Lecture 14

Chapter 11 Recursion

11.1 A Bullseye Class

4.3 Case Study: Drawing a Pyramid

Recursion

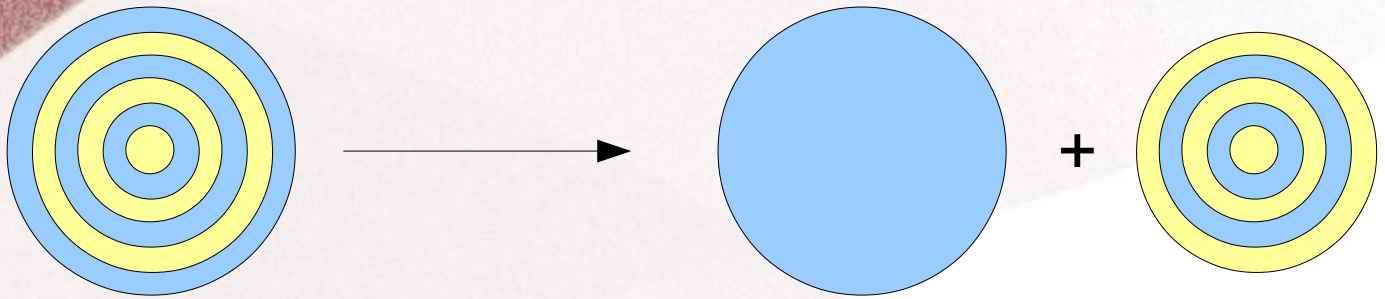
There two forms of recursion:

Structural Recursion (data recursion), and
Functional Recursion

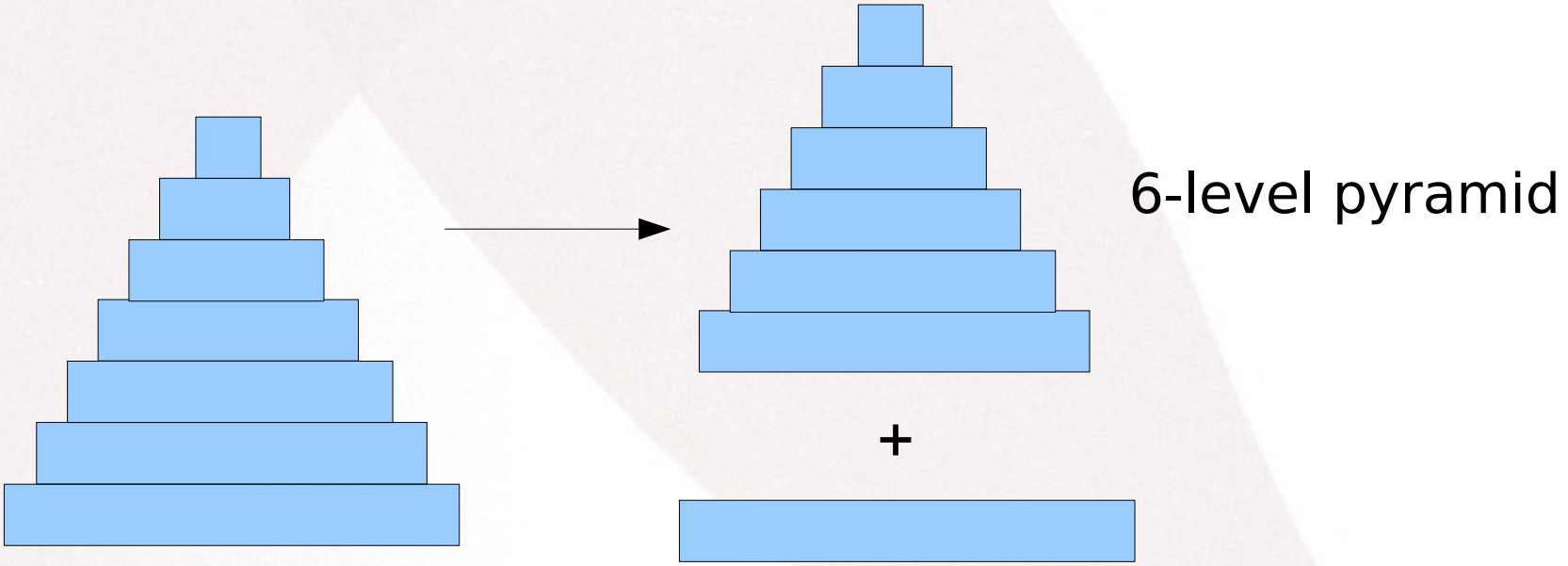
Structural recursion is when objects are defined in terms of other objects of the same type. The entire class of objects can then be built up from a few initial values and a small number of rules.

Example: Pyramid and Bullseye (see on next slide)

Structural Recursion examples



a bullseye



Recursion

There two forms of recursion:

Structural Recursion (data recursion), and

Functional Recursion

Functional recursion is a method of defining functions in which the function being defined is applied within its own definition.

Example: Fibonacci sequence (*recursive definition*):

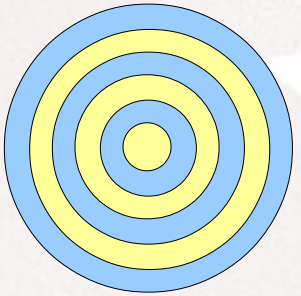
$F(0)=1$ (*base case*)

$F(1)=1$ (*base case*)

$F(n)=F(n-1)+F(n-2)$ for all integers $n>1$

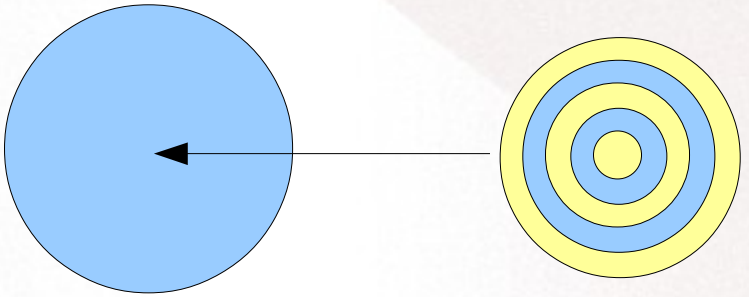
11.1 A Bullseye Class

Let's develop a bullseye class using our graphics library (module). We envision a bullseye as a sequence of concentric circles with alternating colors.



We can create this image using a loop, but let's follow this way: a bullseye can be viewed as a single outer circle with a smaller bullseye drawn on top of it.

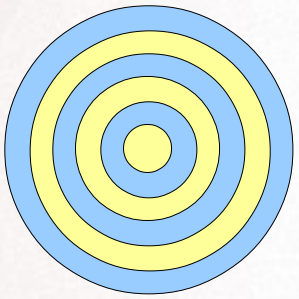
Comment: of course the smaller bullseye should be positioned so that its center is exactly the same as the outer circle.

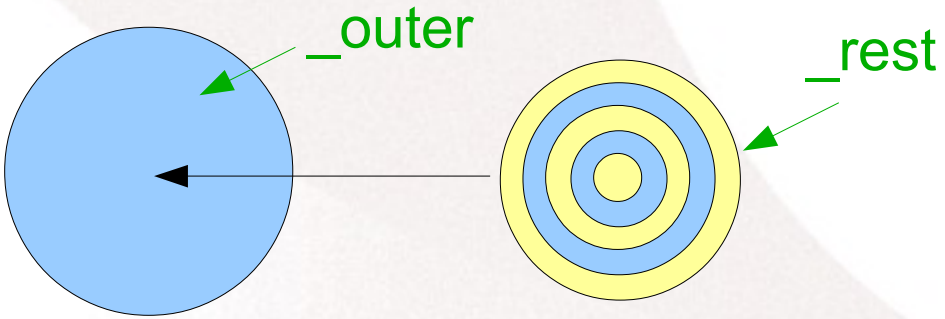


11.1 A Bullseye Class

One of the possible Bullseye Class designs:

Bullseye: FillableShape
_outer _rest
Bullseye(numBands,radius,primary,secondary) setColors(primary,secondary) getNumBands() getRadius() _draw()





11.1 A Bullseye Class

The book gives us a nice top-level trace of recursion and unfolding of the same recursion:

Figures 11.3 on page 364 and **11.4** on page 365.

We will see them after we go over the first sketch of the program: bullseye.py

11.1 A Bullseye Class

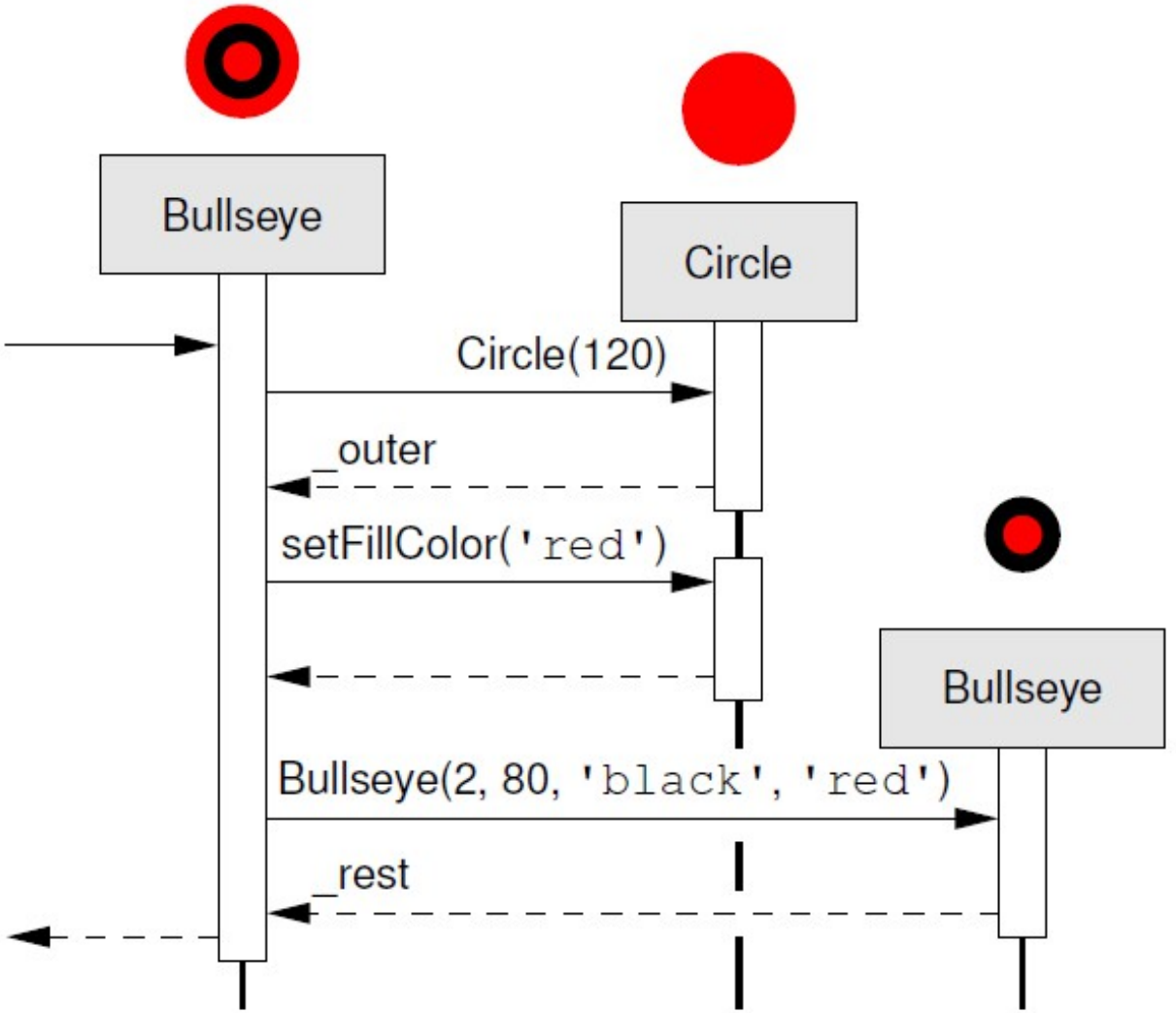
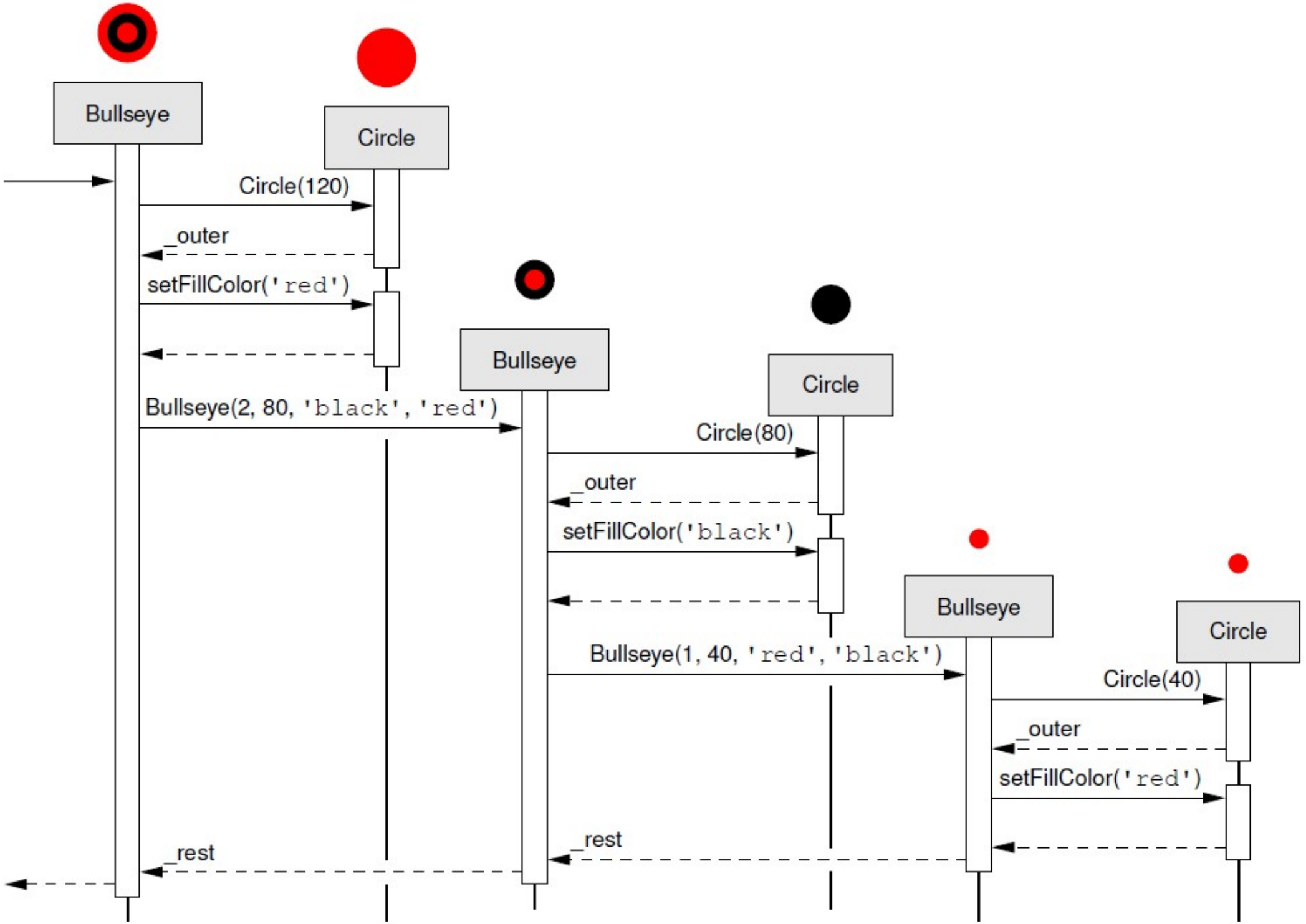


FIGURE 11.3: Top-level trace of Bullseye(3, 120, 'red', 'black')


Bullseye
Circle
Circle(120)
_outer
setFillColor('red')
Bullseye
Circle
Bullseye(2, 80, 'black', 'red')
Circle(80)
_outer
setFillColor('black')
Bullseye
Circle
Bullseye(1, 40, 'red', 'black')
Circle(40)
_outer
setFillColor('red')
_rest
_rest

11.1 A Bullseye Class

Now, let's make it nicer: input is from the graphics window, and exit button.

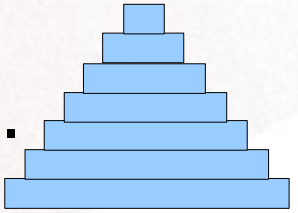
We'll have three handlers:
ExitButtonHandler : handler for exit button

TextHandler: handler for data input
(number of Bands, radius)

ReadyHandler: handler for ready button
(when we are ready to draw a bulls eye,
we'll press the ready button)

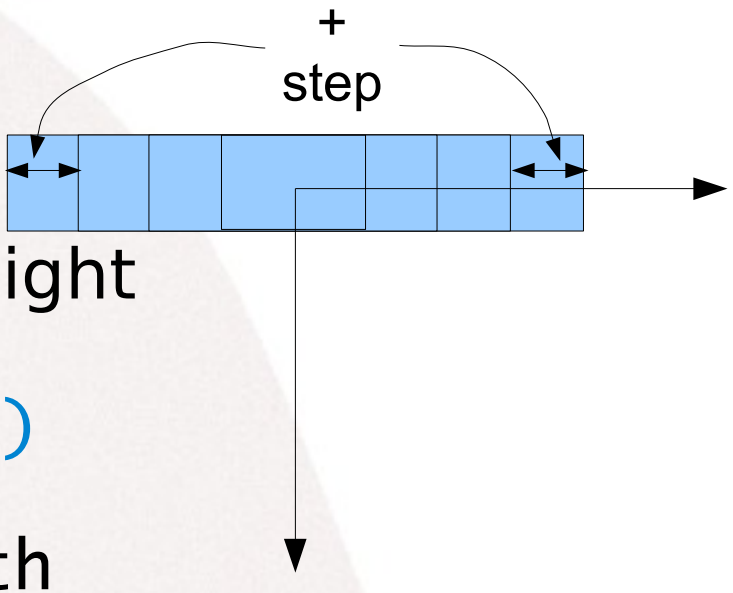
See the program bullseye2.py
(with use of monitors)

4.3 Case Study: Drawing a Pyramid



There are two ways of drawing a pyramid.

One is using a for loop, and another one is using the structural recursion.



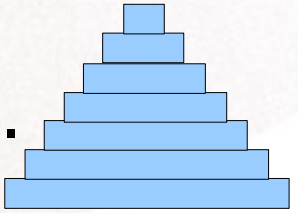
With the for loop:

`base_w, base_h` are width and height

`levels` in the number of levels

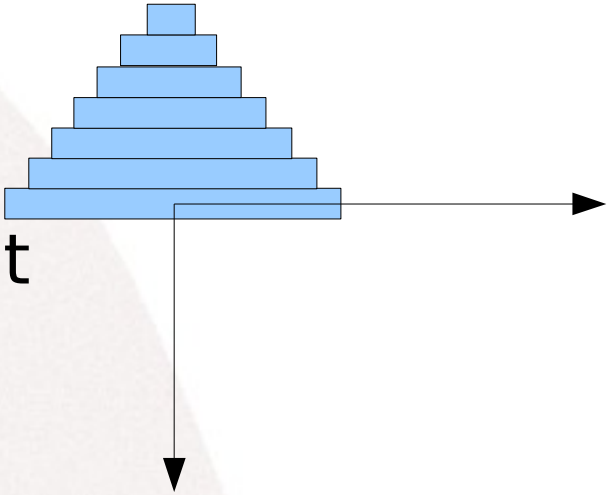
```
step=int(base_w/float(levels))
  for i in range(levels):
    create a rectangle of width
    "previous rectangle's width - step"

    move up (base_h pixels)

    add to the paper
```

4.3 Case Study: Drawing a Pyramid



There are two ways of drawing a pyramid.

One is using a for loop, and another one is using the structural recursion.



With the for loop:

`base_w, base_h` are width and height

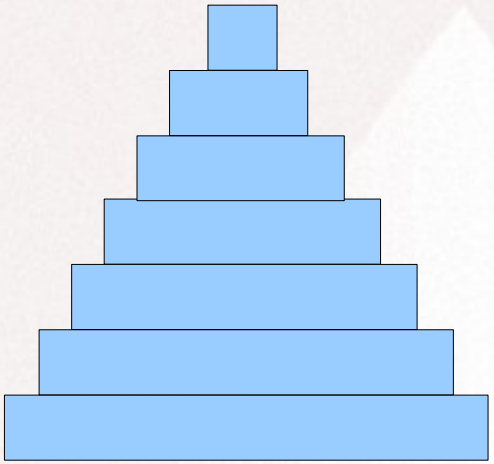
`levels` in the number of levels

```
step=int(base_w/float(levels))
    for i in range(levels):
        Level = Rectangle(base_w-i*step,base_h)
        level.move(paper_w/2,paper_h/2 - i*
                                    base_h)
        paper.add(level)
```

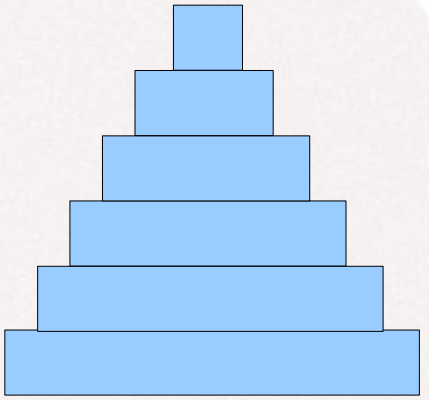
See program pyramid.py

4.3 Case Study: Drawing a Pyramid

Another method: using the structural recursion.
Here is the idea:
A 7-level pyramid can be viewed as a single bottom level with a 6-level pyramid built on top. The 6-level pyramid is itself a bottom level with a 5-level pyramid built on top, and so on.



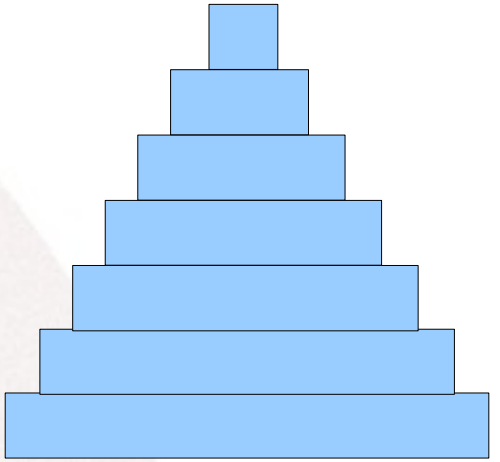
7-level pyramid



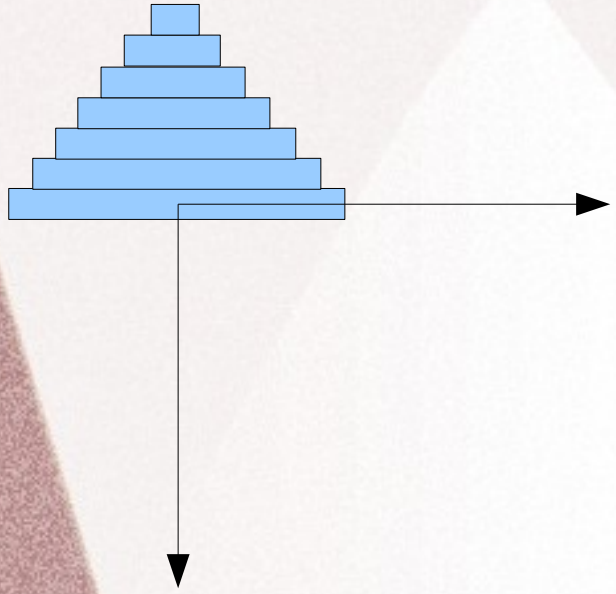
6-level pyramid

bottom level

4.3 Case Study: Drawing a Pyramid



See program pyramid_as_class.py

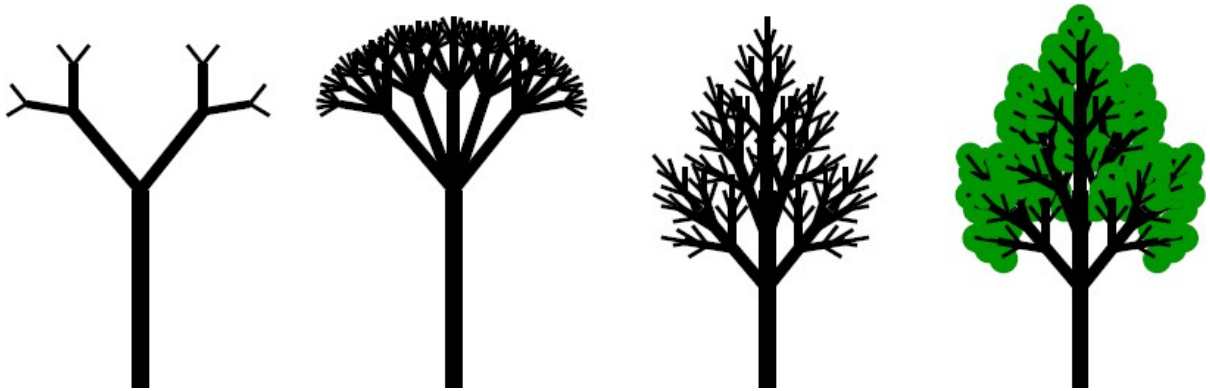


And then, its enhanced version pyramid_as_class2.py

Suggested in-class and home works

- Trees can be “built” using recursion as well

  see exercise 11.6 in the book



- Stars can be nested recursively,

  creating a flower.

  See exercise 11.5