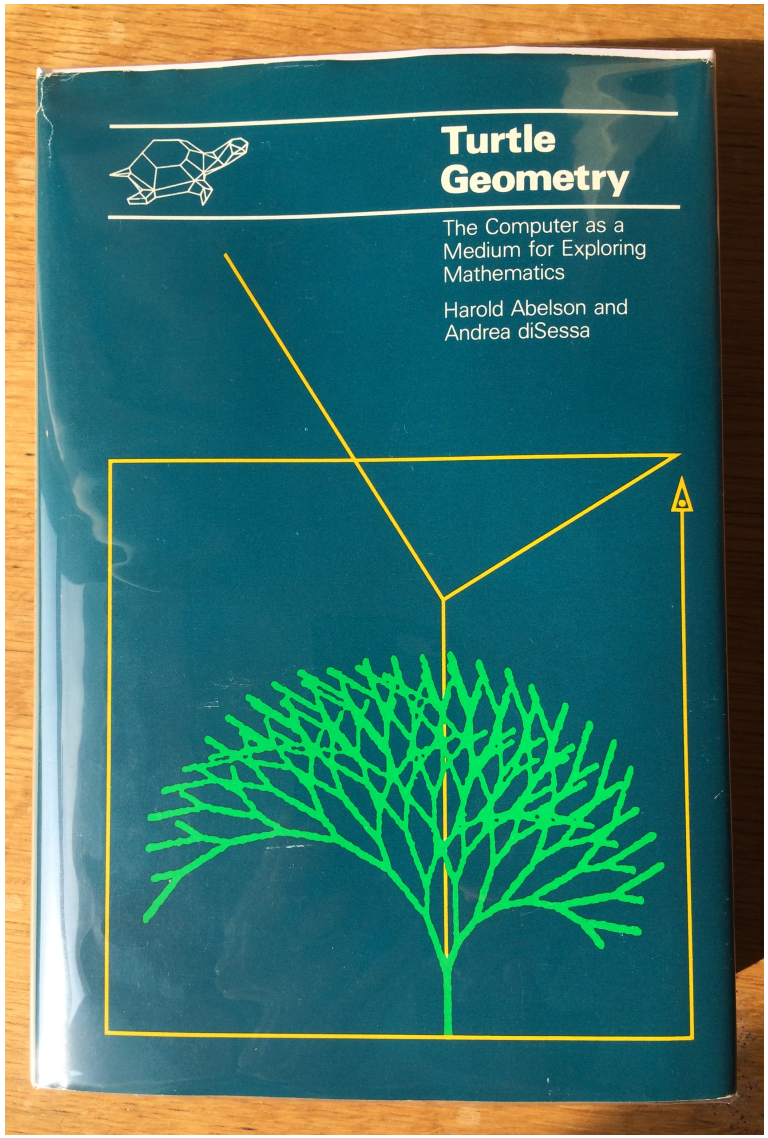
Turtle Geometry the Python Way

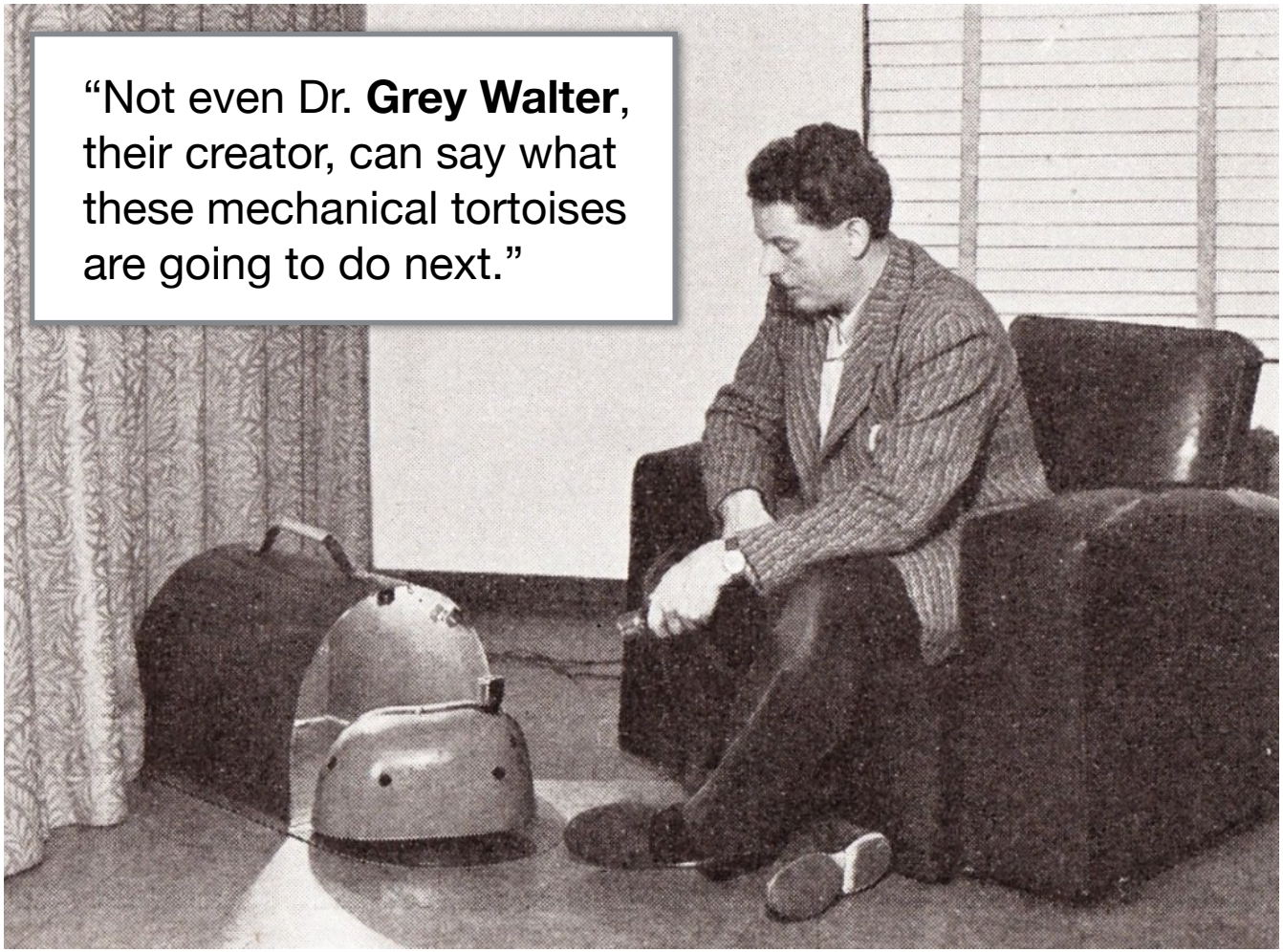
@SteveBattle

Turtle Geometry



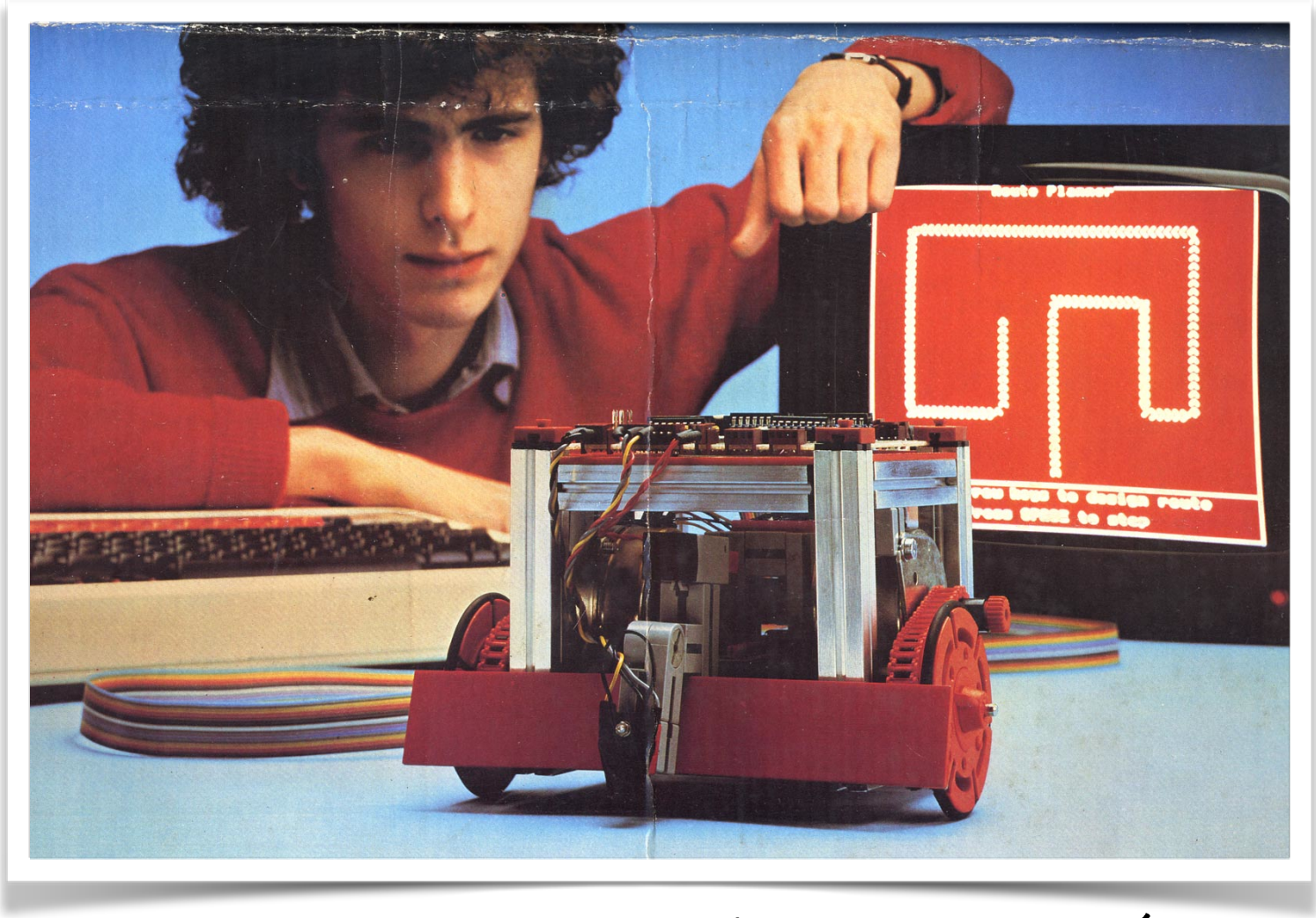
"The tradition of calling our display creatures 'turtles' started with ***Grey Walter****, a neurophysiologist who experimented in Britain… with tiny robot creatures he called 'tortoises'. These inspired the first turtles designed at MIT." (1980)*

Turtles and Tortoises



Bristol based Inventor of the first autonomous robots in 1948.

BBC Buggy (1983)



The Buggy was tethered to the BBC Micro and used the 'Logo' language.

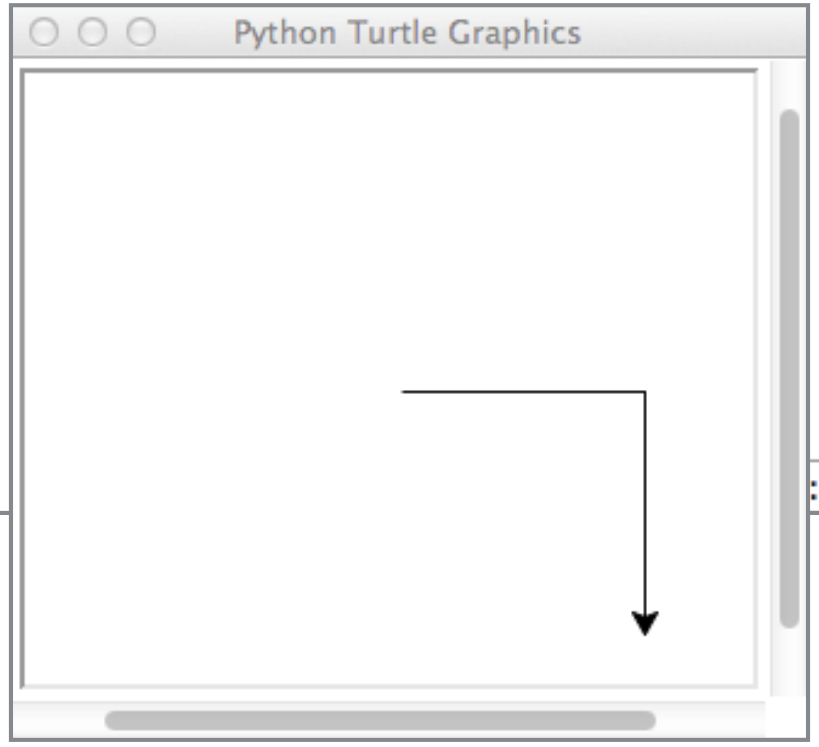
The Python Way

Python 3.4.1 Shell

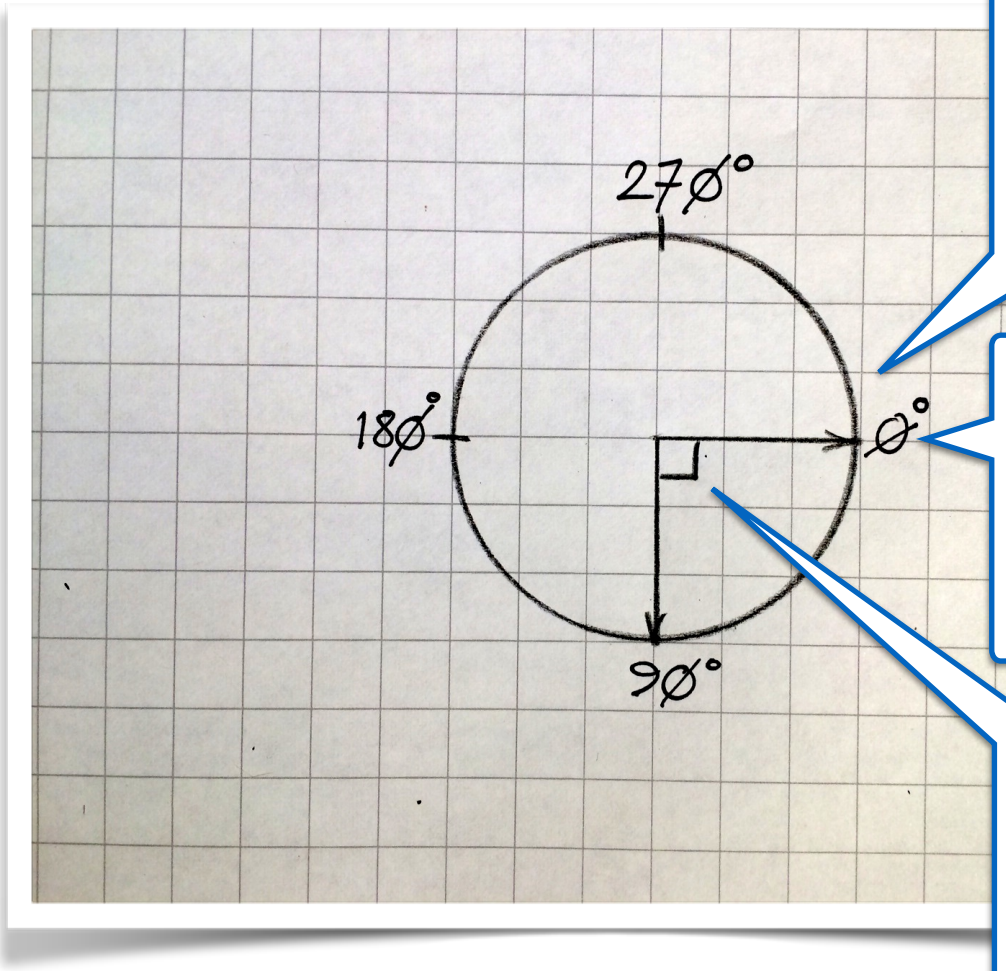
```
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 00:54:21)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> from turtle import *
>>> forward(100)
>>> right(90)
>>> forward(100)
>>>
```

Start IDLE and start typing

Python Turtle Graphics



angles



A 360° turn returns the turtle to 0°

The turtle starts off facing this way

A right angle going clockwise

sequence

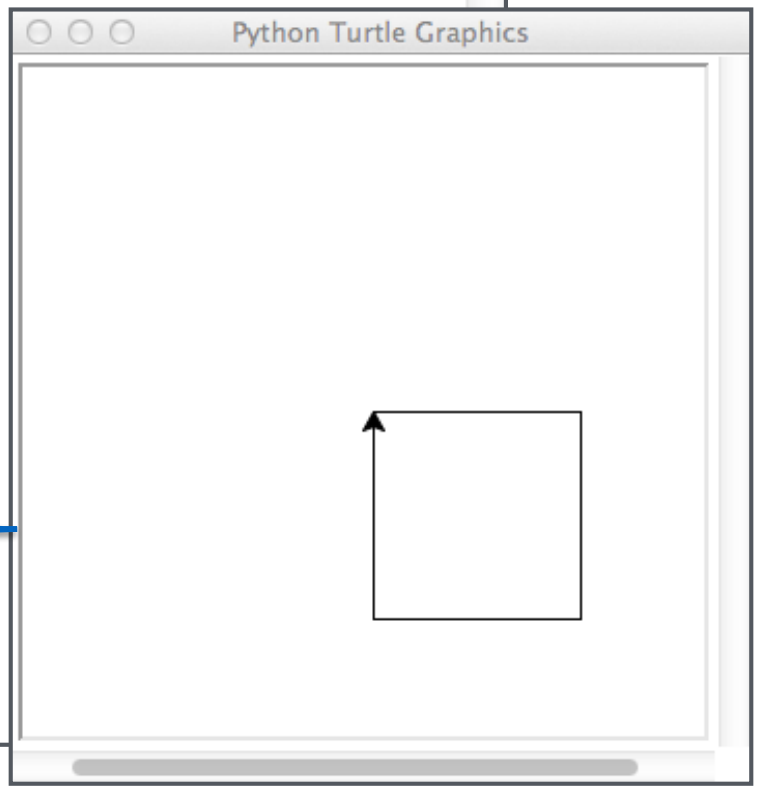
Create a new file
(Python module)

Save the file
(but not as 'turtle')

Run

Python 3.4.1: sequence.py – /Users/Steve/Do...

```
from turtle import *

pendown()

forward(100)
right(90)
forward(100)
right(90)
forward(100)
right(90)
forward(100)

done()
```



turtle functions

forward(**length**)	backward(**length**)
left(**angle**)	right(**angle**)
penup()	pendown()
done()	
speed(**s**)	*e.g. 'slow', 'fast', 'fastest'*
shape(**name**)	*e.g. 'turtle', 'classic'*
goto(**x**,**y**)	*x,y coordinates*

for loops

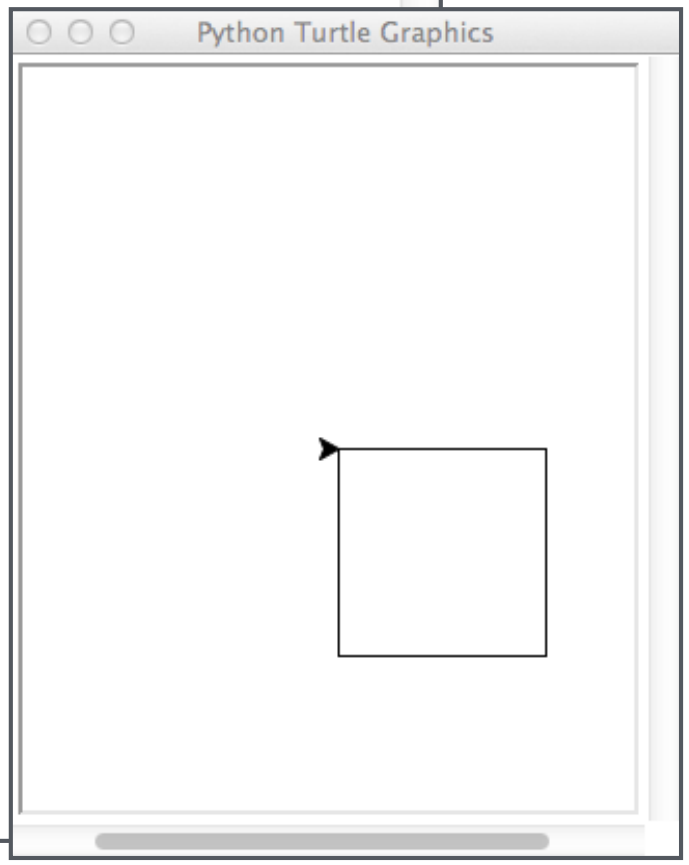
The loop repeats a fixed number of times (4).

Everything inside the loop is indented. Use TAB not SPACE

Python 3.4.1: square.py – /Users/Steve/Doc...

```
from turtle import *

speed('slow')
pendown()

for i in range(4):
    forward(100)
    right(90)

done()
```

slow down the output



defining functions

Function definition

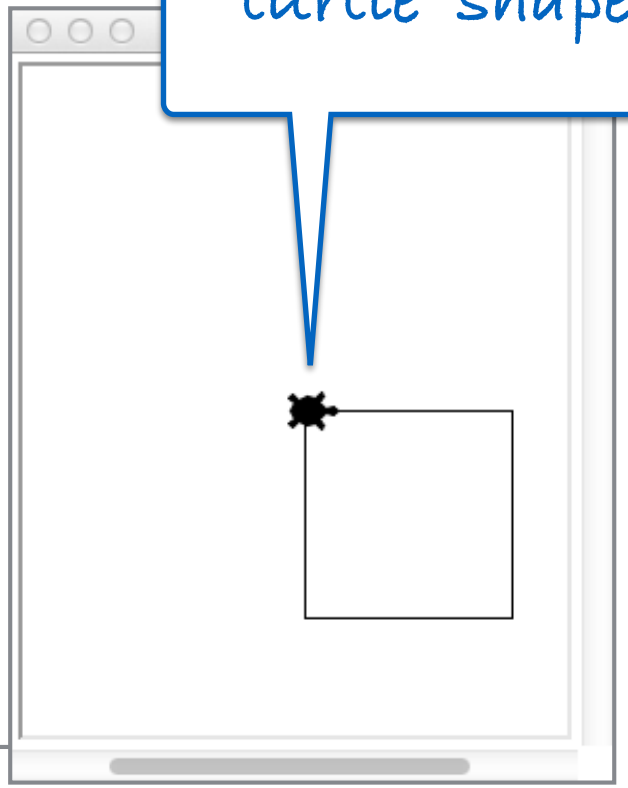
Note the double indentation

Call the function

Python 3.4.1: square_fun.py – /Users/Steve/...

```
from turtle import *

def square():
    for i in range(4):
        forward(100)
        right(90)

speed('slow')
shape('turtle')
pendown()

square()

done()
```

'turtle' shape



function parameters

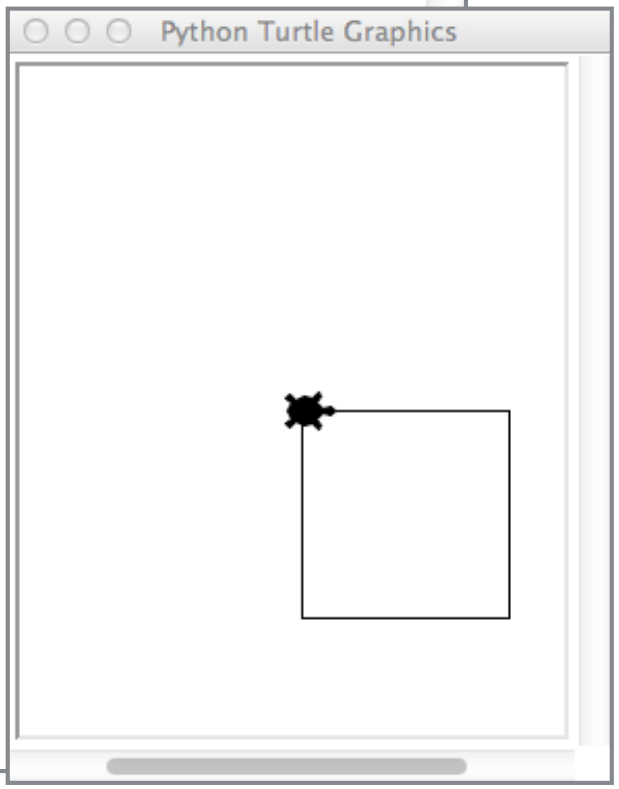
Python 3.4.1: square_len.py – /Us

```
from turtle import *

def square(length):
    for i in range(4):
        forward(length)
        right(90)

speed('slow')
shape('turtle')
pendown()

square(100)

done()
```

Add parameter 'length'

Call 'square' with argument 100

Python Turtle Graphics



polygons

calculate angle and assign to a variable

Comma separated parameters

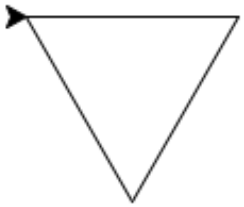
The angles of a polygon sum to 360°

The output from 'print' appears in the IDLE console

Python 3.4.1: polygon.py - /Users/Steve/Do...

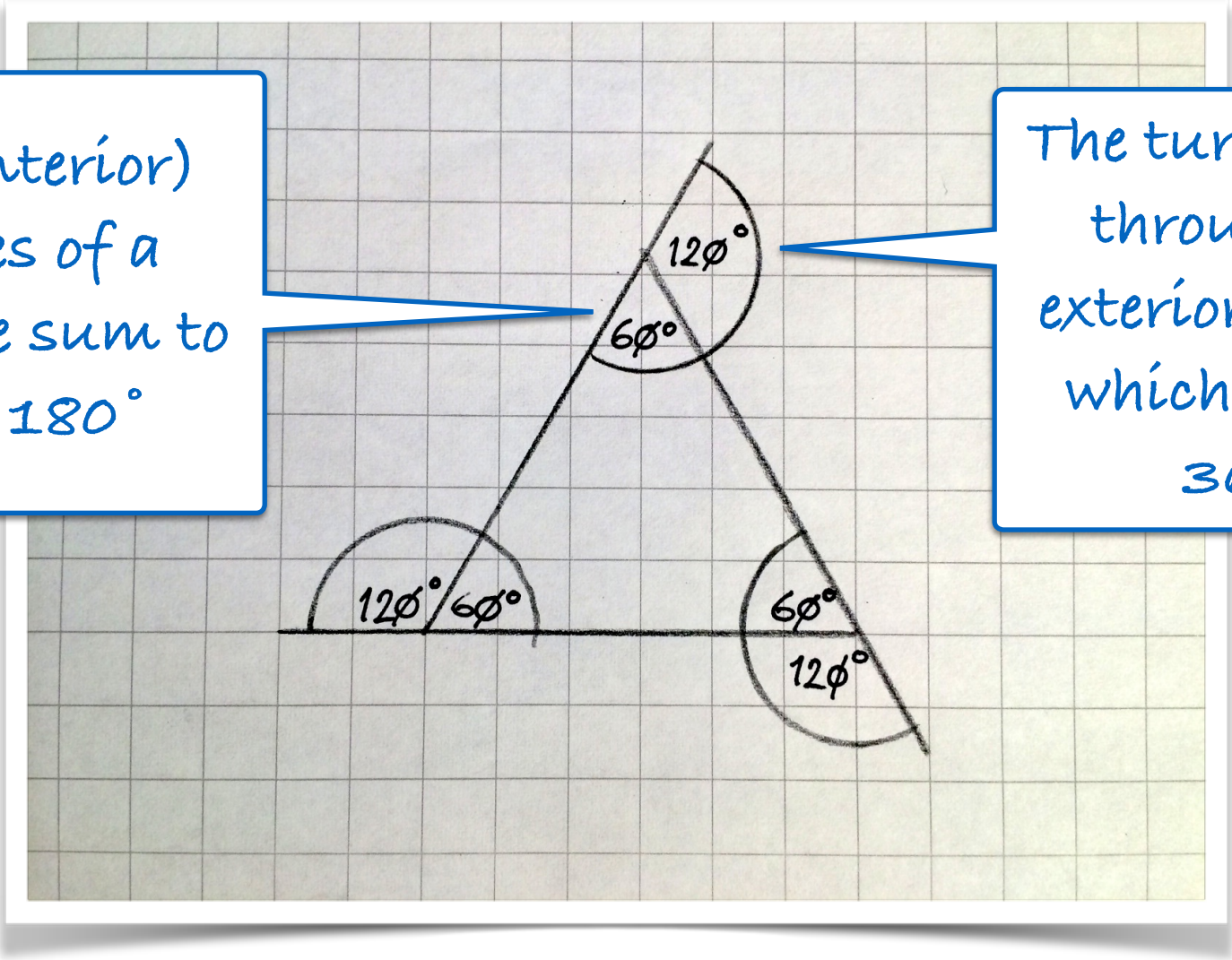
```
from turtle import *

def polygon(length, sides):
    for i in range(sides):
        forward(length)
        angle = 360/sides
        print(angle)
        right(angle)

speed('slow')
pendown()
polygon(100,3)
done()
```

```
Python 3.
[GCC 4.2.
Type "cop
>>> =====
>>>
120.0
120.0
120.0
```



Interior & Exterior angles

The (interior) angles of a triangle sum to only 180°



The turtle turns through the exterior angles, which sum to 360°

stars

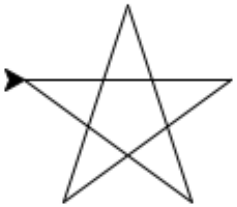
What happens if we turn turtle through a multiple of 360°?

Python 3.4.1: star.py – /Users/Steve/Documents/Python/star.py

```
from turtle import *

def star(length, sides, multiple):
    for i in range(sides):
        forward(length)
        angle = multiple * 360/sides
        print(angle)
        right(angle)

speed('slow')
pendown()
star(100,5,2)
done()
```

```
Python
[GCC 4.
Type "co
>>> ====
>>>
144.0
144.0
144.0
144.0
144.0
```

Double the angle of a normal pentagon



Try also heptagons:

star(100,7,2)

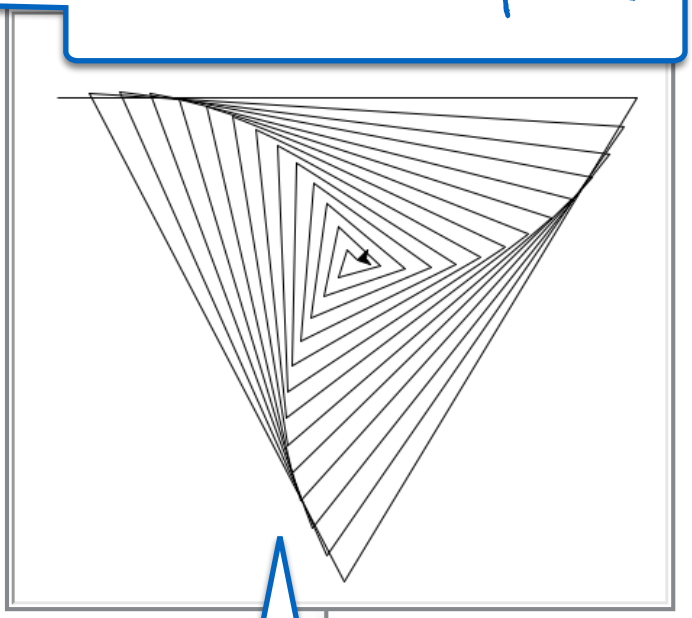
star(100,7,4)

while loops

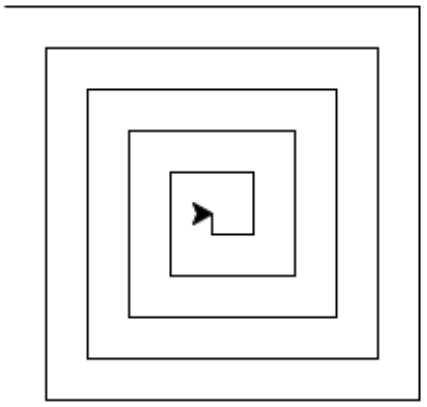
Python 3.4.1: spiral.py – /Users/Steve/Documents/Python/spi

```
from turtle import *

def spiral(length, sides, multiple, decrement):
    while length>0:
        forward(length)
        right(multiple * 360/sides)
        length = length - decrement

speed('slow')
penup()
goto(-200,200)
pendown()
spiral(200,4,1,10)
done()
```

Exit the 'while' loop when the condition is false.



Python Turtle Graphics



'goto' these coordinates.

Try a non-integer multiple:
spiral(400, 3, 1.01, 10)

recursion

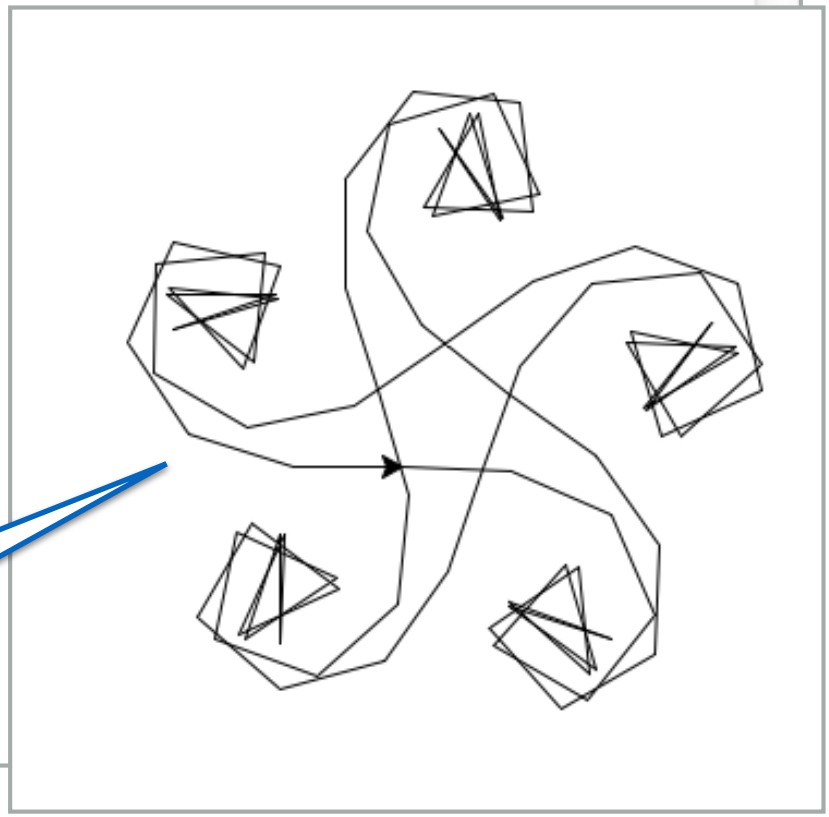
Python 3.4.1: inspiral.py – /Users/Steve/Docume

```
from turtle import *

def inspiral(length, angle, increment):
    forward(length)
    right(angle)
    inspiral(length, (angle + increment) % 360, increment)

speed('slow')
pendown()
inspiral(50,2,20)
done()
```

We can also get spiral motion by increasing the angle

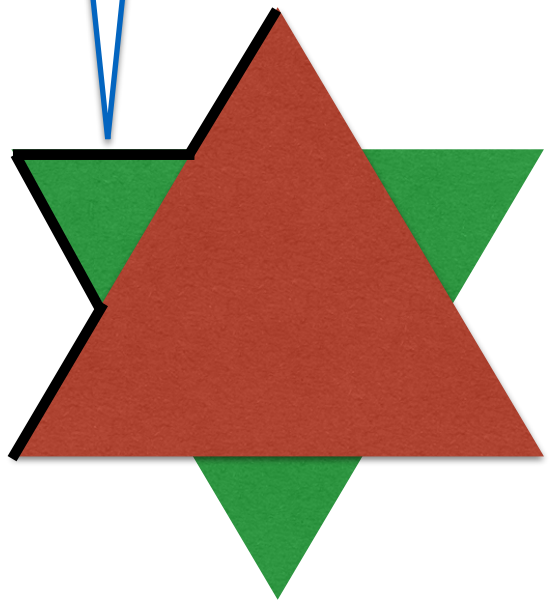
Recursive functions call themselves.

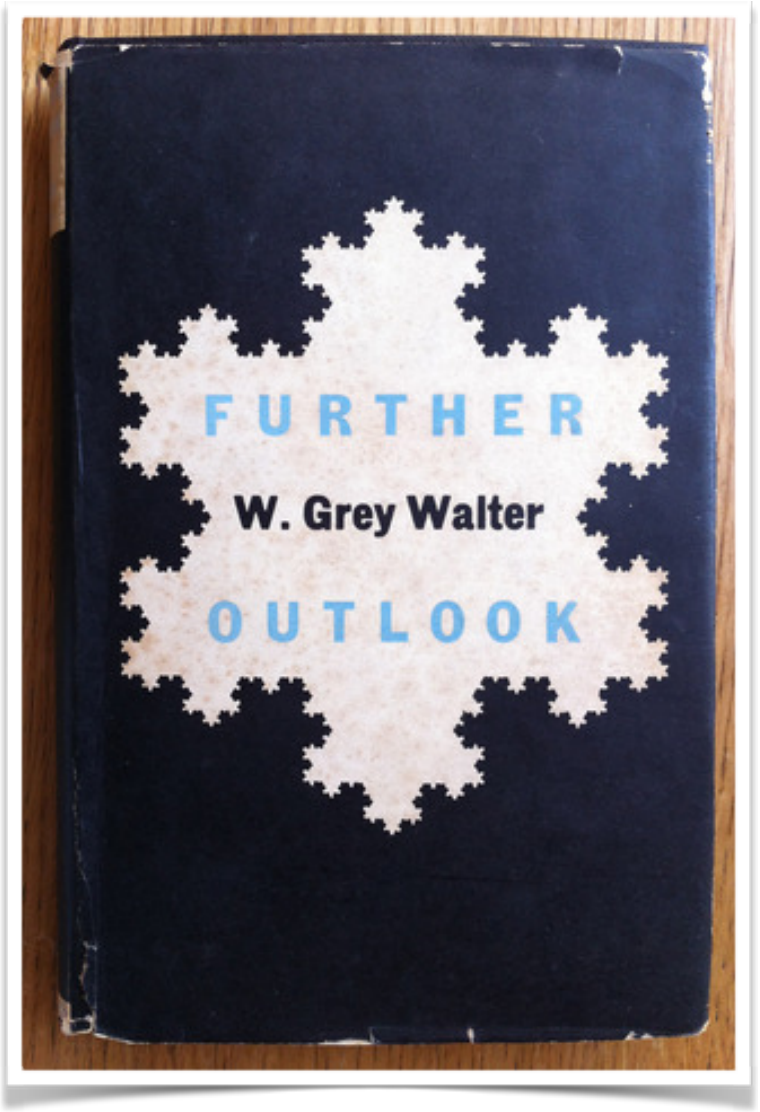
The angle is close to 0 on the 'straights'



The Snowflake Curve

Every side has smaller sides and so on, ad infinitum.





conditionals

Python 3.4.1: snowflake.py – /Users/Steve/Documents/Python/snowflake.py

```
from turtle import *

def snowflake(length, level):
    for i in range(3):
        side(length,level)
        right(120)

def side(length,level):
    if level==0:
        forward(length)
    else:
        side(length/3, level-1)
        left(60)
        side(length/3, level-1)
        right(120)
        side(length/3, level-1)
        left(60)
        side(length/3, level-1)

speed('fastest')
penup()
goto(-100,100)
pendown()
snowflake(200,4)
done()
```

test if we've bottomed out.