



# Python 3 – Turtle graphics

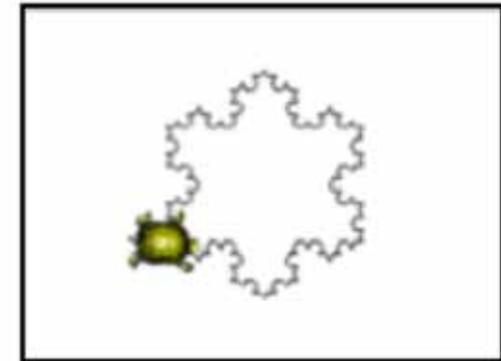
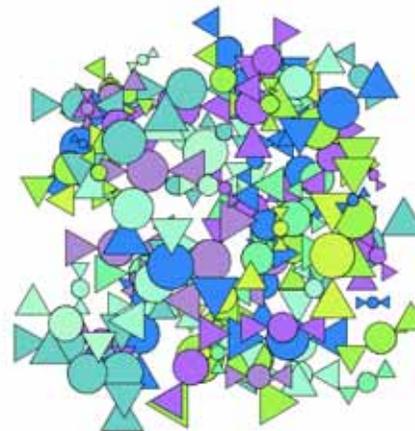
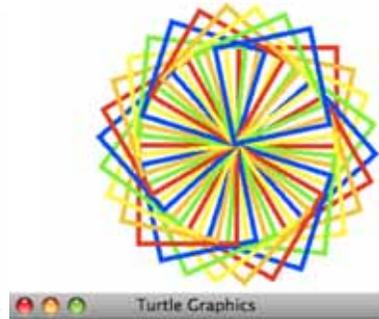
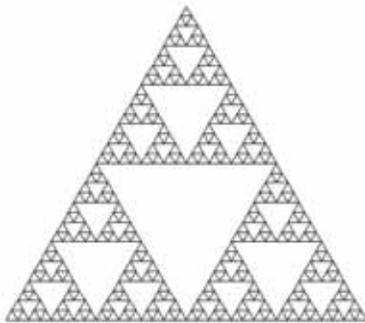
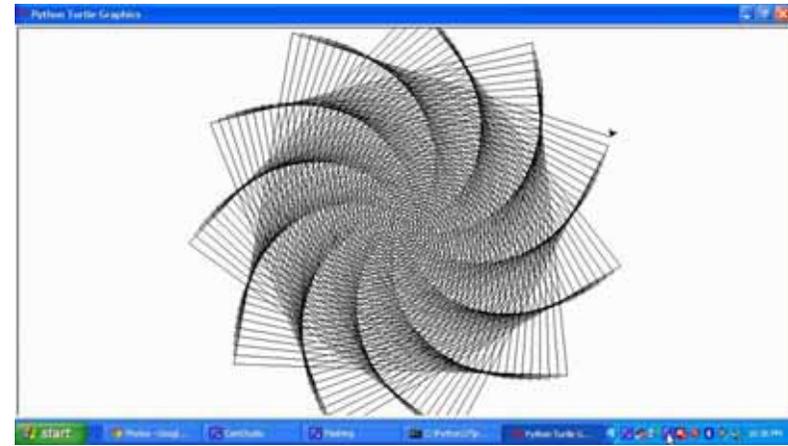
Lecture 25 – COMPSCI111/111G SS 2018





# Today's lecture

- ▶ The Turtle graphics package
  - ▶ Brief history
  - ▶ Basic commands
  - ▶ Drawing shapes on screen





# Logo and Turtle graphics

---

- ▶ In 1967, Seymour Papert and Wally Feurzeig created an interpretive programming language called Logo.
- ▶ Papert added commands to Logo so that he could control a turtle robot, which drew shapes on paper, from his computer
- ▶ Turtle graphics is now part of Python
- ▶ Using the Turtle involves instructing the turtle to move on the screen and draw lines to create the desired shape



# The Turtle package

---

- ▶ Some functions are part of Python's core libraries, in other words they are 'built-in'
  - ▶ `print()`
  - ▶ `input()`
  - ▶ `float()`
- ▶ Other functions need to be imported into your Python program
- ▶ The `turtle` module needs to be imported at the start of any Python program that uses it:  
`import turtle`



# Basic Turtle commands

---

- ▶ There are four basic turtle commands
- ▶ `turtle.forward(x)`
  - ▶ Moves turtle forward in direction it is facing by  $x$  steps
- ▶ `turtle.back(x)`
  - ▶ Moves turtle backward from its facing direction by  $x$  steps
- ▶ `turtle.left(x)`
  - ▶ Turns the turtle  $x$  degrees counterclockwise
- ▶ `turtle.right(x)`
  - ▶ Turns the turtle  $x$  degrees clockwise





# Turtle example

- ▶ Using the Python interpreter in IDLE to demonstrate how to use Turtle graphics
- ▶ First, import the `turtle` package

A screenshot of a Python Shell window. The title bar reads 'Python Shell'. The menu bar includes 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The main text area contains the following code:

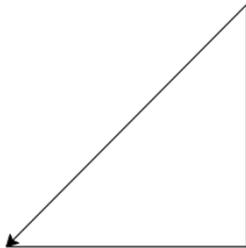
```
>>> import turtle
>>>
```

The status bar at the bottom right shows 'Ln: 12 | Col: 4'.

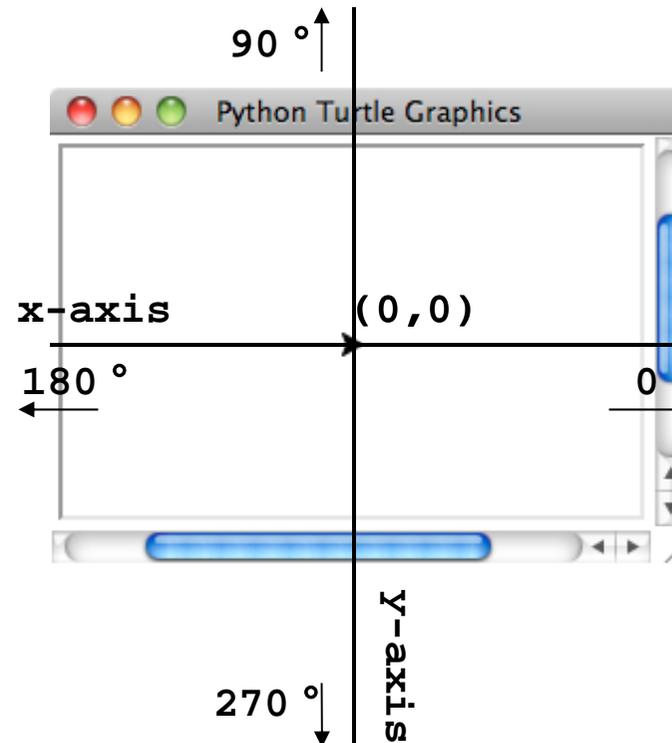


# Turtle example

- ▶ We are going to draw a right-angled triangle

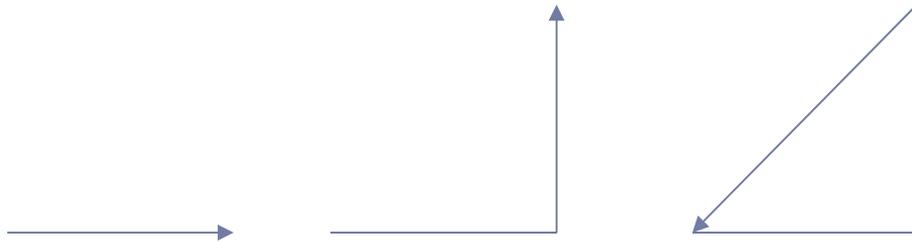
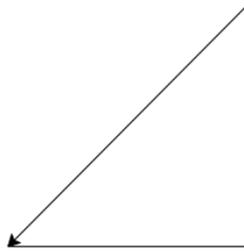


- ▶ Important information:
  - ▶ The turtle appears as an icon
  - ▶ Initial position:  $(0, 0)$
  - ▶ Initial direction: East ( $0^\circ$ )
  - ▶ Colour: black
  - ▶ Line width: 1 pixel
  - ▶ Pen: down (ready to draw)





# Algorithm



draw a line

Turn 90 degrees left (anti-clockwise)

draw a line

Turn 135 degrees left (anti-clockwise)

draw a line



# Turtle example

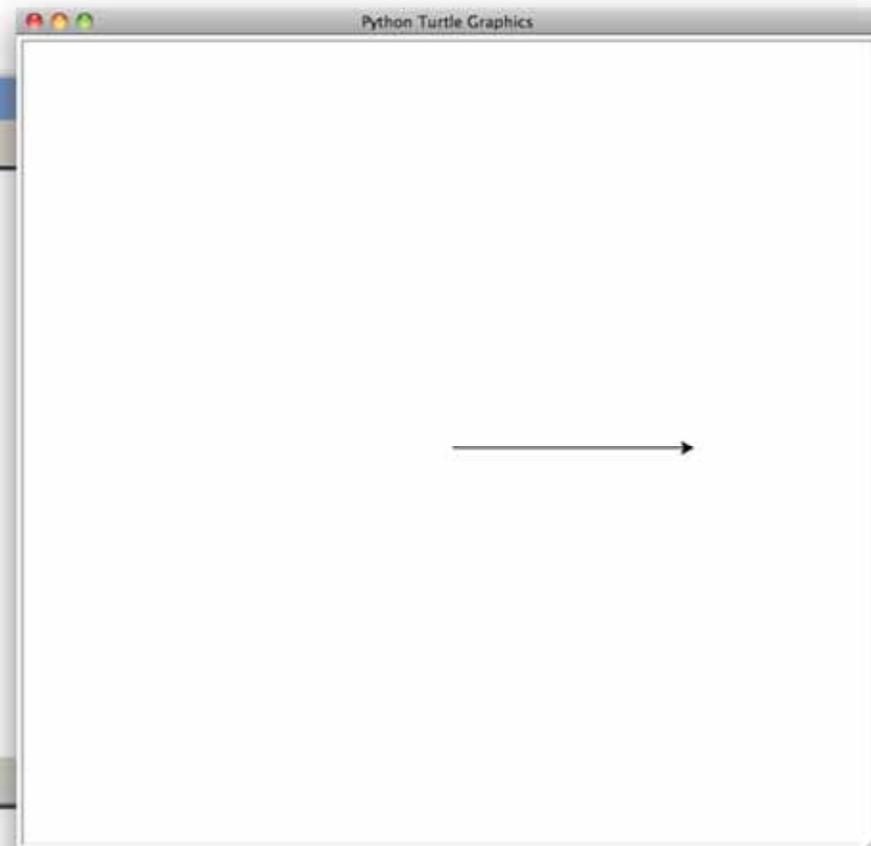


- ▶ Step 1: Draw a line



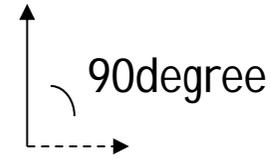
```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import turtle
>>>
>>> turtle.forward(200)
>>>
```

1. Draw a line





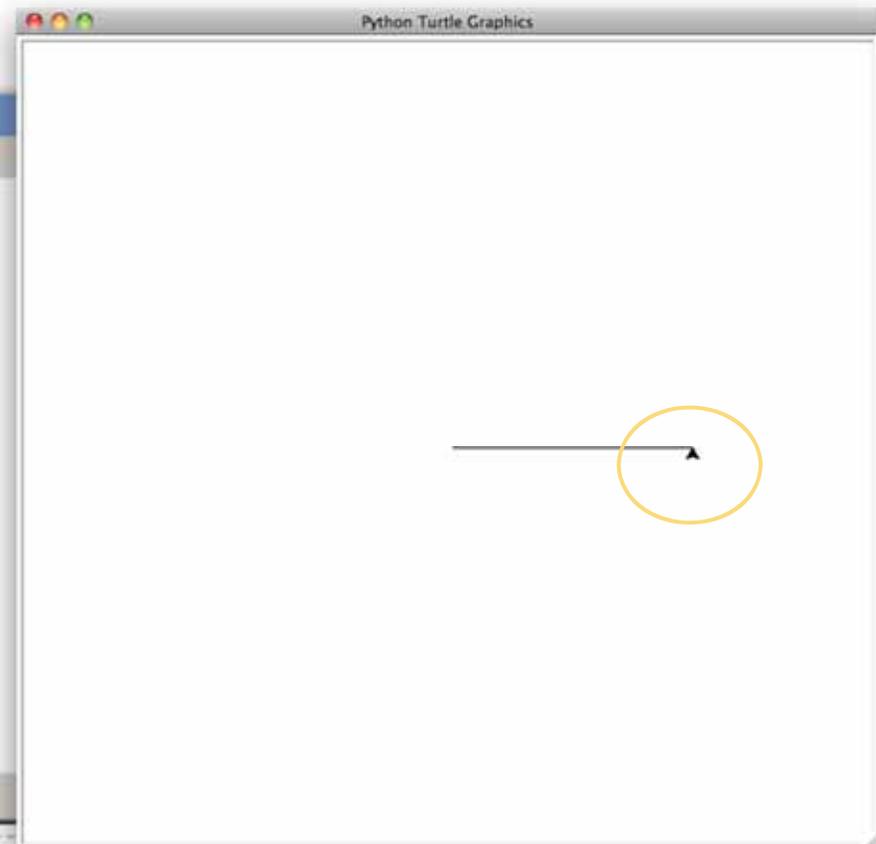
# Turtle example



Initial direction: 0

- ▶ Note how the turtle is now facing upward after being turned 90 degrees left

```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import turtle
>>>
>>> turtle.forward(200)
>>> turtle.left(90)
>>>
```

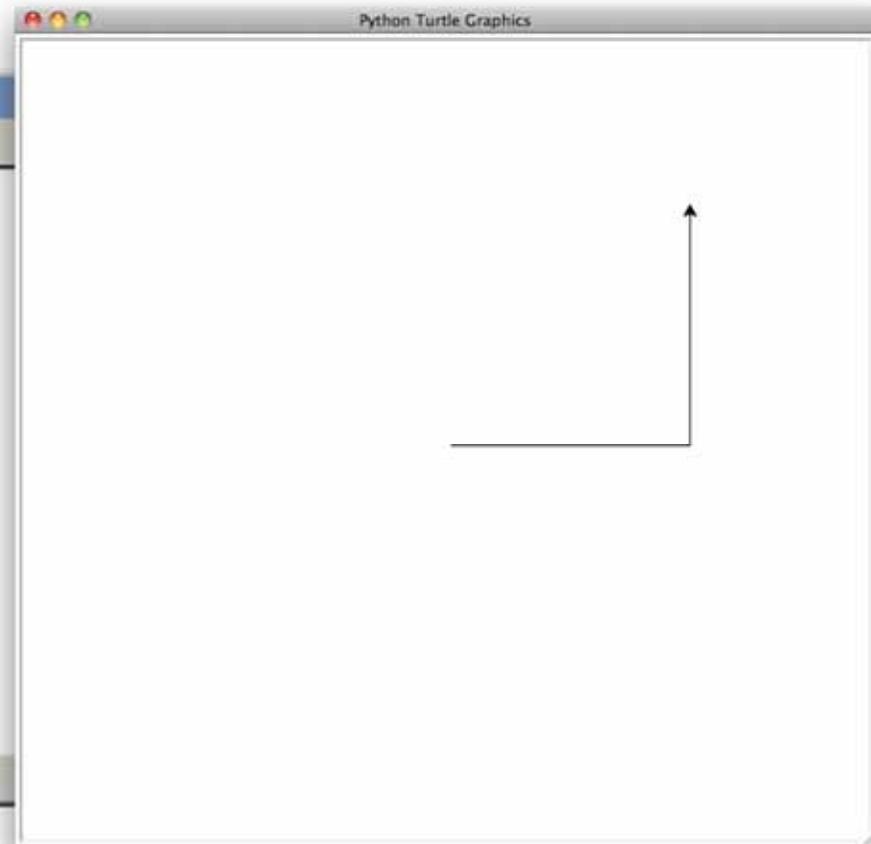




# Turtle example

- ▶ Step 3: draw a line

```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import turtle
>>>
>>> turtle.forward(200)
>>> turtle.left(90)
>>> turtle.forward(200)
>>>
```



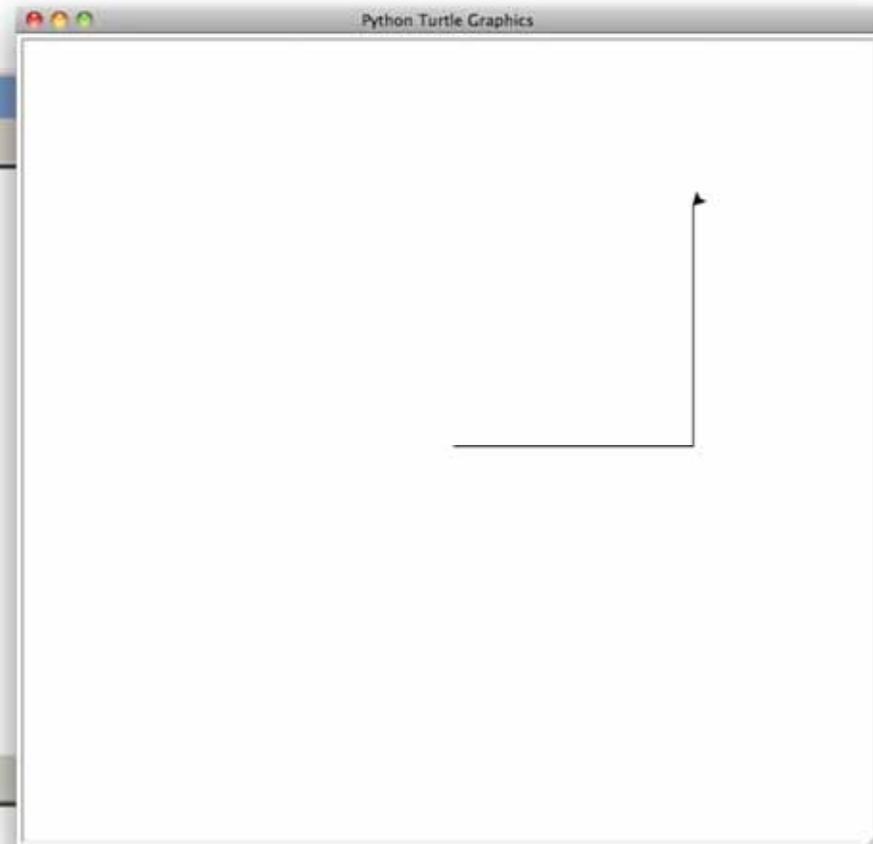


# Turtle example

current direction  
135degree

- ▶ Step 4: turn 135 degree left (anti-clockwise)

```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import turtle
>>>
>>> turtle.forward(200)
>>> turtle.left(90)
>>> turtle.forward(200)
>>> turtle.left(135)
>>>
```





# Turtle example

- ▶ Working out the length of the longest side using the Pythagoras' formula

```
Python Shell
File Edit Shell Debug Options Windows Help
>>> import turtle
>>>
>>> turtle.forward(200)
>>> turtle.left(90)
>>> turtle.forward(200)
>>> turtle.left(135)
>>> c = ((200**2)+(200**2))**0.5 #around 283 steps
Ln: 12 Col: 4
```



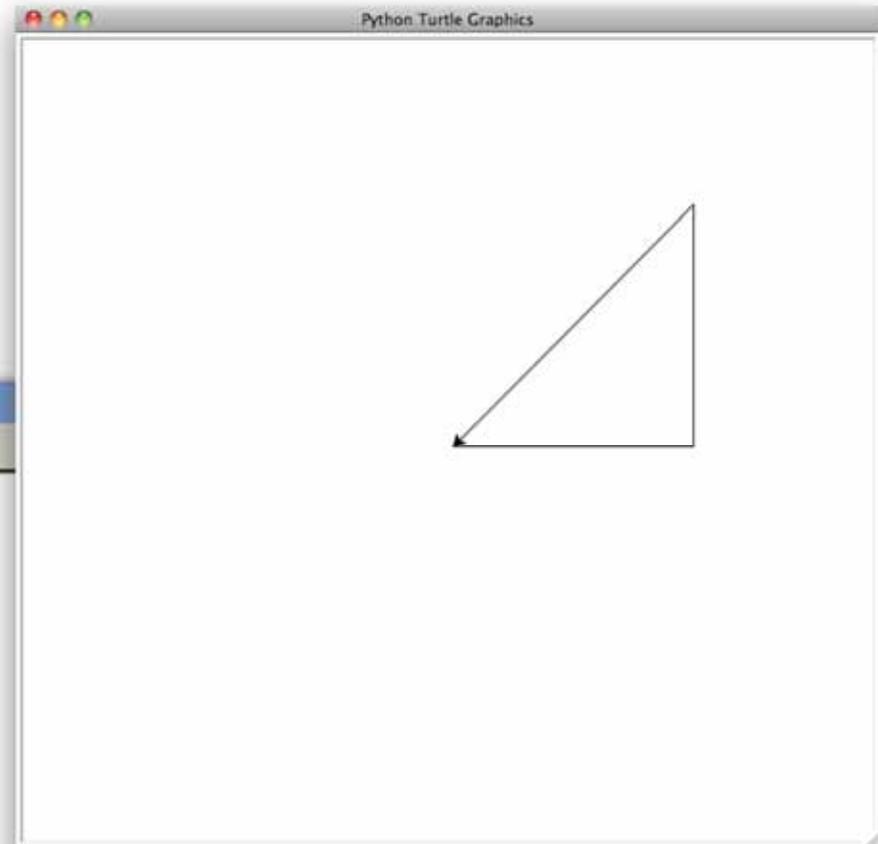
# Turtle example

- ▶ Step 6: draw a line
- ▶ The finished image

```
Python Shell
File Edit Shell Debug Options Windows Help

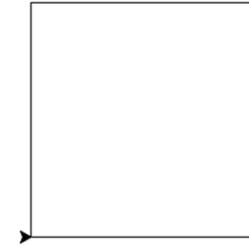
>>> import turtle
>>>
>>> turtle.forward(200)
>>> turtle.left(90)
>>> turtle.forward(200)
>>> turtle.left(135)
>>> c = ((200**2)+(200**2))**0.5
>>> turtle.forward(c)
```

Ln: 12 Col: 4

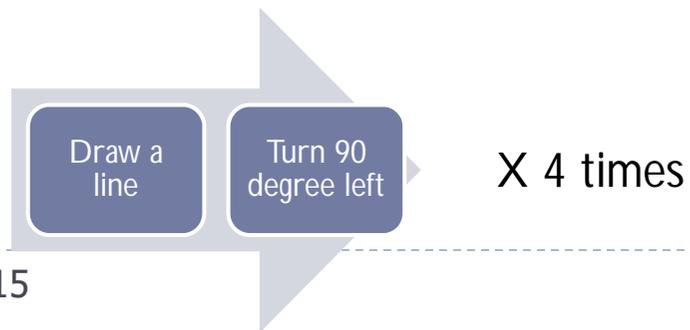
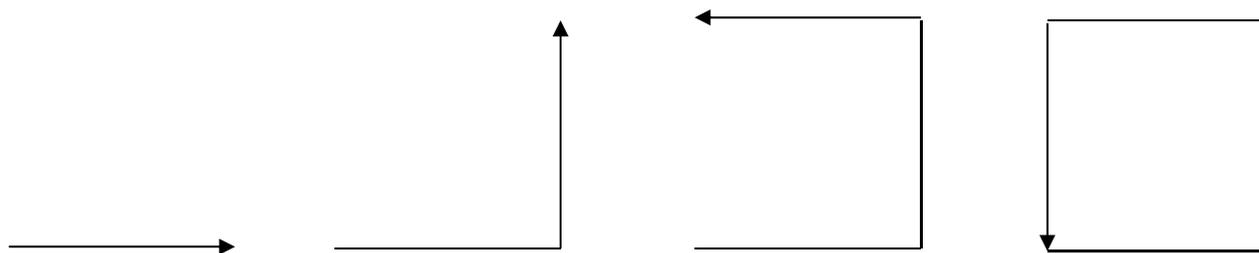




# Turtle example



- ▶ We can use loops when drawing shapes using Turtle graphics
- ▶ Write a program that will draw a square using a loop





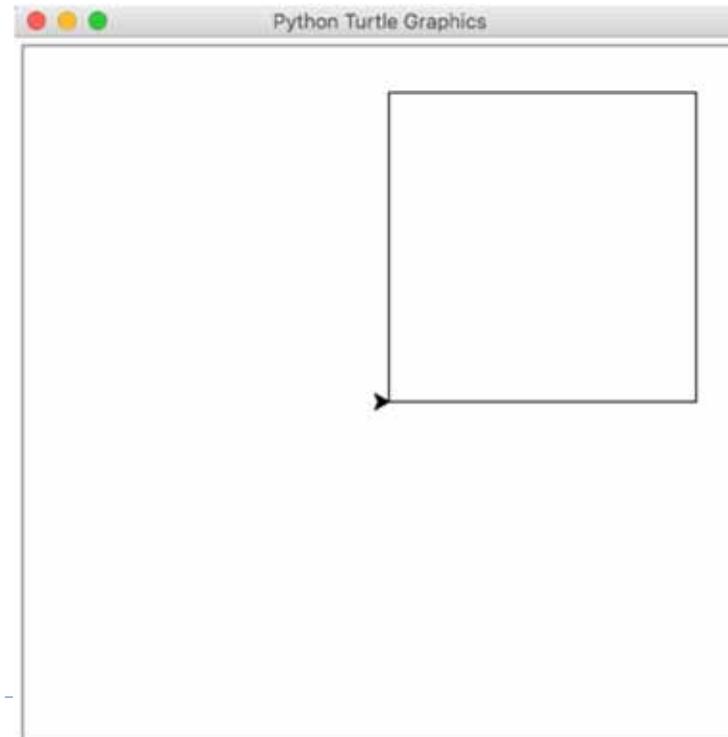
# Turtle example



- ▶ We can use loops when drawing shapes using Turtle graphics
- ▶ Write a program that will draw a square using a loop

```
import turtle

count = 0
while count < 4:
    turtle.forward(200)
    turtle.left(90)
    count = count + 1
```

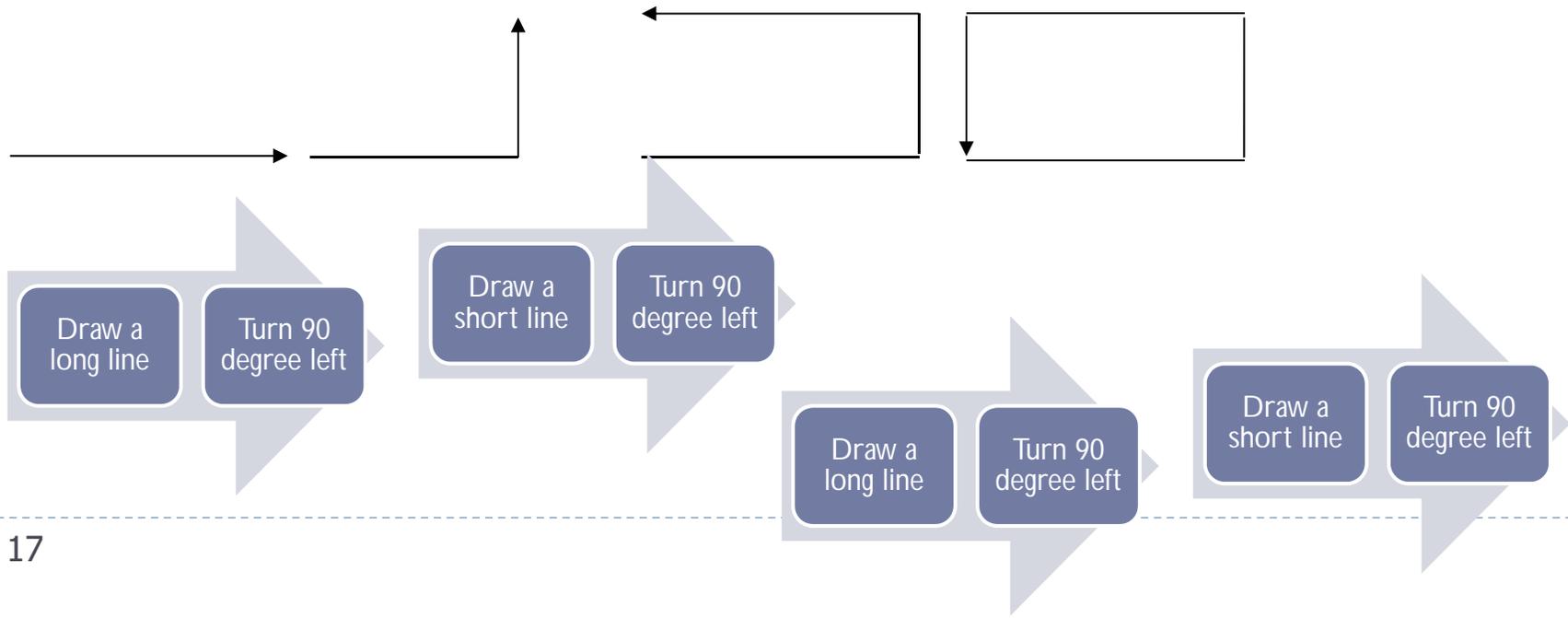
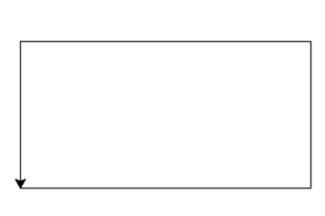




# Exercise 1

TRY IT OUT!

- Write a Python program that draws a rectangle. The long sides must be 300 steps long and the short sides must be 150 steps long



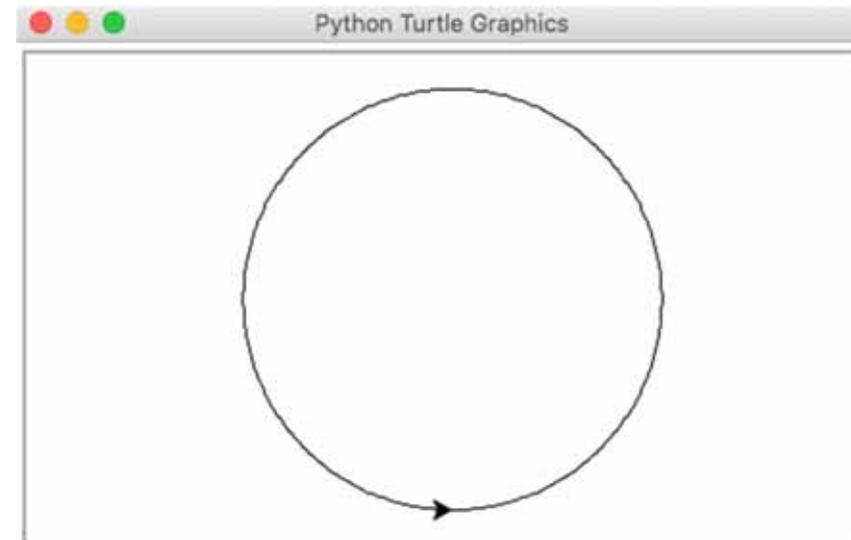


# Turtle example



- ▶ Write a program that will draw a circle

- ▶ Steps:
  - ▶ Draw a short line (2 pixels)
  - ▶ Turn 1 degree
  - ▶ Repeat the above steps 360 times



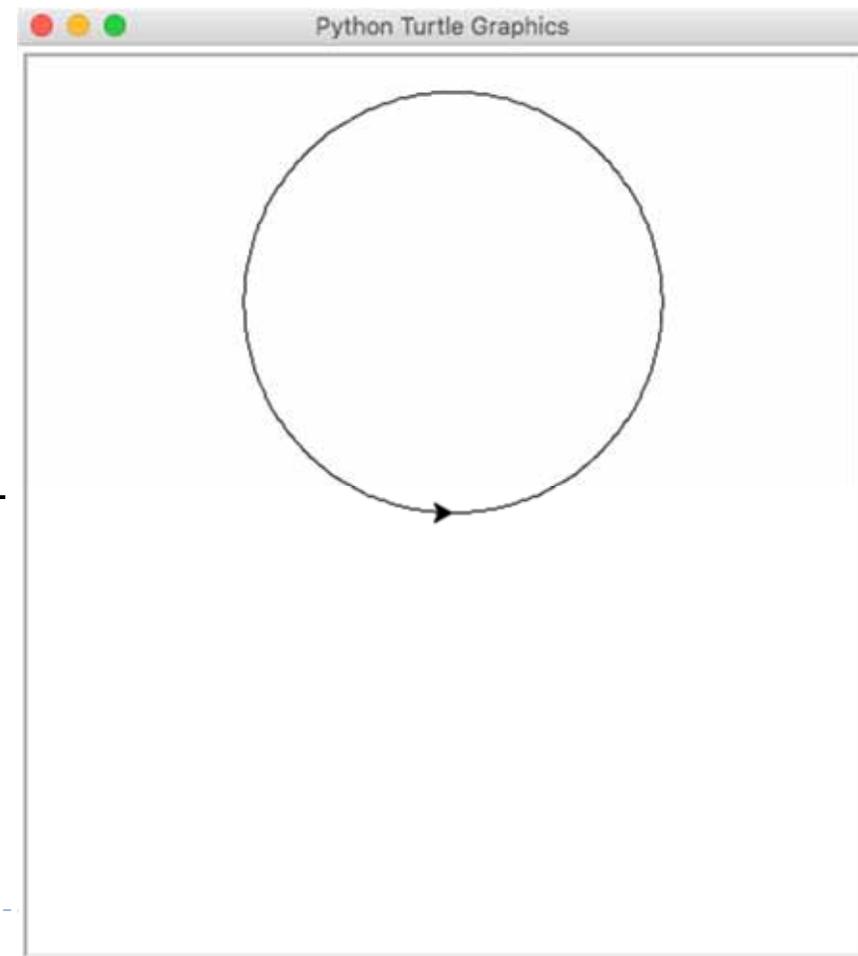


# Turtle example



- ▶ Write a program that will draw a circle

```
import turtle
count = 0
while(count < 360):
    turtle.forward(2)
    turtle.left(1)
    count = count + 1
print("Finished!")
```

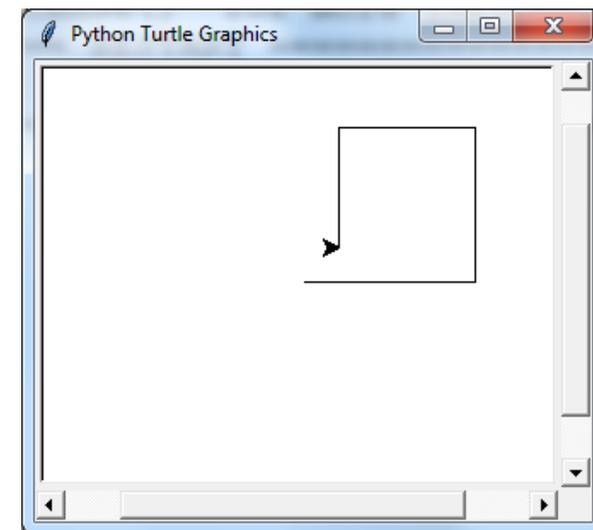
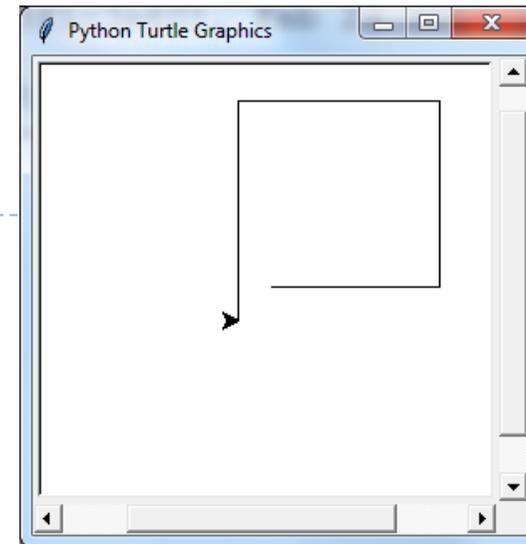




## Question

- ▶ Consider the following program:

```
import turtle
count = 0
length = 100
while count < 4:
    turtle.forward(length)
    turtle.left(90)
    count = count + 1
    length = length - 10
```

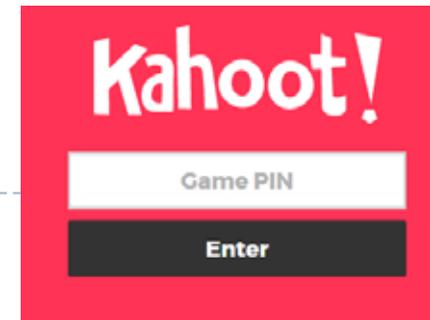


- ▶ Which of the following pictures demonstrates the output generated by the program above?

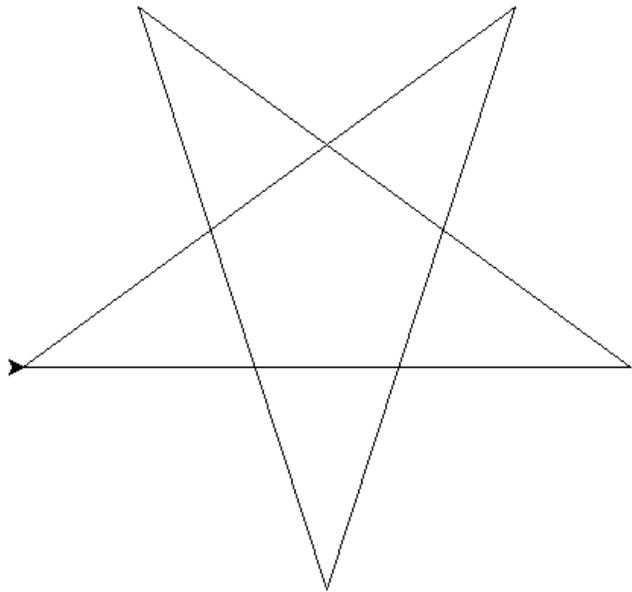


## Exercise 2

---



- ▶ How to draw a star?
  - ▶ How many steps do you need?
  - ▶ What is the size/length for each step? E.g. 400 pixels
  - ▶ What is the turning angle for each step?





## Exercise 3

TRY IT OUT!

- ▶ Draw the shape that is produced by the following Python program:

```
import turtle
count = 0
while(count < 180):
    turtle.forward(2)
    turtle.right(1)
    count = count + 1
turtle.right(45)
turtle.forward(300)
turtle.left(90)
turtle.back(150)
turtle.right(45)
turtle.back(250)
```



## Exercise 4

TRY IT OUT!

- ▶ Draw the shape that is produced by the following Python program:

```
import turtle
big_line = 100
little_line = 50
angle = 90

turtle.left(angle)
turtle.forward(big_line)
count = 0
while count < 4:
    turtle.right(angle//2)
    if count != 3:
        turtle.forward(little_line)
    else:
        turtle.forward(big_line)
    count = count + 1
turtle.right(90)
```

23 turtle.forward(130)



# Summary

---

- ▶ The Turtle package must be imported into every Python program that uses it
- ▶ The Turtle has four basic commands; forward, back, left and right