COMP10001 Foundations of Computing
Semester 2 2014
Lecture 12 (advanced lecture, not examinable)

Lindenmayer Systems

Bernie Pope, bjpope@unimelb.edu.au

Outline

- Turtle Graphics
- Lindenmayer Systems

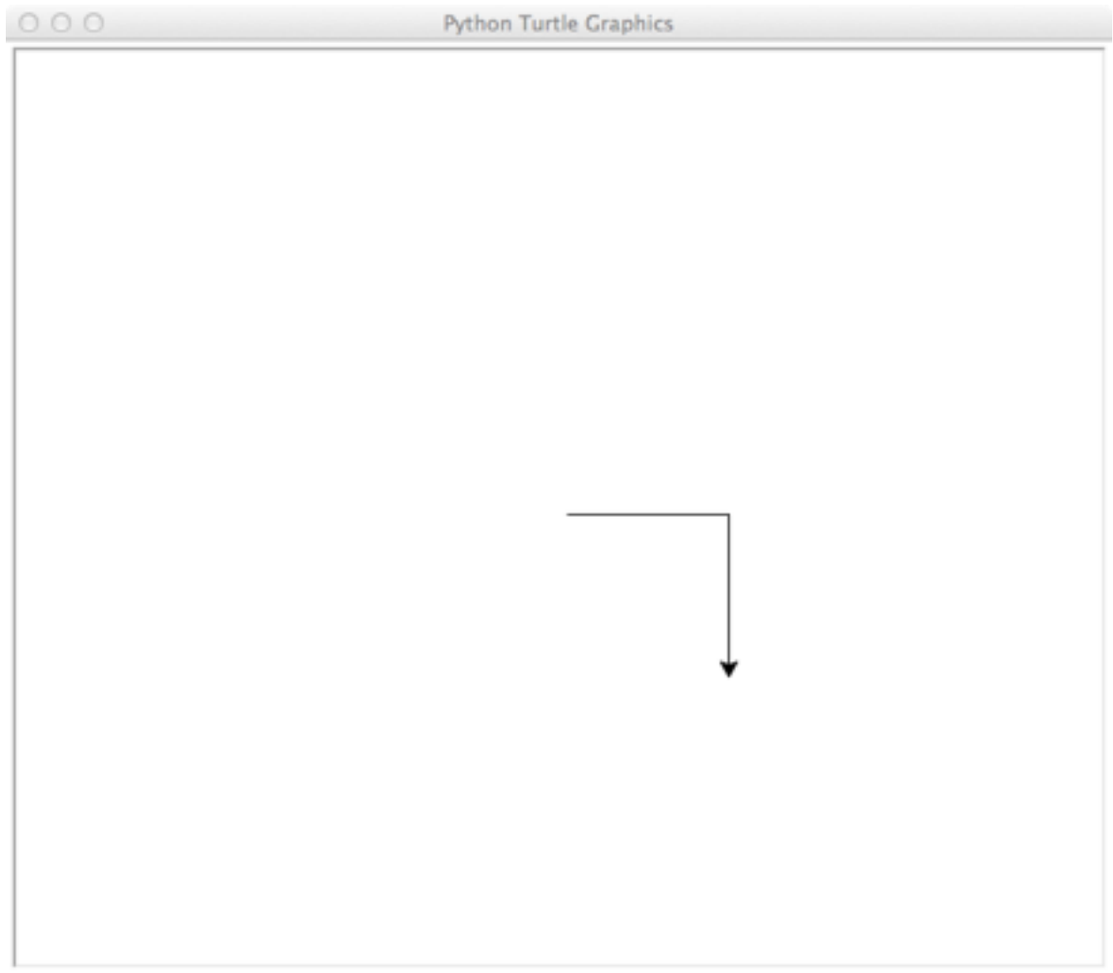
http://en.wikipedia.org/wiki/Turtle

Turtle graphics in Python

```
>>> from turtle import *
>>> forward(100)
>>> right(90)
>>> forward(100)
```

- This does not work in IVLE - if you want to try this out you will need to install Python on your own computer.

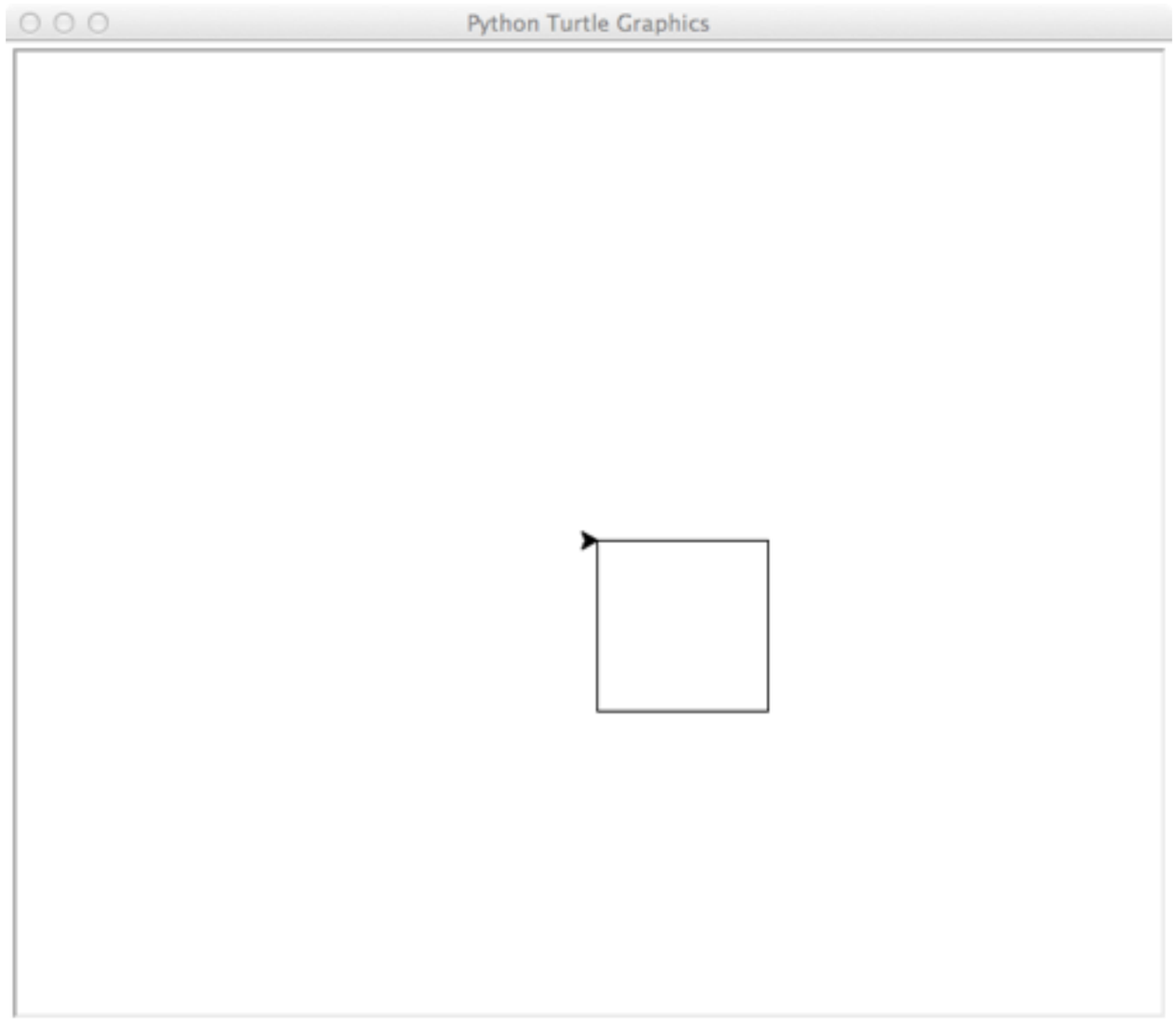
Turtle graphics in Python



Turtle graphics in Python

```
>>> reset()
>>> def square(side):
...     for _repeat in range(4):
...         forward(side)
...         right(90)
...
>>> square(100)
```

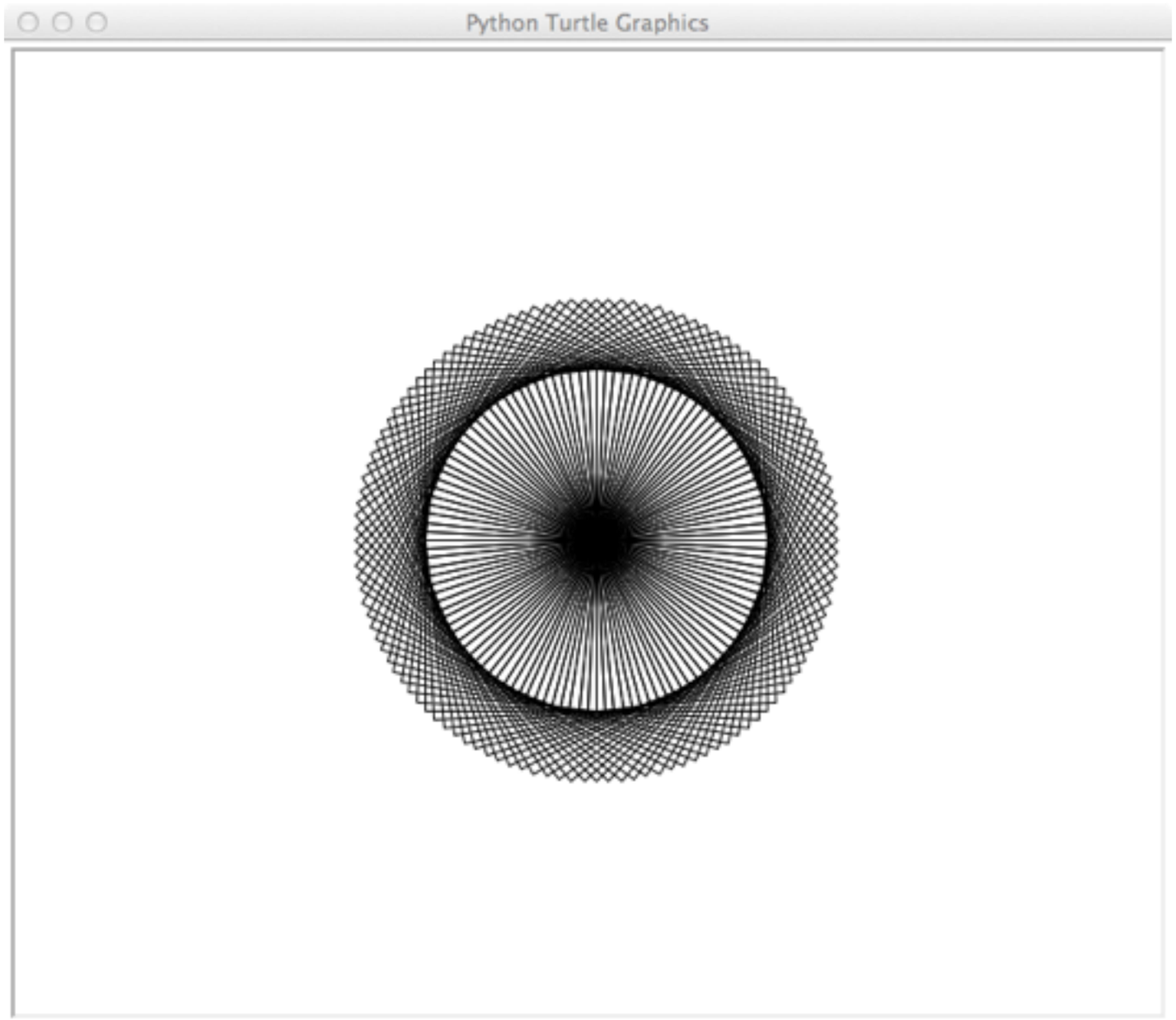
Turtle graphics in Python



Turtle graphics in Python

```
>>> reset()
>>> speed(0) # make the turtle go fast
>>> for _repeat in range(120):
...     square(100)
...     right(3)
...
```

Turtle graphics in Python



Turtle graphics in Python

Some turtle commands

```
goto(x, y)

penup()

pendown()

left(angle)

right(angle)

back(distance)

forward(distance)

speed(n)
```

```
heading()

setheading()

pos()

setpos(x,y)

hideturtle()

showturtle()

reset()

home()
```

http://docs.python.org/library/turtle.html

Lindenmayer Systems

- Lindenmayer Systems (L-Systems) are a kind of **term rewriting system**.
- Example rules:

$$A \rightarrow AB$$

$$B \rightarrow A$$

Lindenmayer Systems

- We pick a start symbol and iterate the rules for N steps.
- For example, suppose the start symbol is A.

iteration	term
0	A
1	AB
2	ABA
3	ABAAB
4	ABAABABA

Lindenmayer Systems

```
def example_ab(n):
    result = 'A'
    for _repeat in range(n):
        next = ''
        for char in result:
            # replace A with AB
            if char == 'A':
                next += 'AB'
            # replace B with A
            elif char == 'B':
                next += 'A'
            # leave other characters unchanged
            else:
                next += char
    result = next
return result
```

Lindenmayer Systems

```
>>> example_ab(0)
'A'
>>> example_ab(1)
'AB'
>>> example_ab(2)
'ABA'
>>> example_ab(3)
'ABAAB'
>>> example_ab(4)
'ABAABABA'
>>> len(example_ab(20))
17711
```

Lindenmayer Systems

```
def rule_a(n):
    if n == 0:
        return 'A'
    else:
        return rule_a(n-1) + rule_b(n-1)

def rule_b(n):
    if n == 0:
        return 'B'
    else:
        return rule_a(n-1)
```

A recursive implementation of the rules.

Lindenmayer Systems

```
>>> rule_a(0)
'A'
>>> rule_a(1)
'AB'
>>> rule_a(2)
'ABA'
>>> rule_a(3)
'ABAAB'
>>> rule_a(4)
'ABAABABA'
>>> len(rule_a(20))
17711
```

Lindenmayer Systems

- Another rule set: F → FF-[-F+F+F]+[+F-F-F]

```
def rule_f(n):
    if n == 0:
        return 'F'
    else:
        next_f = rule_f(n-1)
        return ''.join([next_f, next_f,
            '-', '[', '-', next_f, '+',
            next_f, '+', next_f, ']',
            '+', '[', '+', next_f, '-',
            next_f, '-', next_f, ']'])
```

Lindenmayer Systems

```
>>> rule_f(0)
'F'
>>> rule_f(1)
'FF-[-F+F+F]+[+F-F-F]'
>>> len(rule_f(2))
172
>>> len(rule_f(5))
88940
>>> len(rule_f(6))
711532
```

Lindenmayer Systems

- So what is this all about?

Lindenmayer Systems

- We can interpret the symbols in the generated string as commands for the turtle:
    - `F` means go forward
    - `+` means go right
    - `-` means go left
    - `[` means remember the current position and orientation
    - `]` means return to the most recently remembered position and orientation

Lindenmayer Systems

```
def interpret(lsystem_str, angle, distance):
    mem = []
    for char in lsystem_str:
        if char == '+':
            right(angle)
        elif char == '-':
            left(angle)
        elif char == 'F':
            forward(distance)
        elif char == '[':
            mem.append((pos(), heading()))
        elif char == ']':
            old_pos, old_dir = mem.pop()
            penup()
            setpos(old_pos[0], old_pos[1])
            setheading(old_dir)
            pendown()
```

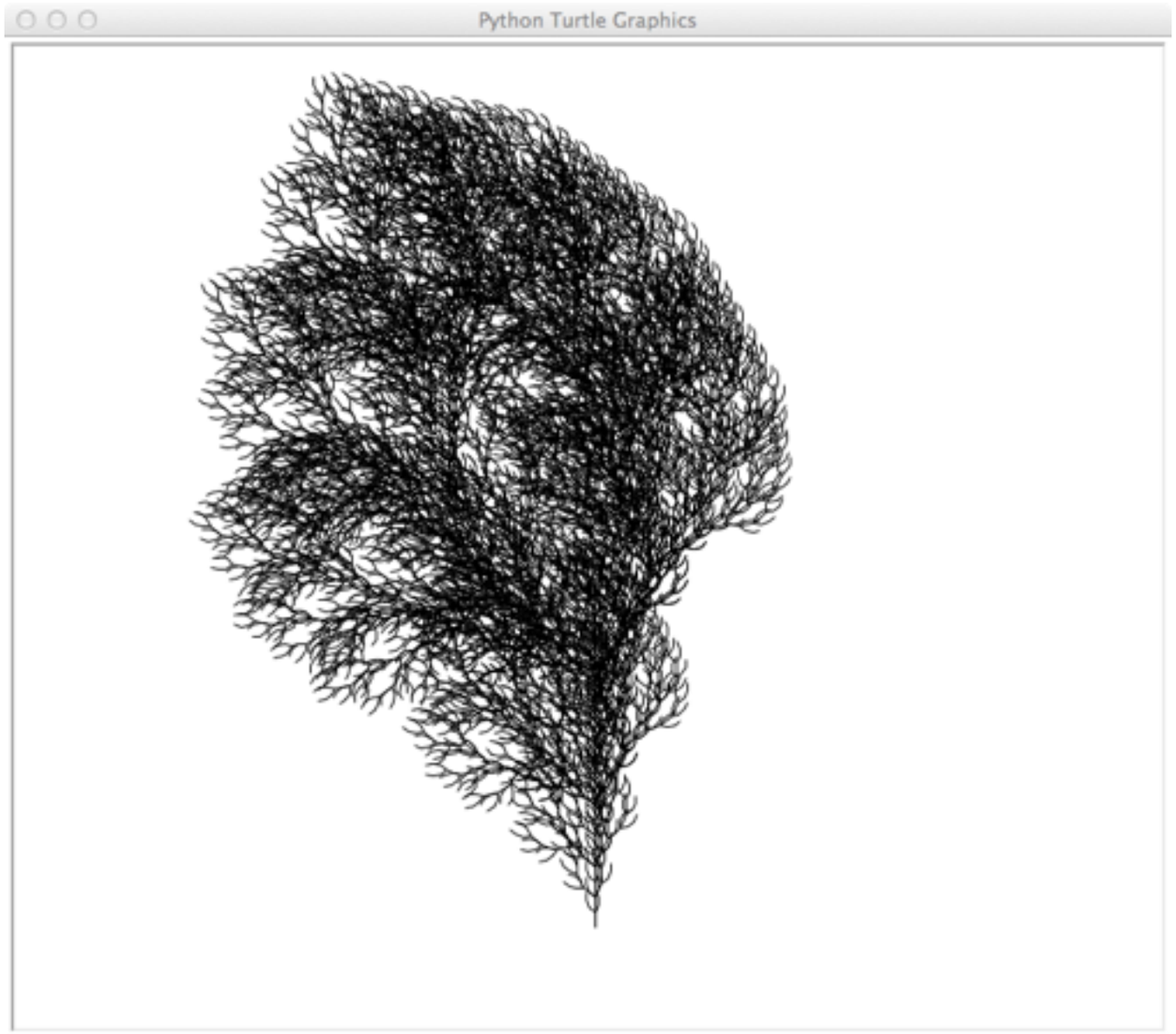
Lindenmayer Systems

```
def init():
    reset()         # reset turtle
    penup()         # lift up the pen
    goto(0,-250)  # move to starting point
    tracer(50,0)  # speed up drawing
    speed(0)        # speed up drawing
    hideturtle()  # speed up drawing
    left(90)        # face upwards
    pendown()       # put the pen down
```

Lindenmayer Systems

```
>>> from lsystem import rule_f, init, interpret
>>> init()
>>> term = rule_f(5)
>>> interpret(term, 25, 5)
```

Lindenmayer Systems



Lindenmayer Systems

- More examples:

  http://www.berniepope.id.au/lsystem.html