Lecture 10: Recursion

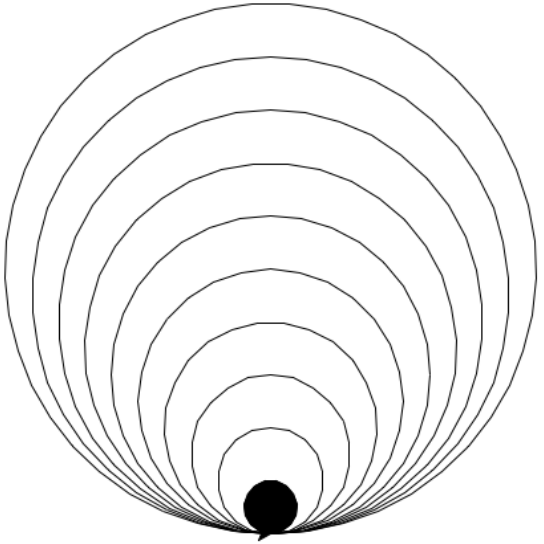
CS 51P

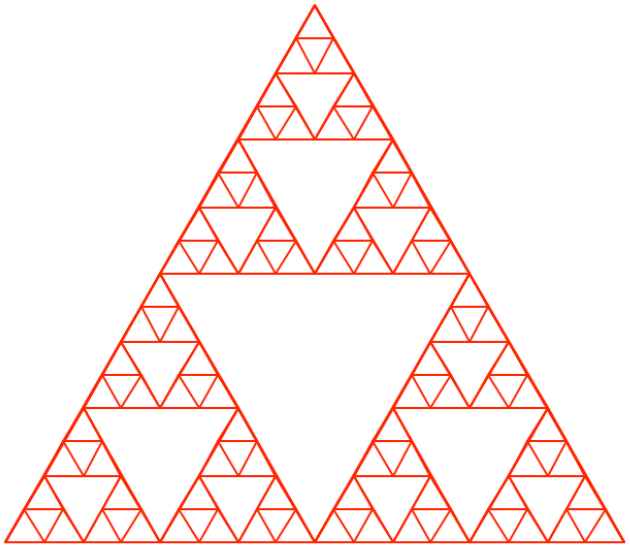
October 9, 2023

Recursion

- Objects containing smaller copies of itself
- Define a base case when an answer can be returned directly, e.g., doesn’t need to recursively call the function itself anymore
- Define a recursive case when the function calls itself with a different (e.g., smaller) arguments
- Solution is built up as you come back up the call stack

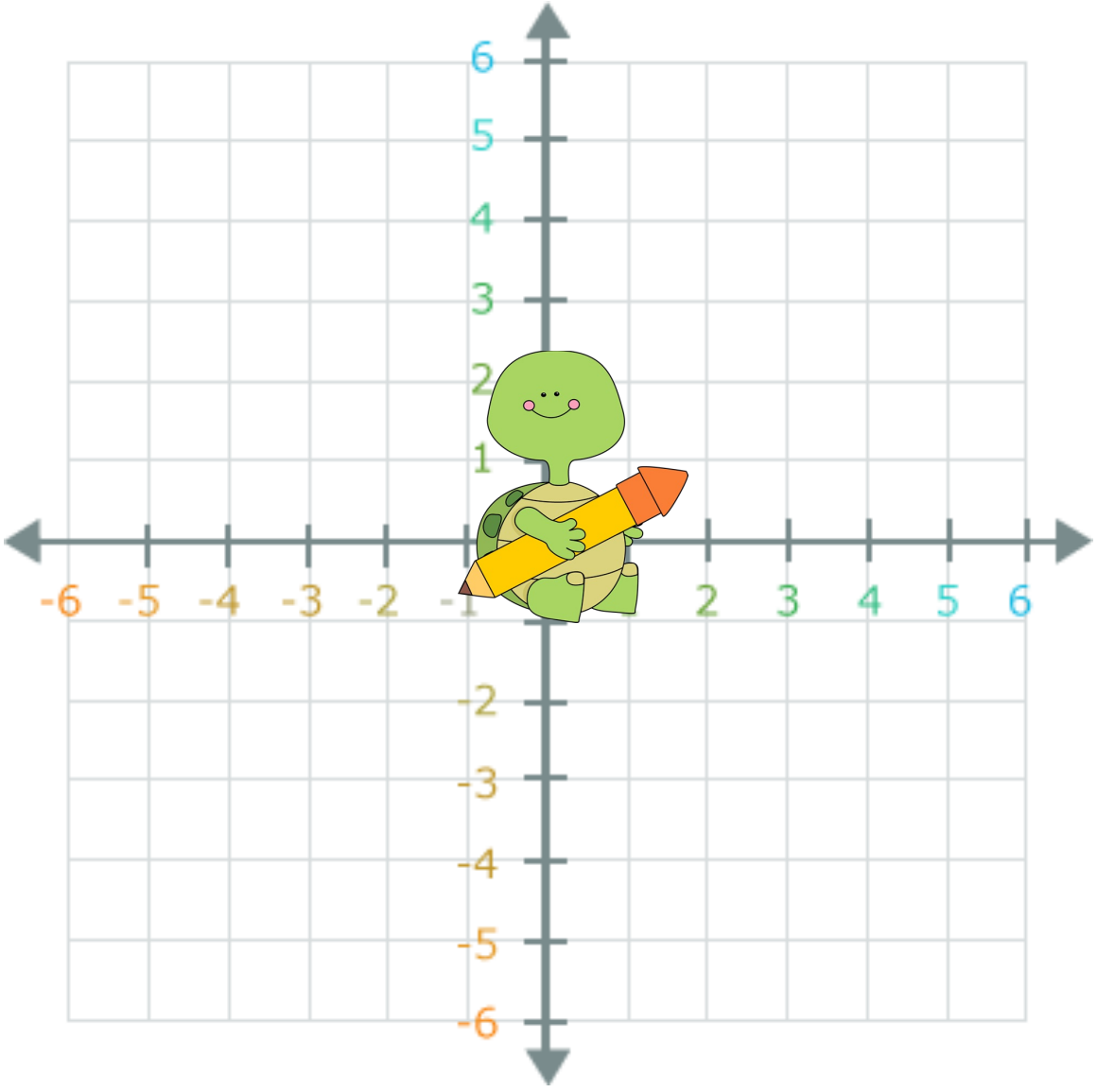
Recursion





1, 1, 2, 3, 5, 8, 13, 21, ...

Turtle graphics

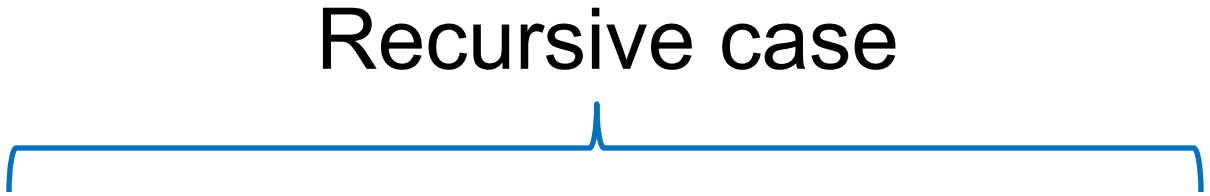


Example

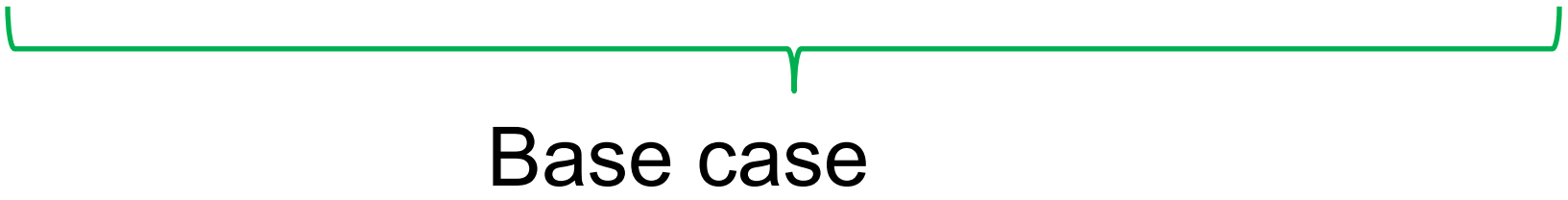
- Define a function called draw_triangle, which takes in the length of a side (e.g., a float) as input, and draws a triangle with equal length of each side.

- Can you create another function to draw a square with length as input?

Recursive definition



- A _x_ is _y_ plus _#_ smaller _x_. unless it is very small, in which case it is _z_.



Example: matryoshka

a _x_ is _y_ plus _#_ smaller _x_. unless it is very small, in which case it is _z_.

- What is a matryoshka?

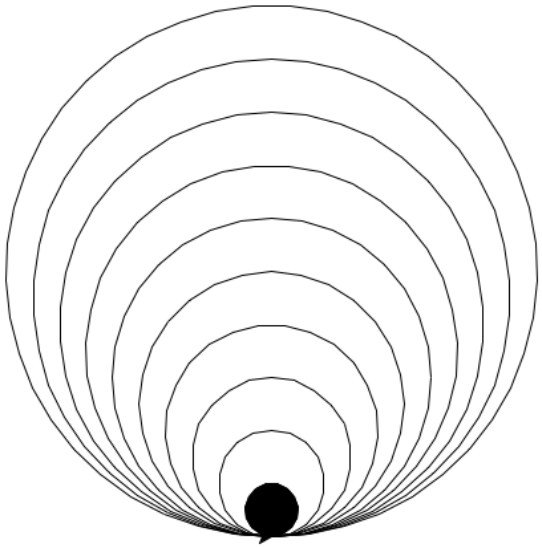
a matryoshka is a doll plus 1 smaller matryoshka. unless it is very small, in which case it is nothing.

Example: circle_drawing

a _x_ is _y_ plus _#_ smaller _x_. unless it is very small, in which case it is _z_.

- What is a circle_drawing?

a circle_drawing is a circle plus 1 smaller circle_drawing. unless it is very small, in which case it is a filled circle



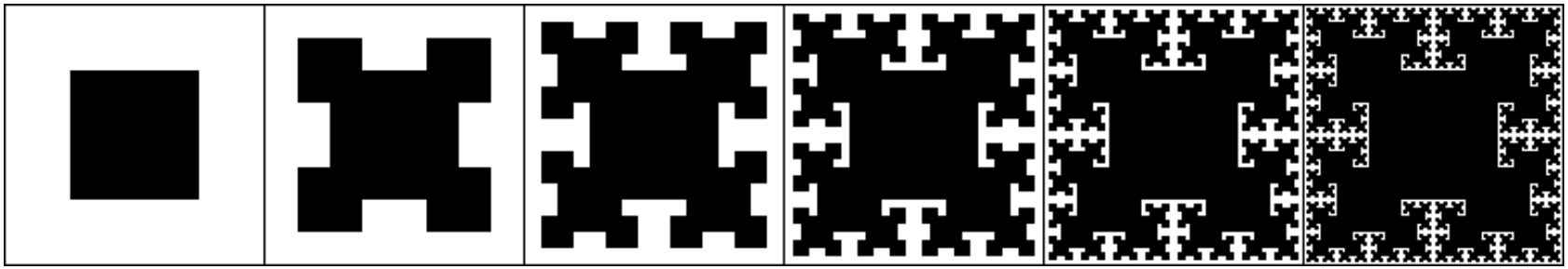
Example

- Draw a set of circles (each circle has radius 20 smaller than the circle outside it), same bottommost point. Once the radius is < 20, draw filled in circle

- Return how many circles are drawn.

Exercise

- a _x_ is _y_ plus _#_ smaller _x_. unless it is very small, in which case it is _z_.
- A recursive_squares is a square
  - plus 4 smaller recursive_squares
    - of half the size
    - centered at each of the corners of the large square
  - unless it's very small in which case it's nothing



Bonus Exercise

- Define a function called `recursive_squares`, which takes in three input parameters, which are `x`, `y`, and `l` (floats). `x` and `y` are the coordinates of the bottom left corner of the big square, and `l` is the length of each side of the square. Recursively draw squares at the four corners of the big square as long as `l` is greater than 10. In each recursion, `l` will be shortened by half.
- Return how many squares are drawn.