffle them together into one sorted pile in O(N)?

Fractals

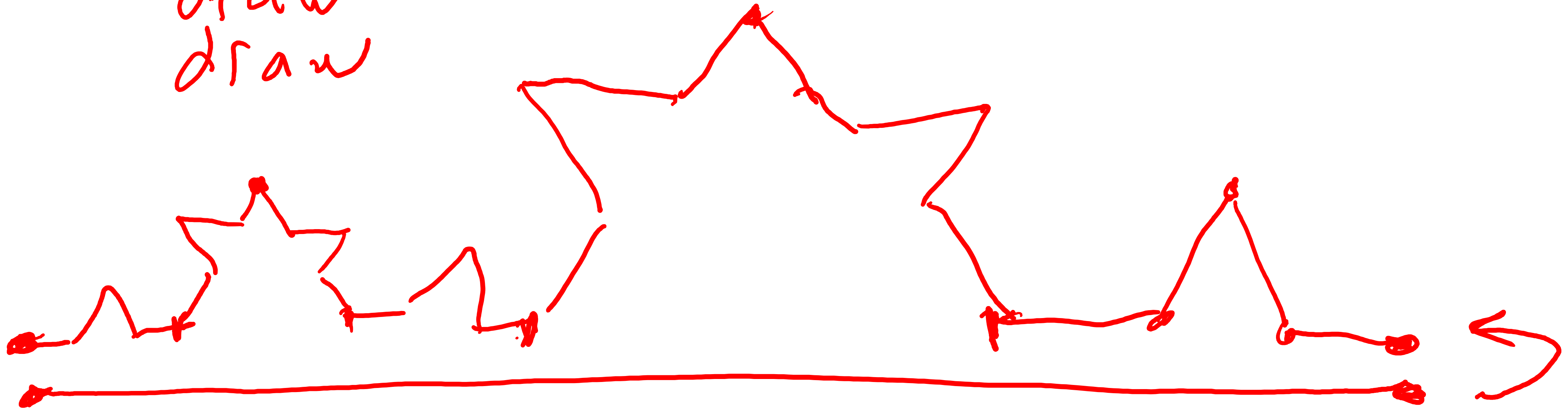
Mandelbrot set

https://www.youtube.com/watch?v=u1pwtSBTnPU

Reminder: Fractals

Koch curve

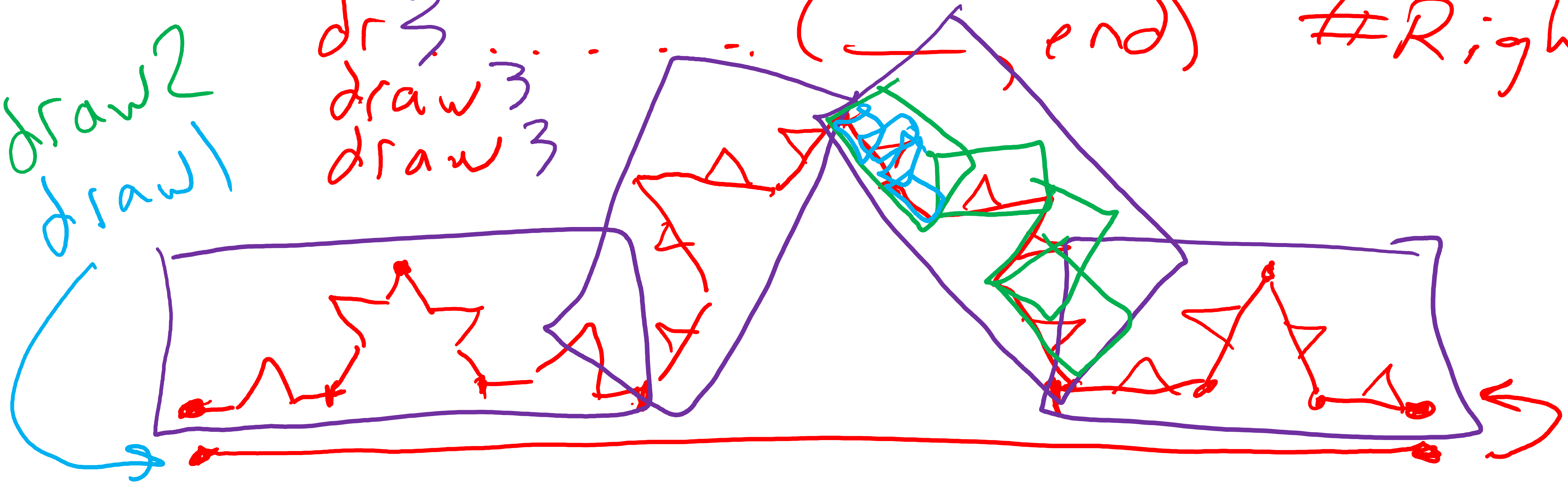
```
drawKochLine(start, end)
    drawKochLine(start, ___)   #Left
    dr.......(___, end)        #Right
    draw
    draw
```



Reminder: Fractals

Koch curve

```
drawKochLine4(start, end)
    drawKochLine3(start, ___)    #Left
    (___, end)                   #Right
    dr3 . . . . .
    draw3
    draw3
draw2
draw1
```



Reminder: Fractals

Koch curve

```
def drawFractal(app, canvas, level, start, end):
    dist = math.sqrt((end[0]-start[0])**2 + (end[1]-start[1])**2)

    if level == 0 or dist <= 1:
        canvas.create_line(start[0], start[1], end[0], end[1])
    else:
        point1, point2, point3 = newKochPoints(start, end)
        drawFractal(app, canvas, level-1, start, point1, color)
        drawFractal(app, canvas, level-1, point1, point2, color)
        drawFractal(app, canvas, level-1, point2, point3, color)
        drawFractal(app, canvas, level-1, point3, end, color)
```

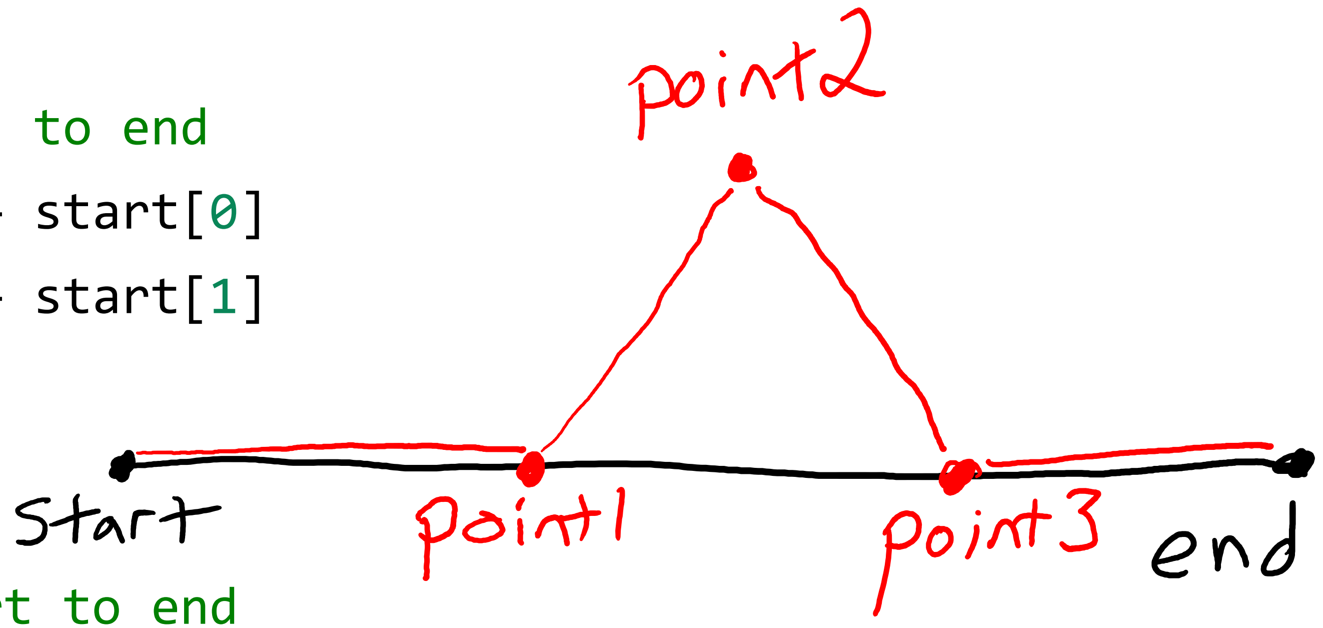
Reminder: Fractals

Koch curve

```
def newKochPoints(start, end):
    # Point1
    # One third of the way from start to end
    point1x = (end[0]-start[0])*1/3 + start[0]
    point1y = (end[1]-start[1])*1/3 + start[1]
    point1 = (point1x, point1y)

    # Point3
    # Two thirds of the way from start to end
    point3x = (end[0]-start[0])*2/3 + start[0]
    point3y = (end[1]-start[1])*2/3 + start[1]
    point3 = (point3x, point3y)

    # Point2 ...
```



Reminder: Fractals

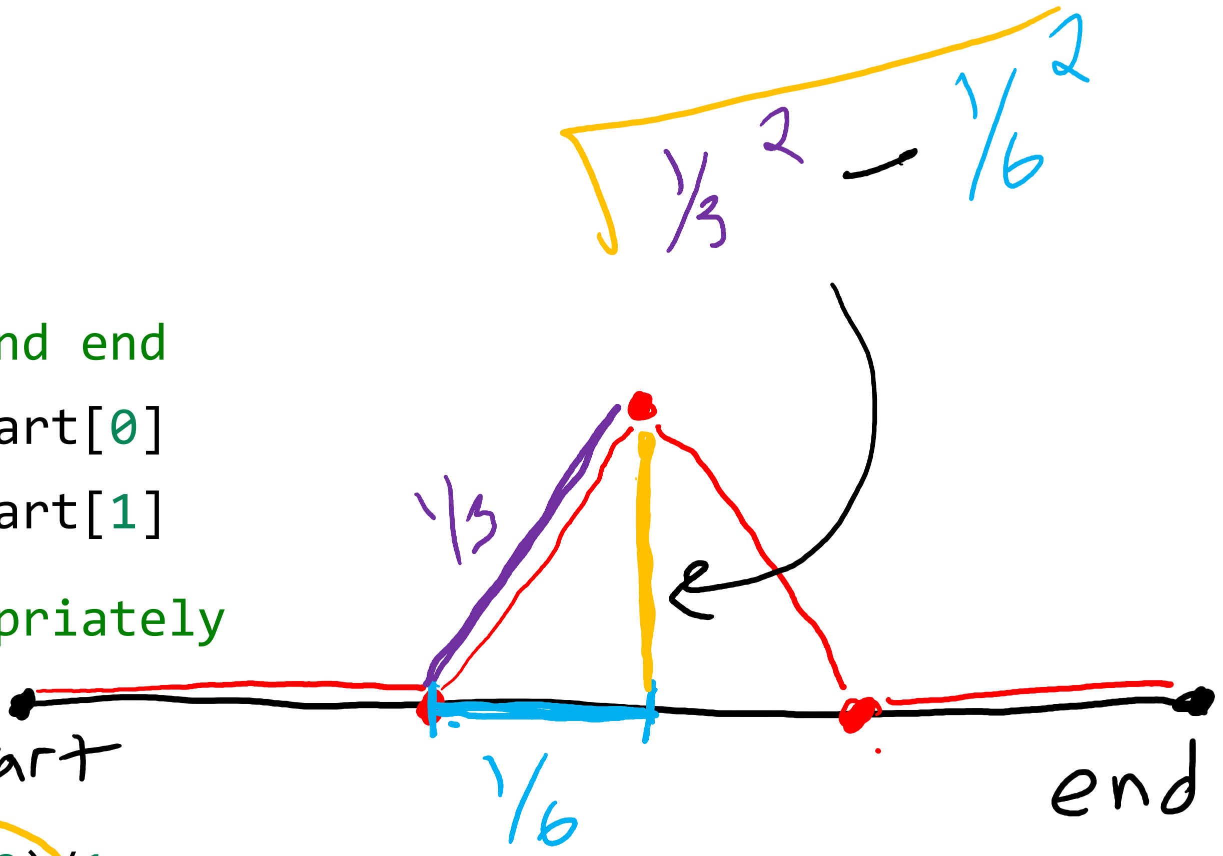
```python
def newKochPoints(start, end):
    ...
    # Point2
    # Start with halfway between start and end
    point2x = (end[0]-start[0])*1/2 + start[0]
    point2y = (end[1]-start[1])*1/2 + start[1]

    # perpendicular change, scaled appropriately
    dy = -(end[0]-start[0])
    dx = (end[1]-start[1])
    scale = math.sqrt((1/3)**2 - (1/6)**2)/1
    point2x += scale*dx
    point2y += scale*dy
    point2 = (point2x, point2y)

    return (point1, point2, point3)
```

intentional

full line



Reminder General Recursive Form

```
def recursiveFunction():
    if (this is the base case):
        do something non-recursive
    else:
        do something recursive
```