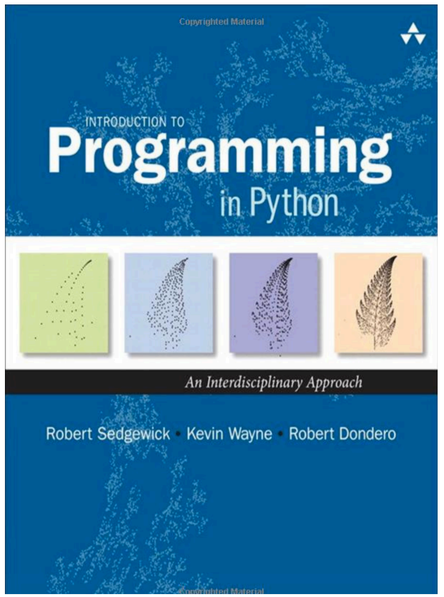
Parte II: Computación científica

Clase 11: Usando tipos de datos 2

Diego Caro
dcaro@udd.cl



Basada en presentaciones oficiales de libro Introduction to Programming in Python (Sedgewick, Wayne, Dondero).

Disponible en https://introcs.cs.princeton.edu/python

Ejemplo: cuadrados de Albert

```
1  import stddraw
2  from color import Color
3
4  def readint(): return int(input())
5
6  r1 = readint()
7  g1 = readint()
8  b1 = readint()
9  c1 = Color(r1, g1, b1)
10
11 r2 = readint()
12 g2 = readint()
13 b2 = readint()
14 c2 = Color(r2, g2, b2)
15
16 stddraw.setCanvasSize(512, 256)
17 stddraw.setYscale(.25, .75)
18
19 stddraw.setPenColor(c1)
20 stddraw.filledSquare(.25, .5, .2)
21
22 stddraw.setPenColor(c2)
23 stddraw.filledSquare(.25, .5, .1)
24
25 stddraw.setPenColor(c2)
26 stddraw.filledSquare(.75, .5, .2)
27
28 stddraw.setPenColor(c1)
29 stddraw.filledSquare(.75, .5, .1)
30
31 stddraw.show()
```

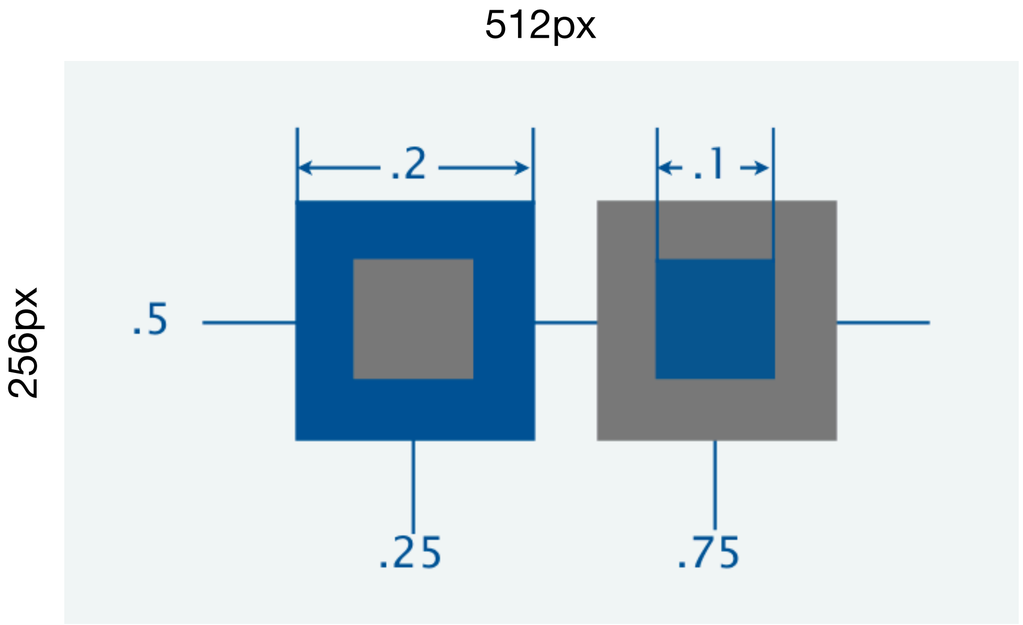
¡Módulo para dibujar!

Crea primer color

Crea segundo color

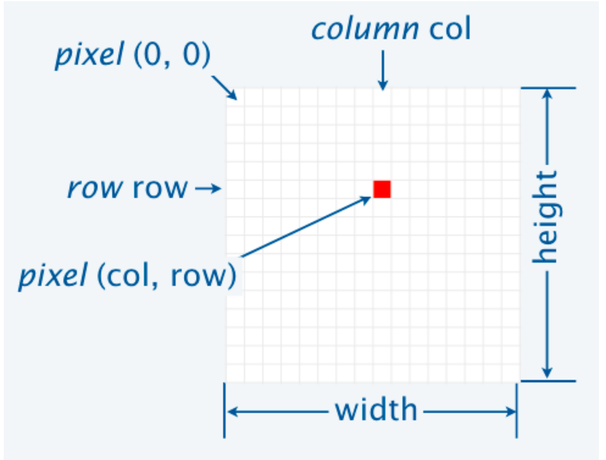
Primer cuadrado

Segundo cuadrado

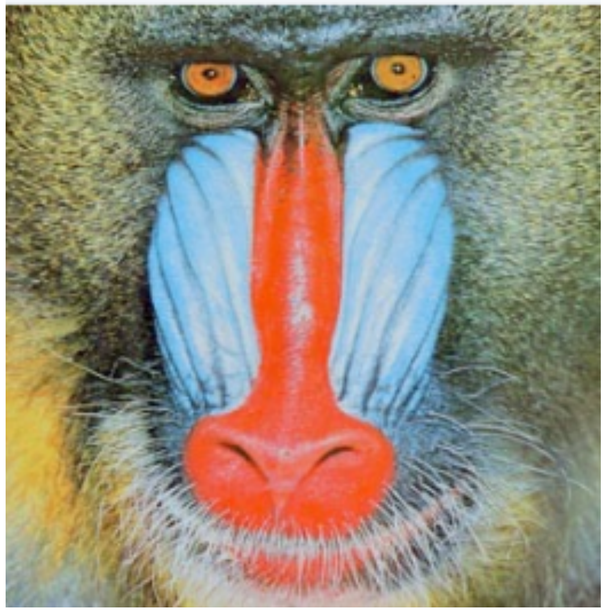



```
$ python3 alberssquares.py
0
64
128
105
105
105
```

API para imágenes



- Una imagen es un arreglo 2D de pixeles.
  - **Valores**: color de cada pixel.



class Picture

	Picture(filename)	Crea una imagen desde archivo
	Picture(w, h)	Crea una imagen vacía de w por h
int	width()	Ancho de la imagen
int	height()	Altura de imagen
Color	get(col, row)	Color del pixel (col, row)
	set(col, row, c)	Asigna color c a pixel en (col, row)
	show()	Muestra la imagen en una ventana
	save(filename)	Guarda imagen en archivo

http://www.cs.princeton.edu/courses/archive/fall18/cos126/lectures/CS.8.ADTs.pdf

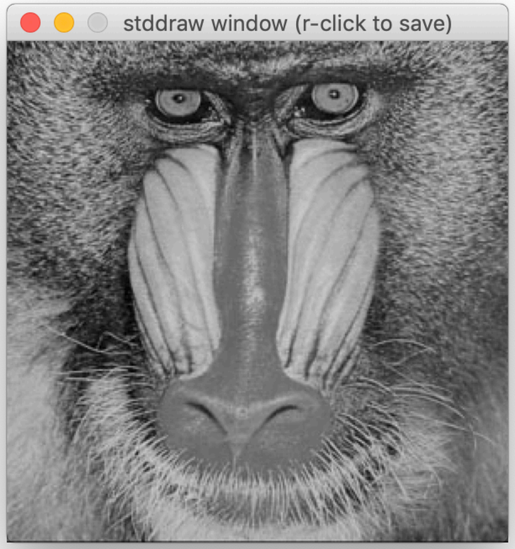
Aplicación: convertir a blanco y negro

```
1  import sys
2  import stddraw
3  import luminance
4  from picture import Picture
5
6  pic = Picture(sys.argv[1])
7
8  for col in range(pic.width()):
9      for row in range(pic.height()):
10         pixel = pic.get(col, row)
11         gray = luminance.toGray(pixel)
12         pic.set(col, row, gray)
13
14 stddraw.setCanvasSize(pic.width(), pic.height())
15 stddraw.picture(pic)
16 stddraw.show()
```

Crea imagen desde archivo

Transforma color en blanco y negro

Crea ventana y muestra imagen



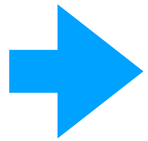

```
$ python3 grayscale.py mandrill.jpg
```

Human-based python interpreter™

- ¿Qué hace el siguiente código?

```
for col in range(pic.width()):
    for row in range(pic.height()):
        pic.set(col, row, pic.get(col, row))

stddraw.show()
```



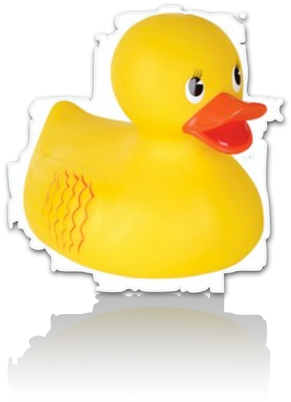
Human-based python interpreter™

- ¿Qué hace el siguiente código? (no tan fácil como el anterior)

```
for col in range(pic.width()):
    for row in range(pic.height()):
        pic.set(col, pic.height()-row-1, pic.get(col, row))

stddraw.show()
```

Intenta reflejar verticalmente la imagen...
Pero falla.



Human-based python interpreter™

- **Solución**: completar una imagen vacía.

```
source = Picture(sys.argv[1])
target = Picture(source.width(), source.height())
for col in range(source.width()):
    for row in range(source.height()):
        target.set(col, source.height()-row-1, source.get(col, row))

stddraw.setCanvasSize(target.width(), target.height())
stddraw.picture(target)
stddraw.show()
```

Ejemplo: escalar imagen

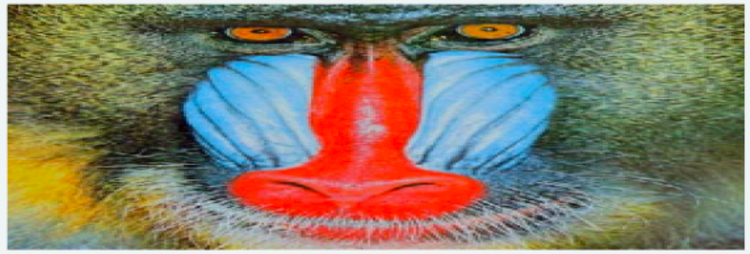
- Objetivo: escalar la imagen independiente del ancho y alto destino.

300x300

Source: mandrill.jpg

500x500

600x200

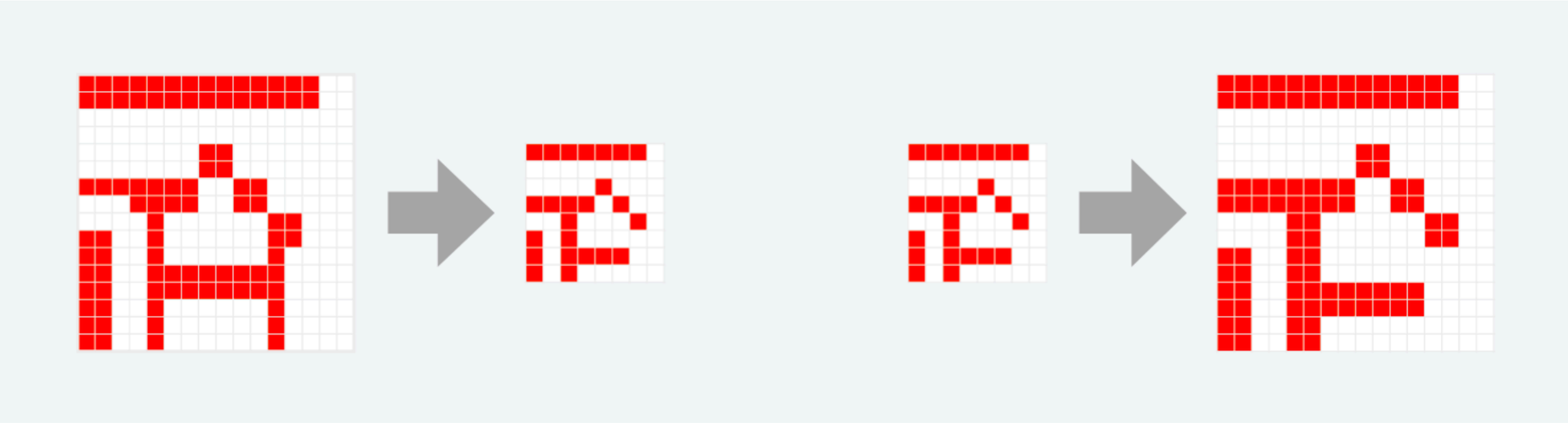


99x99

200x400

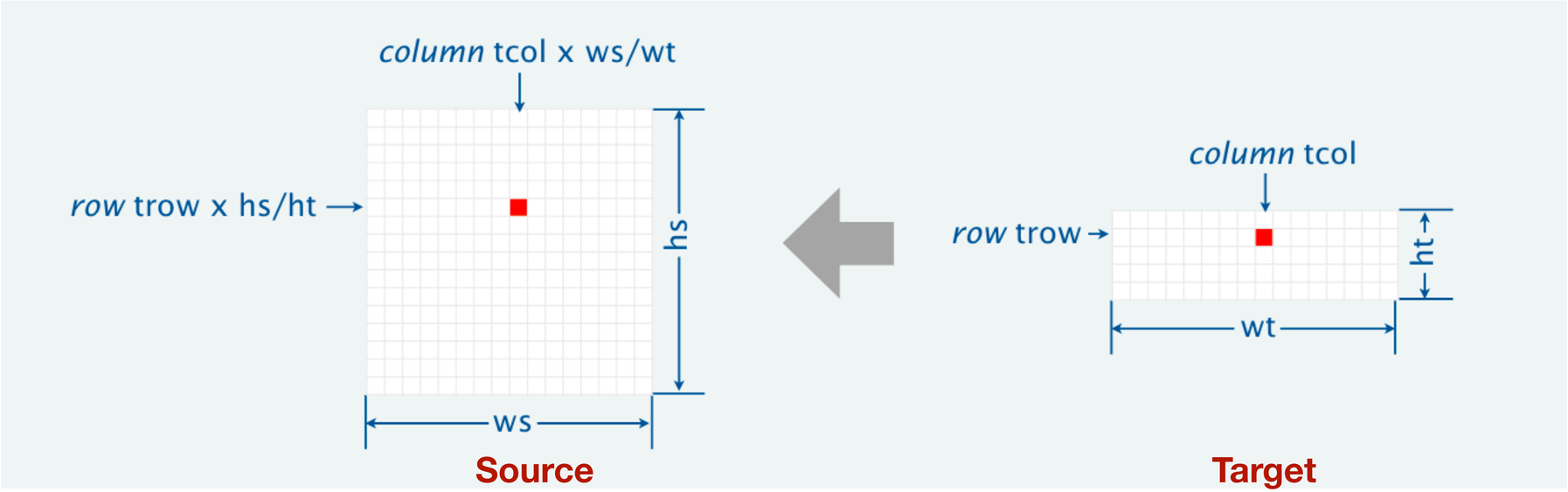
Ejemplo: escalar imagen

- Objetivo: escalar la imagen independiente del ancho y alto destino.
- Ejemplo de estrategia:
  - Achicar imagen eliminando filas y columnas alternadamente.
  - Agrandar imagen cuadruplicando cada pixel.
  - ¿Problema?



Ejemplo: escalar imagen

- Objetivo: escalar la imagen independiente del ancho y alto destino.
- **Estrategia** uniforme:
  - escalar columnas ubicando el índice en posición ws//wt
  - escalar filas ubicando el índice en posición hs//ht
- **Enfoque**: computar cada pixel en la imagen destino (del tamaño a escalar).

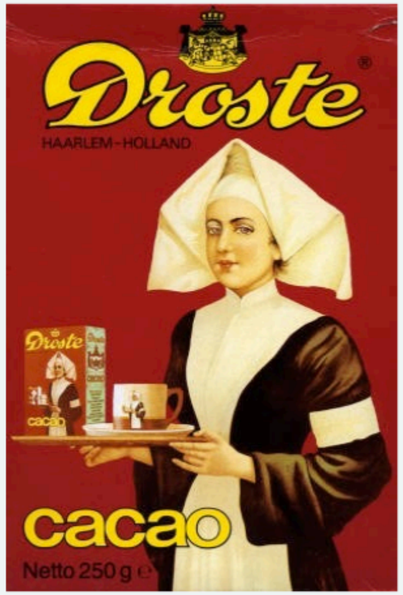


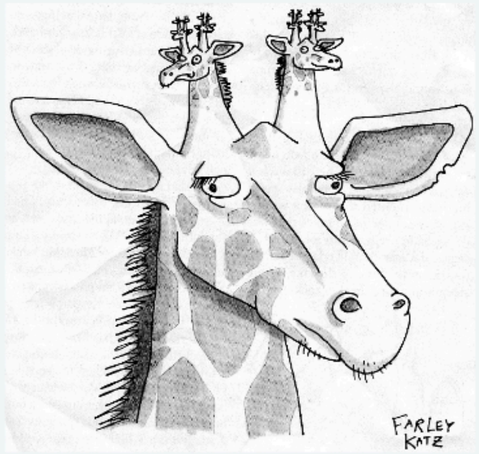
Ejemplo: escalar imagen

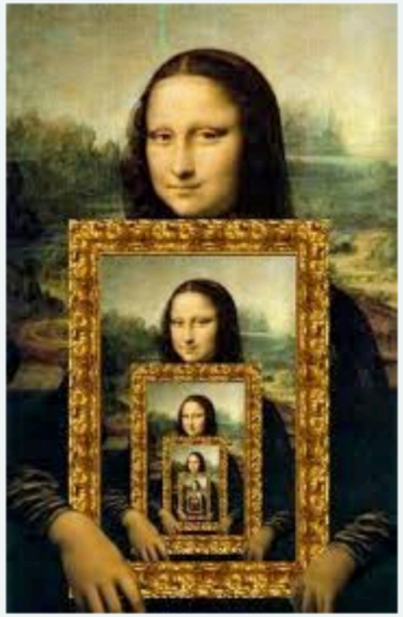
```
1  import stddraw
2  import sys
3  from picture import Picture
4
5  fileName = sys.argv[1]
6  w = int(sys.argv[2])
7  h = int(sys.argv[3])
8
9  source = Picture(fileName)
10 target = Picture(w, h)
11
12 for tCol in range(w):
13     for tRow in range(h):
14         sCol = tCol * source.width() // w
15         sRow = tRow * source.height() // h
16         target.set(tCol, tRow, source.get(sCol, sRow))
17
18 stddraw.setCanvasSize(w, h)
19 stddraw.picture(target)
20 stddraw.show()
```

Posición en imagen origen

Recursividad, imágenes y arte

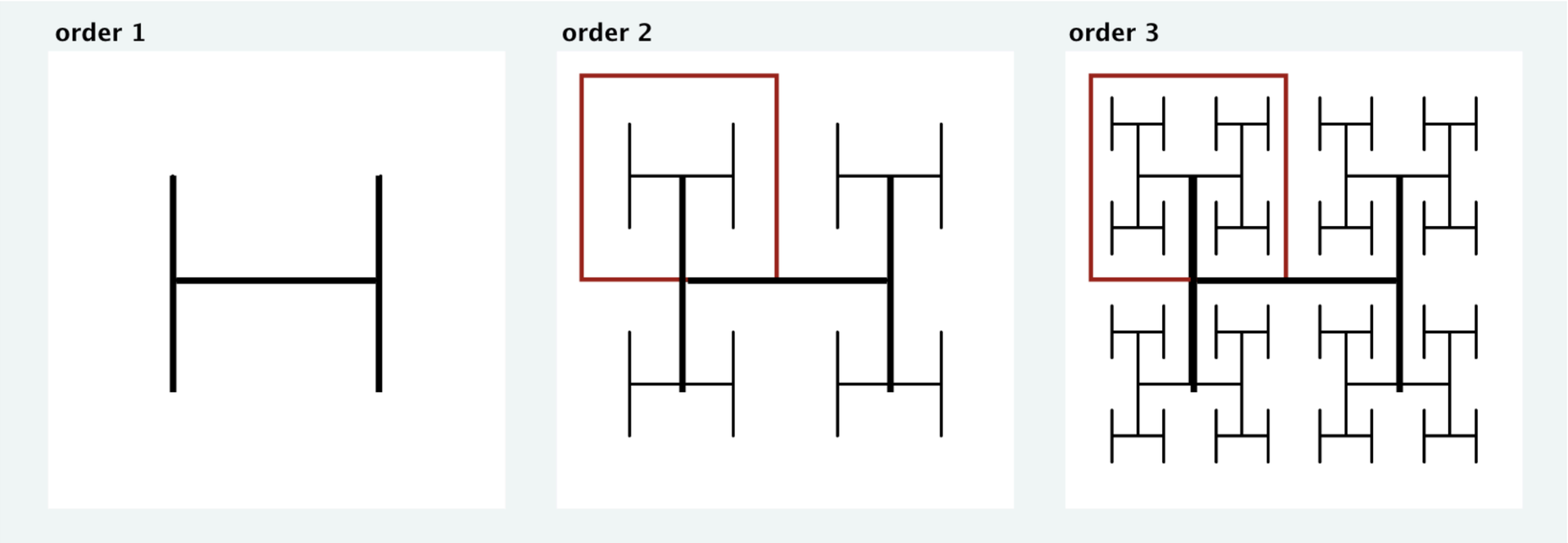






H-Trees

- Un H-Tree de orden n:
  - Si n es 0, hacer nada.
  - Dibujar una H en el centro
  - Dibujar cuatro H-trees de orden n-1, de la mitad del tamaño y centradas en los bordes de la H



Implementación H-Tree

```
1  import stddraw
2  import sys
3
4  def draw(n, lineLength, x, y):
5      if n == 0:
6          return
7      x0 = x - lineLength/2
8      x1 = x + lineLength/2
9      y0 = y - lineLength/2
10     y1 = y + lineLength/2
11
12     stddraw.line(x0, y, x1, y)
13     stddraw.line(x0, y0, x0, y1)
14     stddraw.line(x1, y0, x1, y1)
15
16     draw(n-1, lineLength/2, x0, y0)
17     draw(n-1, lineLength/2, x0, y1)
18     draw(n-1, lineLength/2, x1, y0)
19     draw(n-1, lineLength/2, x1, y1)
20
21 def main():
22     n = int(sys.argv[1])
23     stddraw.setPenRadius(0.0)
24     draw(n, .5, .5, .5)
25     stddraw.show()
26
27 if __name__ == '__main__':
28     main()
```