Week 10

CSC111 — Fall 2018

Dominique Thiébaut
dthiebaut@smith.edu

Q8: Nested For Loop

```
for i in range( 1000 ):
   for j in range( 1000 ):
      print( i+j )
```

This Week…

- **Image Processing** with **Nested For-Loops**

References

- Zelle's documentation on graphics.py http://mcsp.wartburg.edu/zelle/python/graphics/graphics.pdf

- Chapter 8 in Zelle: **nested for-loops**

- **Image Geometry & Coordinate System**
- **Scanning Images using Nested For-Loops**
- **Sweep**
- **How RGB Works**
- **Python Code for Image Processing**
- **Demos**

Image Processing

- Different image *types*: jpg, png, **gif**, eps, svg, tiff, etc.

- Zelle **graphics.py** library compatible with **gif** images only.

- Jpg and png files can be **converted to gif** using Web services
  (e.g. http://image.online-convert.com/convert-to-gif)

Image Geometry

width

height

309 pixels × 163 pixels

~50K pixels

Image Geometry

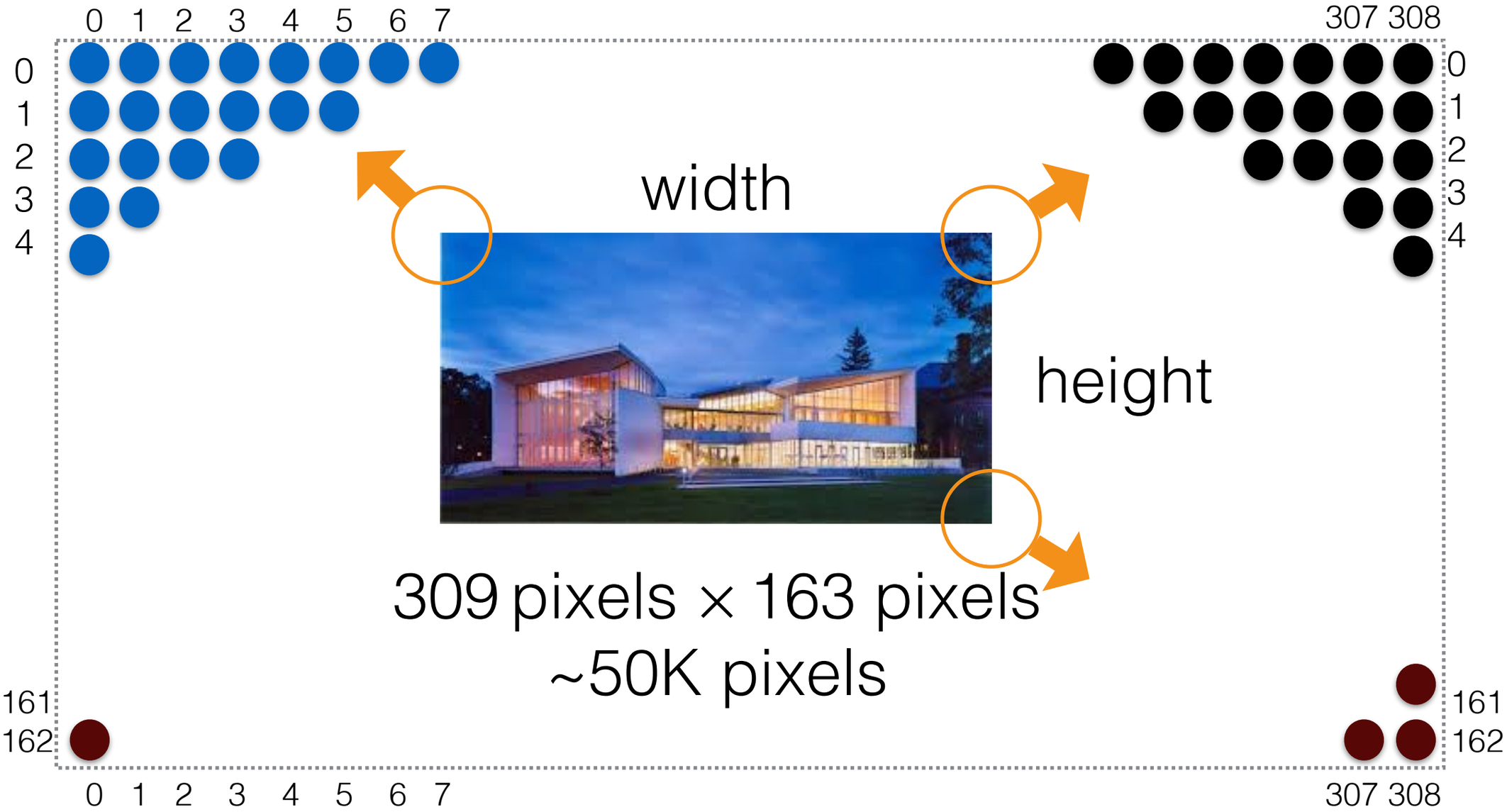
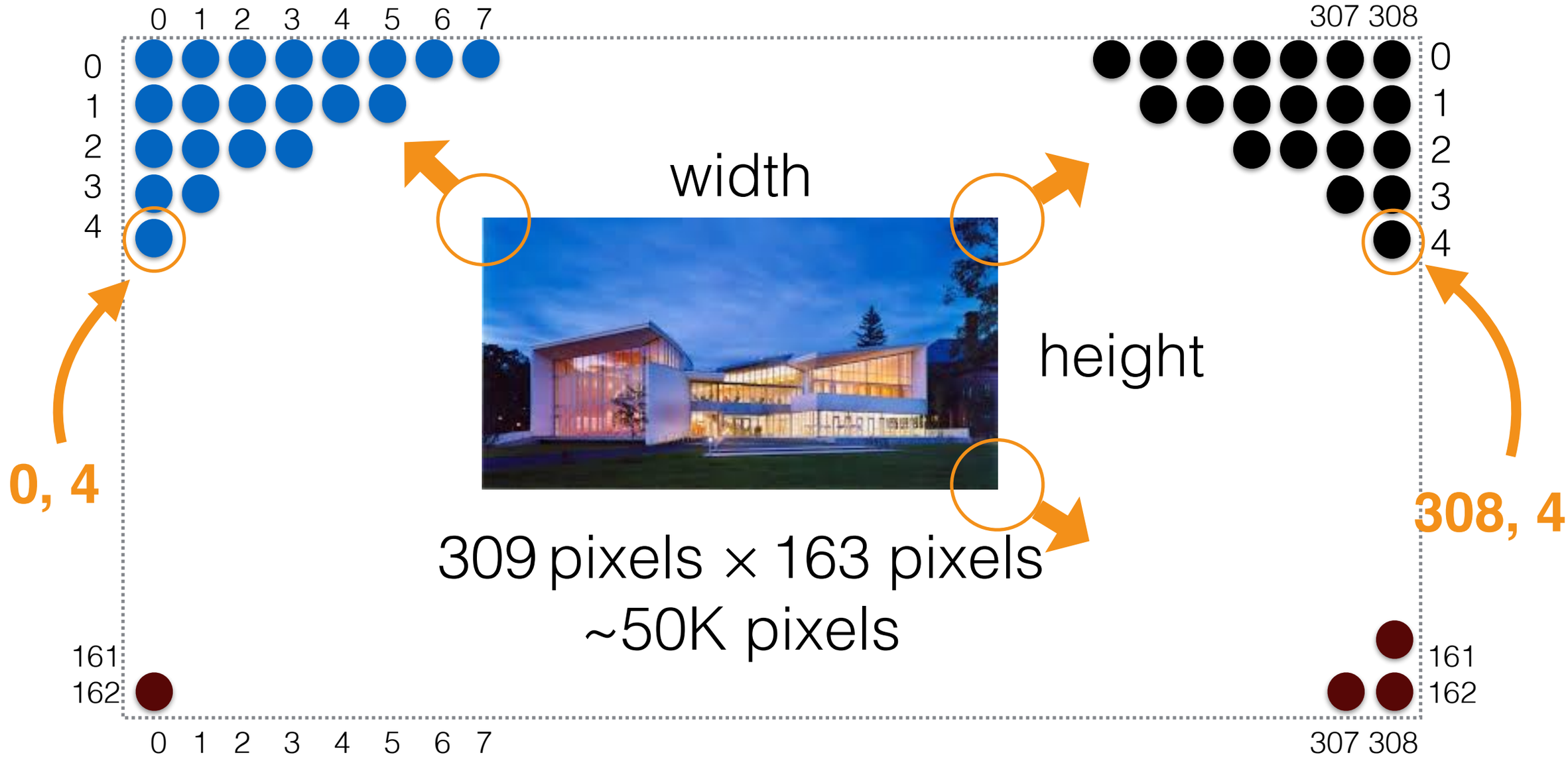
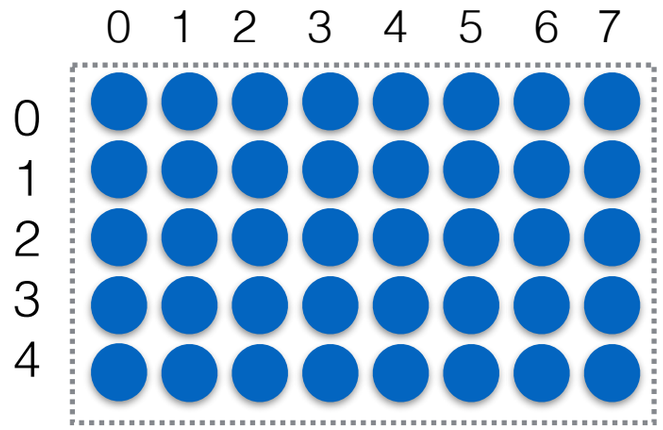


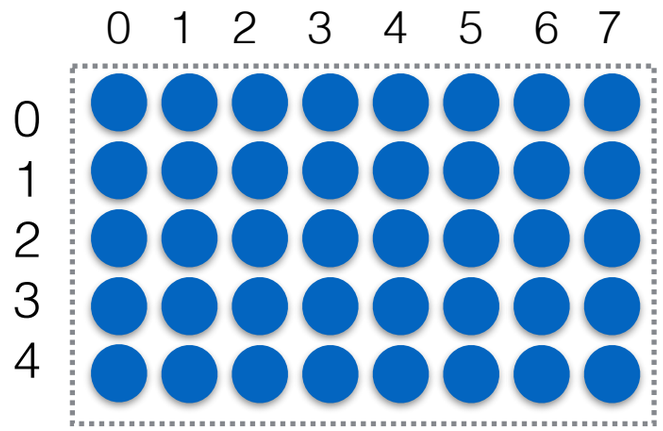
Image Geometry



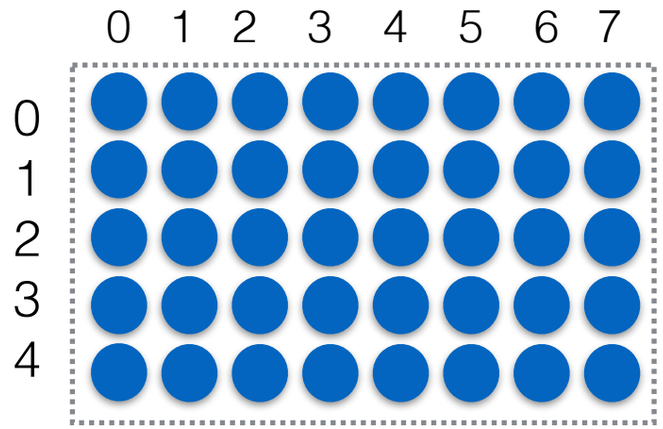
- Image Geometry & Coordinate System
- **Scanning Images using Nested For-Loops**
- Sweep
- How RGB Works
- Python Code for Image Processing
- Demo



Change all the pixels to red

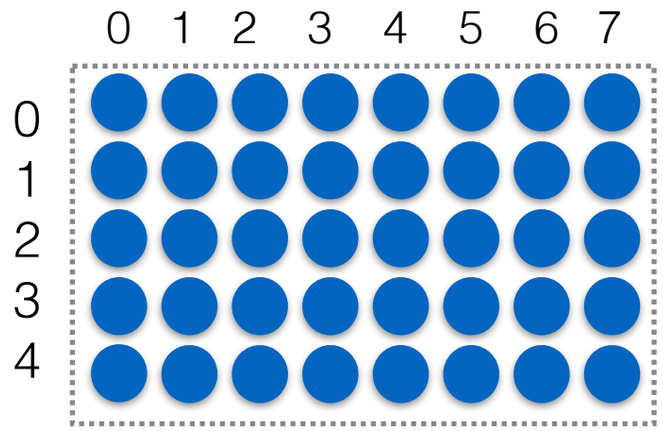



```
WIDTH = 8
HEIGHT = 5

for x in range( WIDTH ):
    for y in range( HEIGHT ):
        makePixelRed( x, y )
```

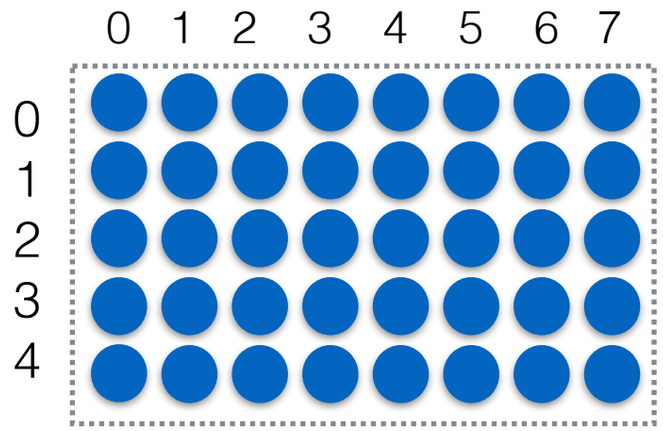


(Switch to numbers: easier to understand)

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```

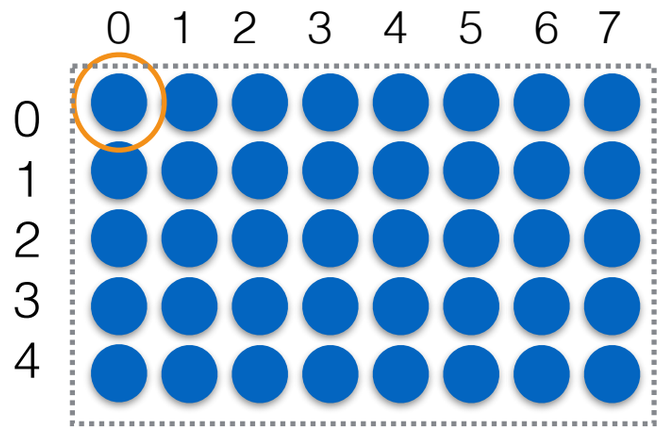



```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



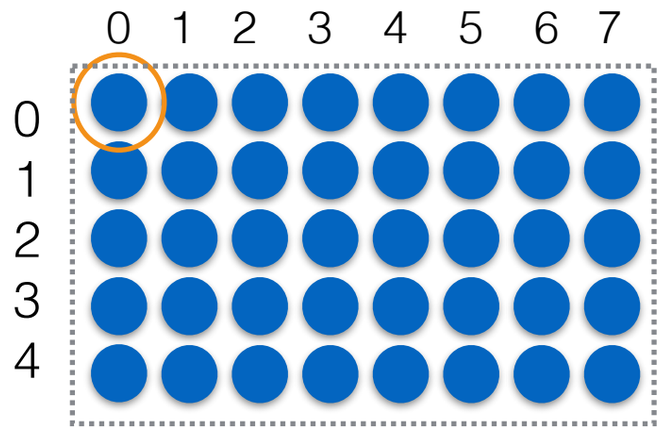
x=0

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



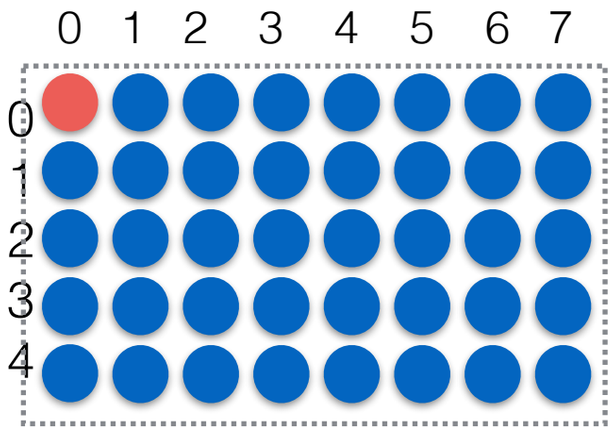
x=0
y=0

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



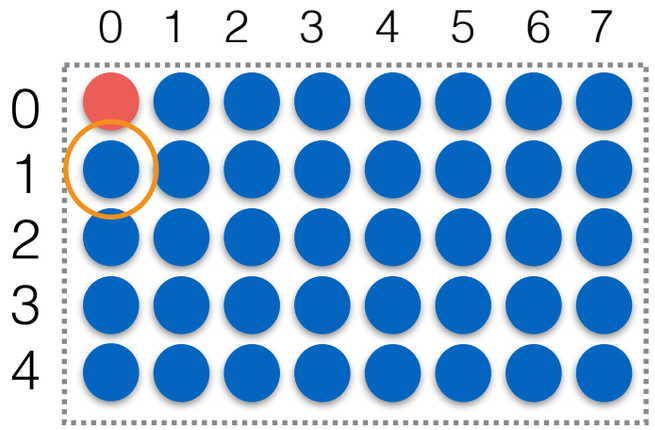
x=0
y=0

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



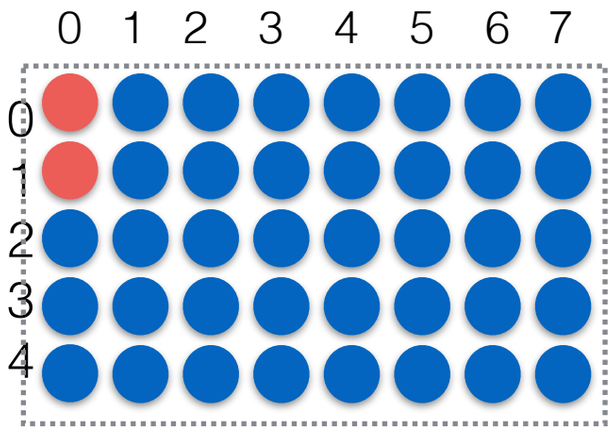
x=0
y=0

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



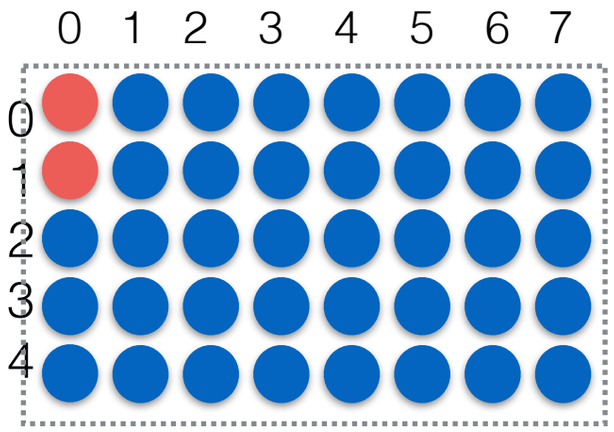
x=0
y=1

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



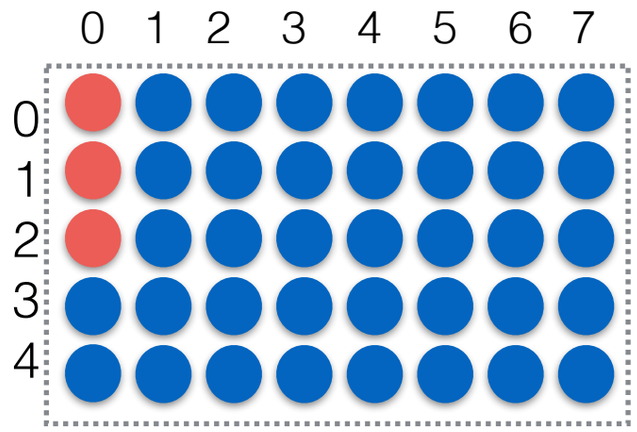
x=0
y=1

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



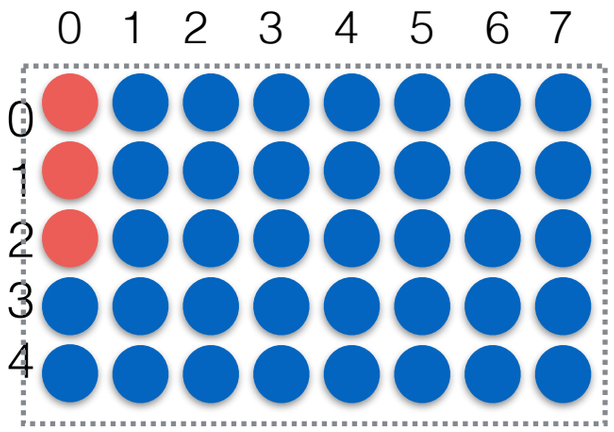
x=0
y=2

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



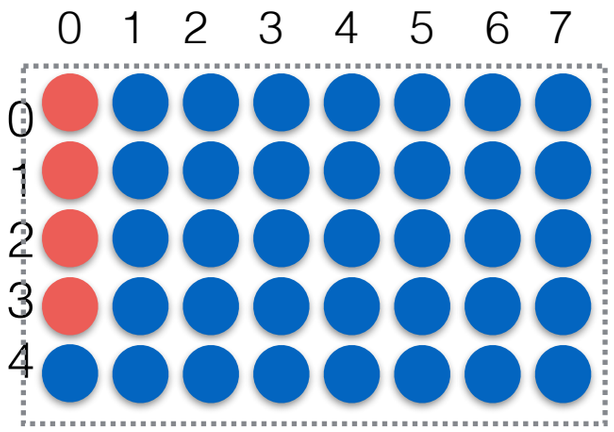
x=0
y=2

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



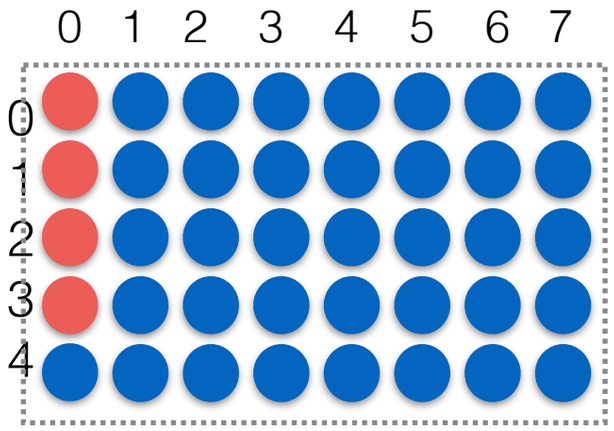
x=0
y=3

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



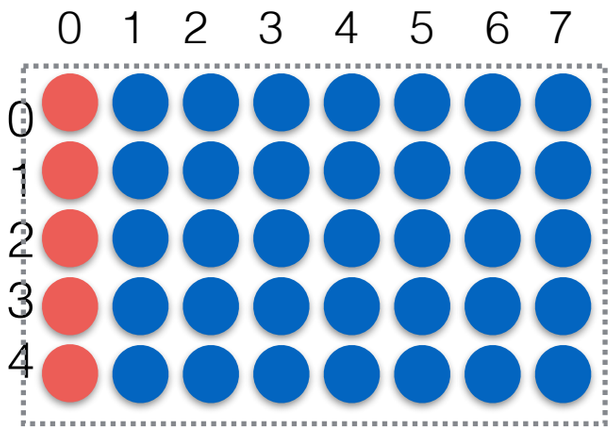
x=0
y=3

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



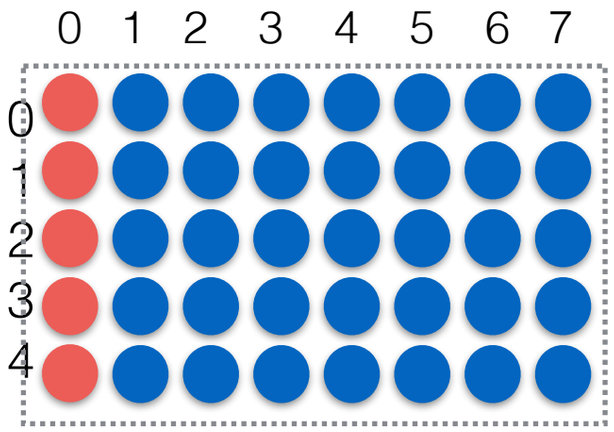
x=0
y=4

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



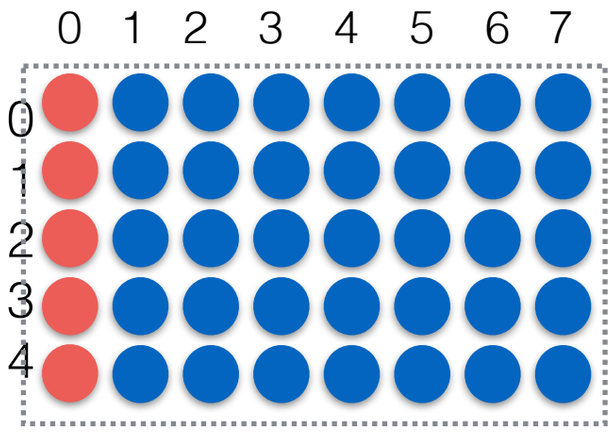
x=0
y=4

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



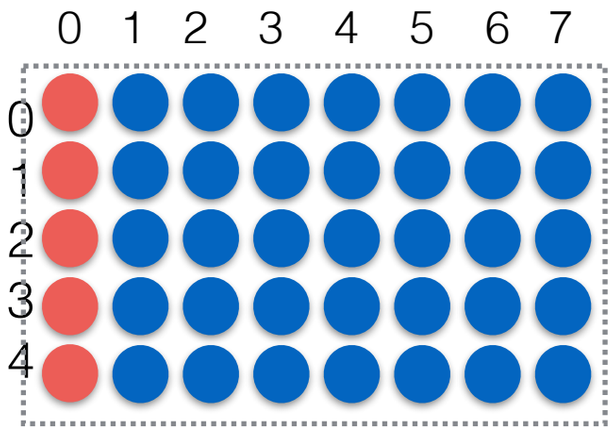
x=0
y=

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



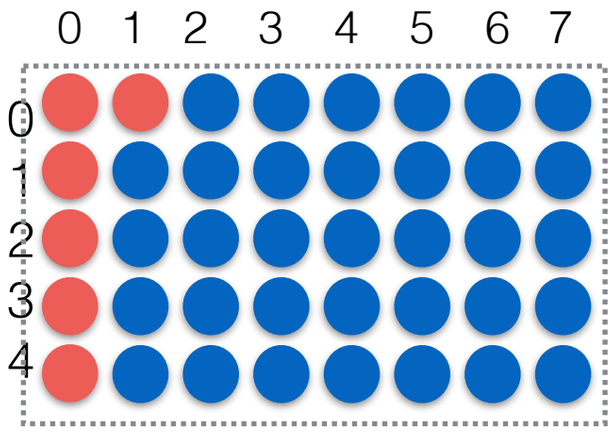
x=1
y=

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



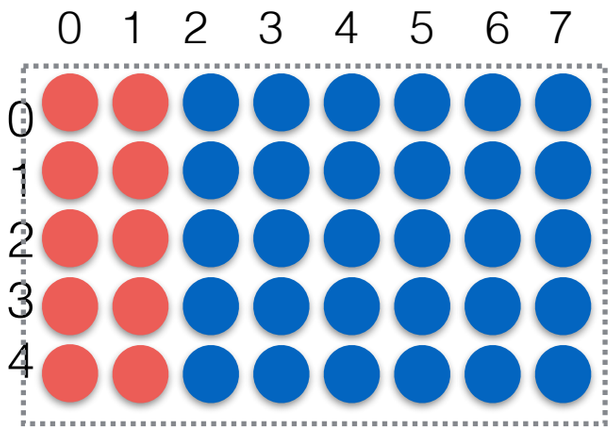
x=1
y=0

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



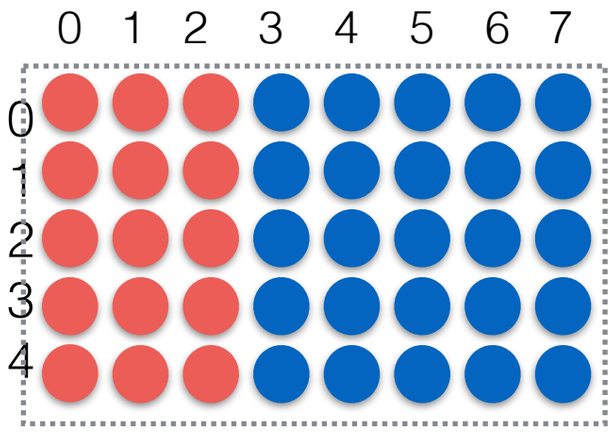
x=1
y=0

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```

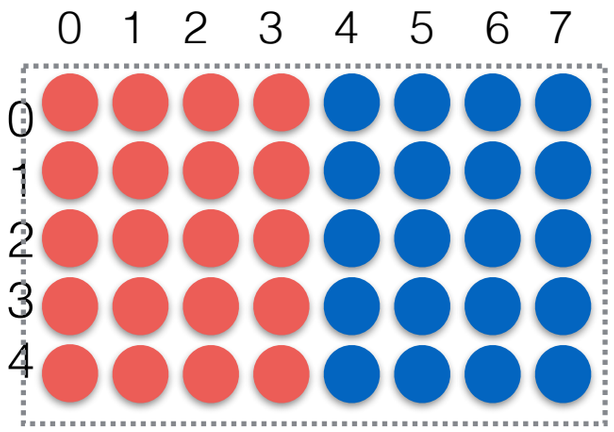


x=1
y=~~0~~1~~2~~3~~4~~

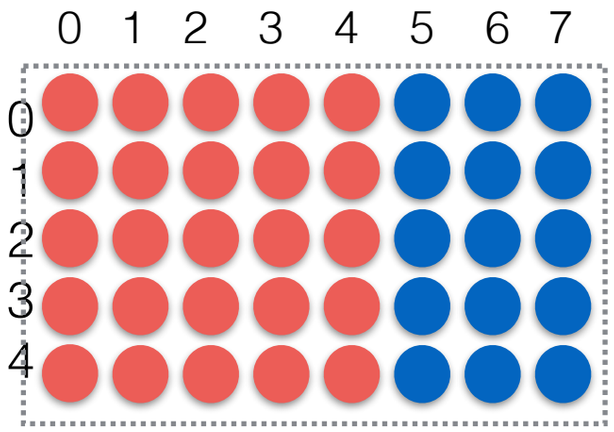
```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```



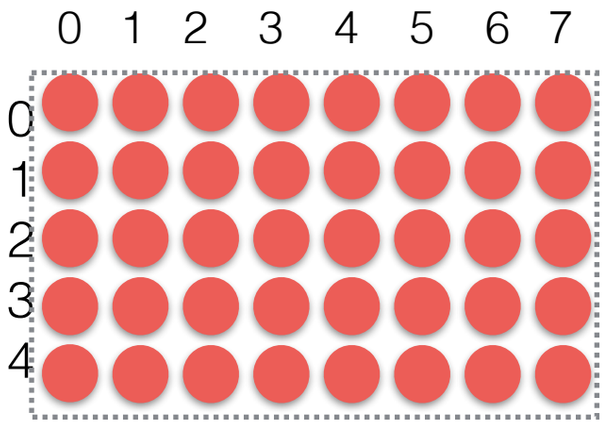

```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```




```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```

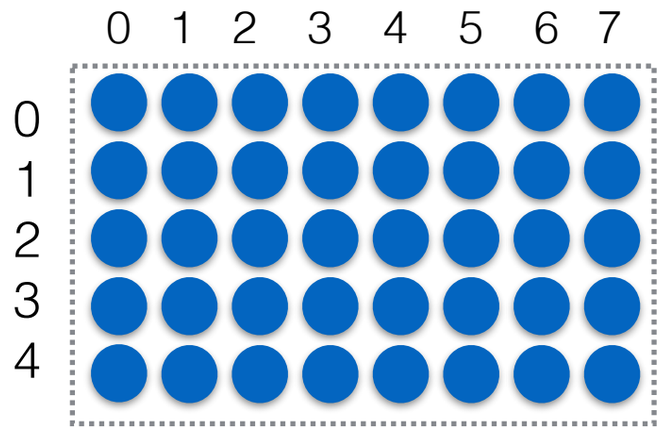



```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```

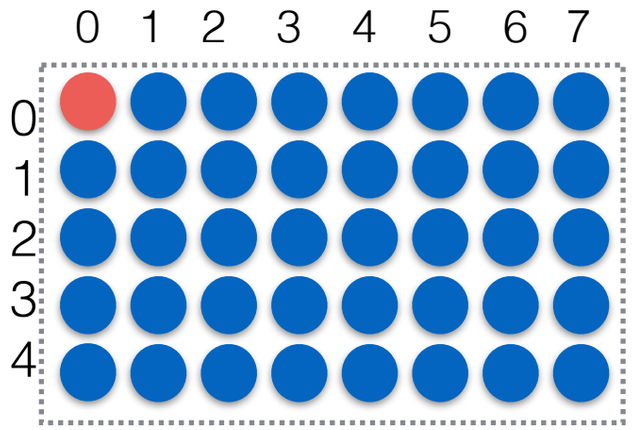



```
WIDTH = 8
HEIGHT = 5

for x in range( 8 ):
    for y in range( 5 ):
        makePixelRed( x, y )
```

Switching the Loops

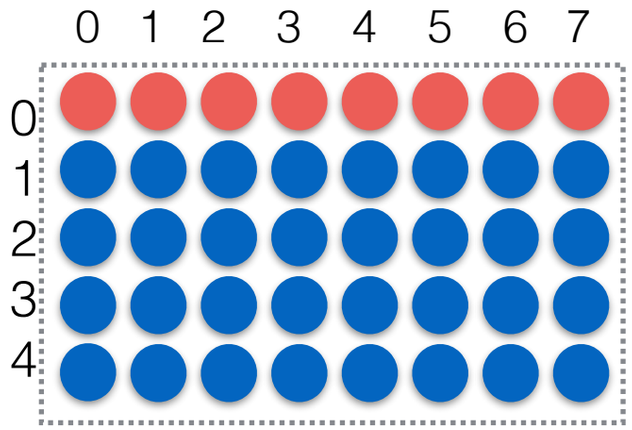



```
WIDTH = 8
HEIGHT = 5

for y in range( 5 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```



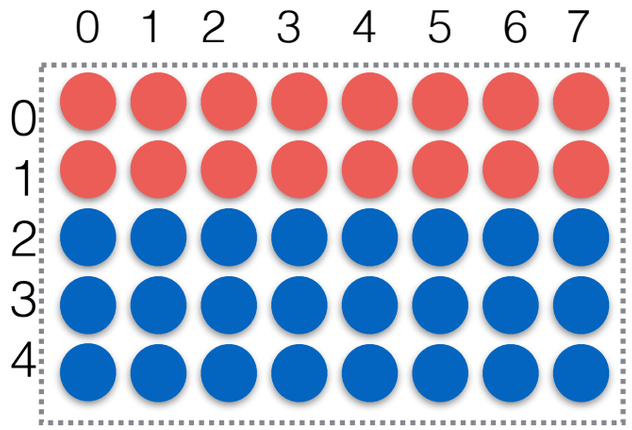
y=0
x=0

```
WIDTH = 8
HEIGHT = 5

for y in range( 5 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```

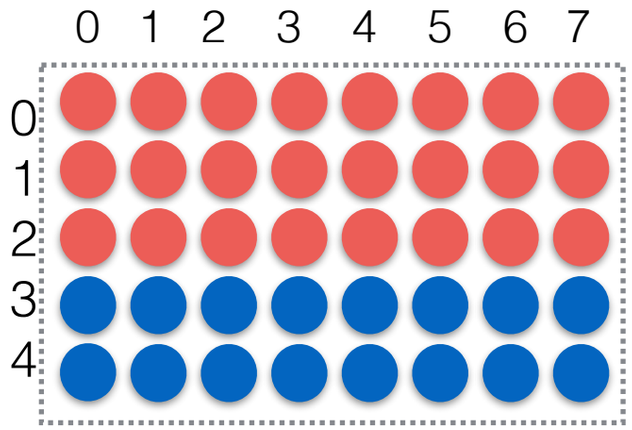


y=0
x=*0 1 2 3 4 5 6 7*

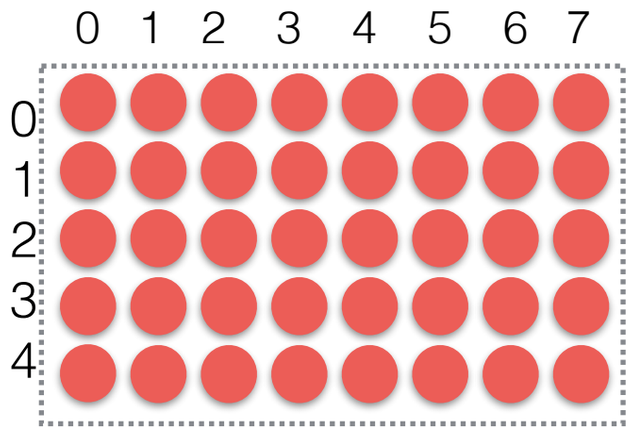
```
WIDTH = 8
HEIGHT = 5

for y in range( 5 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```




```
WIDTH = 8
HEIGHT = 5

for y in range( 5 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```

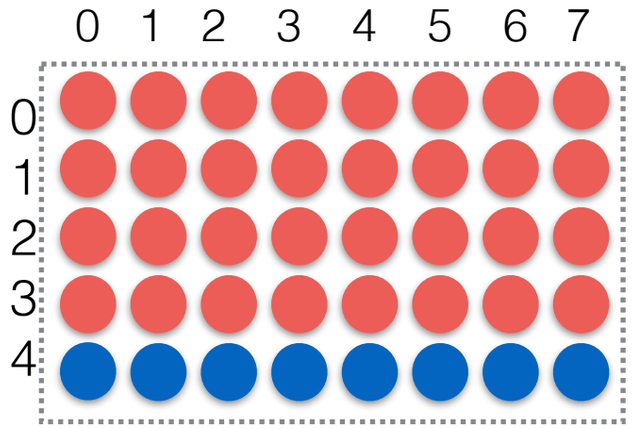



```
WIDTH = 8
HEIGHT = 5

for y in range( 5 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```



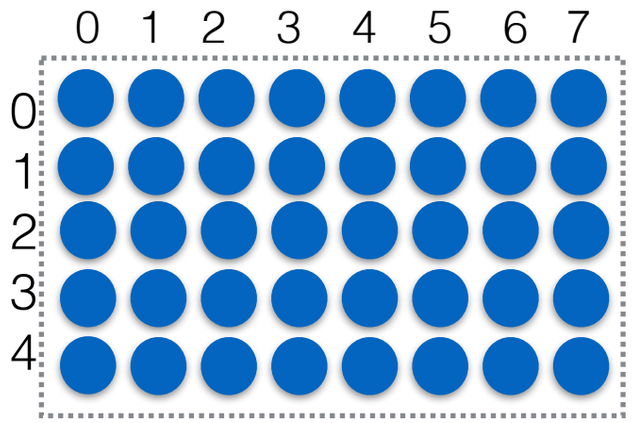

```
WIDTH = 8
HEIGHT = 5

for y in range( 5 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```

- Image Geometry & Coordinate System
- Scanning Images using Nested For-Loops
- **Sweep**
- How RGB Works
- Python Code for Image Processing
- Demo



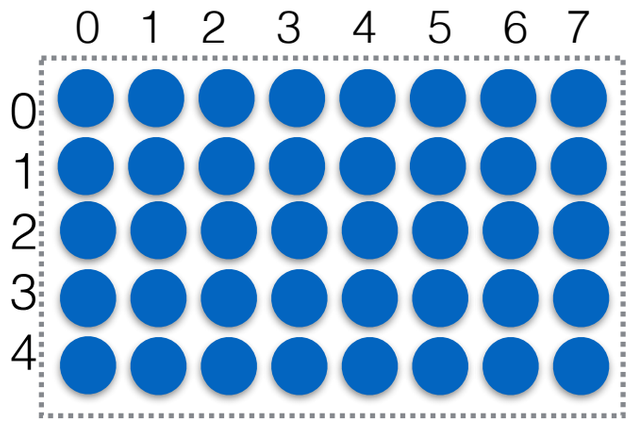
sweep

```
WIDTH = 8
HEIGHT = 5

for y in range( 5 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```



sweep ?

```
WIDTH = 8
HEIGHT = 5

for y in range( ? ):
    for x in range( ? ):
        makePixelRed( x, y )
```



sweep!

```
WIDTH = 8
HEIGHT = 5

for y in range( 4, -1, -1 ):
    for x in range( 8 ):
        makePixelRed( x, y )
```

- Image Geometry & Coordinate System
- Scanning Images using Nested For-Loops
- Sweep
- **How RGB Works**
- Python Code for Image Processing
- Demo

How RGB Works

pixel

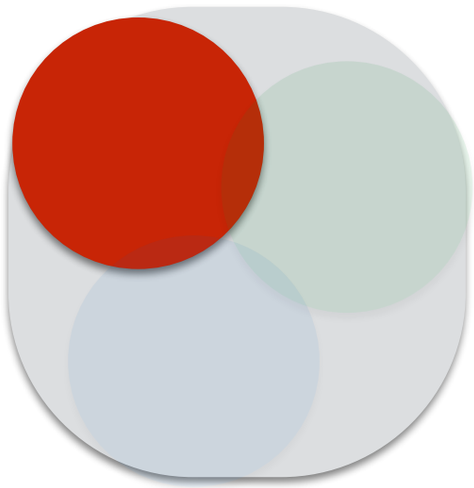
How RGB Works

pixel

Red	**G**reen	**B**lue
0-255	0-255	0-255

RGB System

How RGB Works

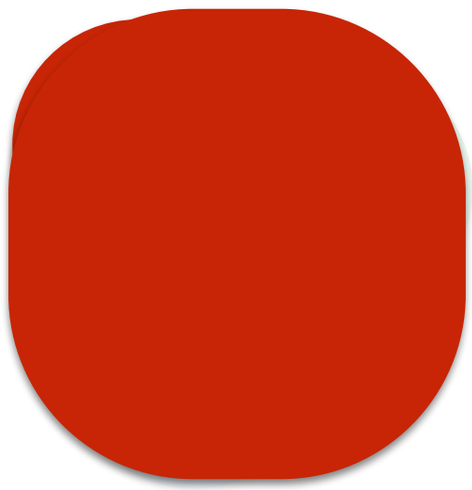


pixel

Red	Green	Blue
255	0	0

RGB System

How RGB Works



pixel

Red	Green	Blue
255	0	0

RGB System

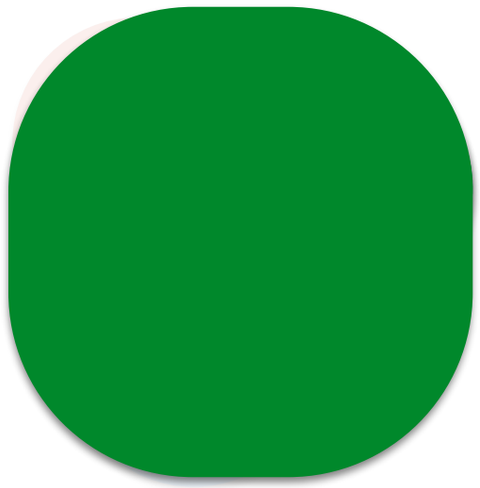
How RGB Works

pixel

Red	Green	Blue
0	255	0

RGB System

How RGB Works

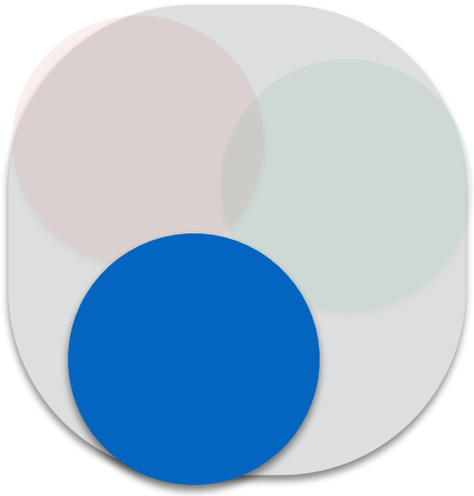


pixel

Red	Green	Blue
0	255	0

RGB System

How RGB Works

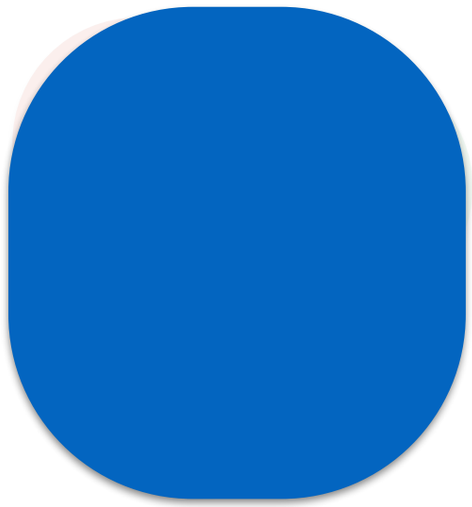


pixel

Red	Green	Blue
0	0	255

RGB System

How RGB Works



pixel

Red	Green	Blue
0	0	255

RGB System

How RGB Works

pixel

Red	Green	Blue
51	255	255

RGB System

How RGB Works

pixel

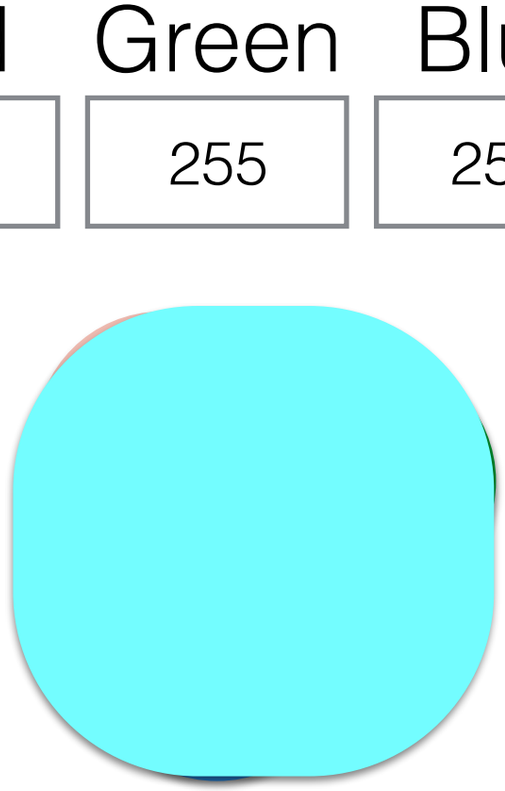
Red	Green	Blue
51	255	255

RGB System

http://www.rapidtables.com/web/color/RGB_Color.htm

Getting a Pixel from an Image

Red	Green	Blue
51	255	255



pixel
at x, y

(we assume we have an ***image*** object called **cat**)

```
red, green, blue = cat.getPixel( x, y )
```

Setting the Color of a Pixel

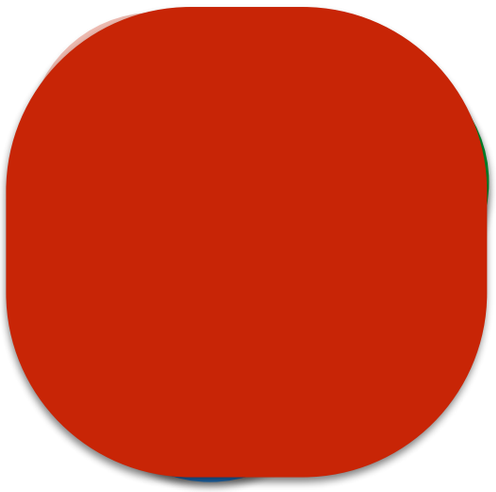
Red	Green	Blue
51	255	255

pixel
at x, y

```
# create color red
color = color_rgb( 255, 0, 0 )

# set pixel at x, y to red
cat.setPixel( x, y, color )
```

Setting the Color of a Pixel

Red	Green	Blue
255	0	0



pixel
at x, y

```
# create color red
color = color_rgb( 255, 0, 0 )

# set pixel at x, y to red
cat.setPixel( x, y, color )
```

binary		**decimal**
1 bit		
0 1		0 - 1

binary		decimal
2 bits		
00 01 10 11		0 - 3

$$2^2 = 4$$

binary		decimal
3 bits		
000 001 010 011 100 101 110 111		0 - 7

$2^3 = 8$

binary

4 bits

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

decimal

0 - 15

$2^4 = 16$

binary

8 bits = *byte*

00000000
00000001
00000010
00000011
00000100
00000101
00000110
. . .
11111000
11111001
11111010
11111011
11111100
11111101
11111110
11111111

decimal

0 - 255

$2^8 = 256$

binary

8 bits = *byte*

00000000
00000001
00000010
00000011
00000100
00000101
00000110
. . .
11111000
11111001
11111010
11111011
11111100
11111101
11111110
11111111

decimal

0 - 255

1 pixel = 3 bytes

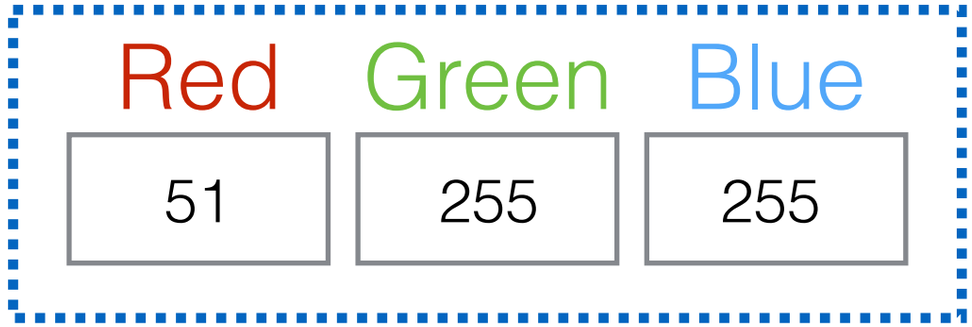
binary

8 bits = *byte*

00000000
00000001
00000010
00000011
00000100
00000101
00000110
. . .
11111000
11111001
11111010
11111011
11111100
11111101
11111110
11111111

decimal

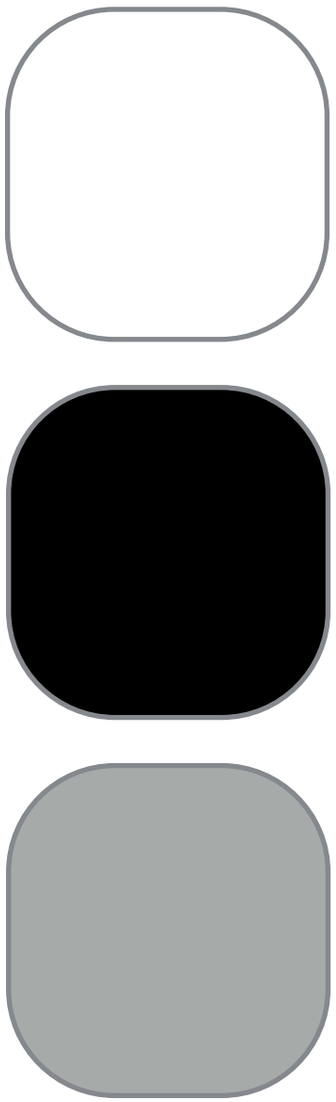
0 - 255



1 pixel = **3 bytes**

256 x **256** x **256** = 16,777,216 colors

Special Colors



Red	Green	Blue
255	255	255

Red	Green	Blue
0	0	0

Red	Green	Blue
101	101	101

- Image Geometry & Coordinate System
- Scanning Images using Nested For-Loops
- Sweep
- How RGB Works
- **Python Code for Image Processing**
- Demo
-

Displaying Images

The graphics module also provides minimal support for displaying and manipulating ima
PPM and GIF images. Display is done with an `Image` object. Images support the generic m
`clone()`. Image-specific methods are given below.

`Image(anchorPoint, filename)`
Constructs an image from contents of the given file, centered at the given anchor po
instead of `filename`. In this case, a blank (transparent) image is created of the give

`getAnchor()`
Returns a clone of the point where the image is centered.

`getWidth()`
Returns the width of the image.

`getHeight()`
Returns the height of the image.

`getPixel(x, y)`
Returns a list `[red, green, blue]` of the RGB values of the pixel at position `(x,y)`
the intensity of the corresponding RGB color. These numbers can be turned into a c
section).

Note that pixel position is relative to the image itself, not the window where the im
always pixel `(0,0)`.

`setPixel(x, y, color)`
Sets the pixel at position `(x,y)` to the given color. Note: this is a slow operation.

`save(filename)`
Saves the image to a file. The type of the resulting file (e.g., GIF or PPM) is determ
`img.save("myPic.ppm")` saves `img` as a PPM file.

http://mcsp.wartburg.edu/zelle/python/graphics/graphics/node13.html

Viewing an Image

imageProcessingSkel0.py - /Users/thiebaut/Desktop/Dropbox/111/imageProcessingSkel0.py (3.5.4)

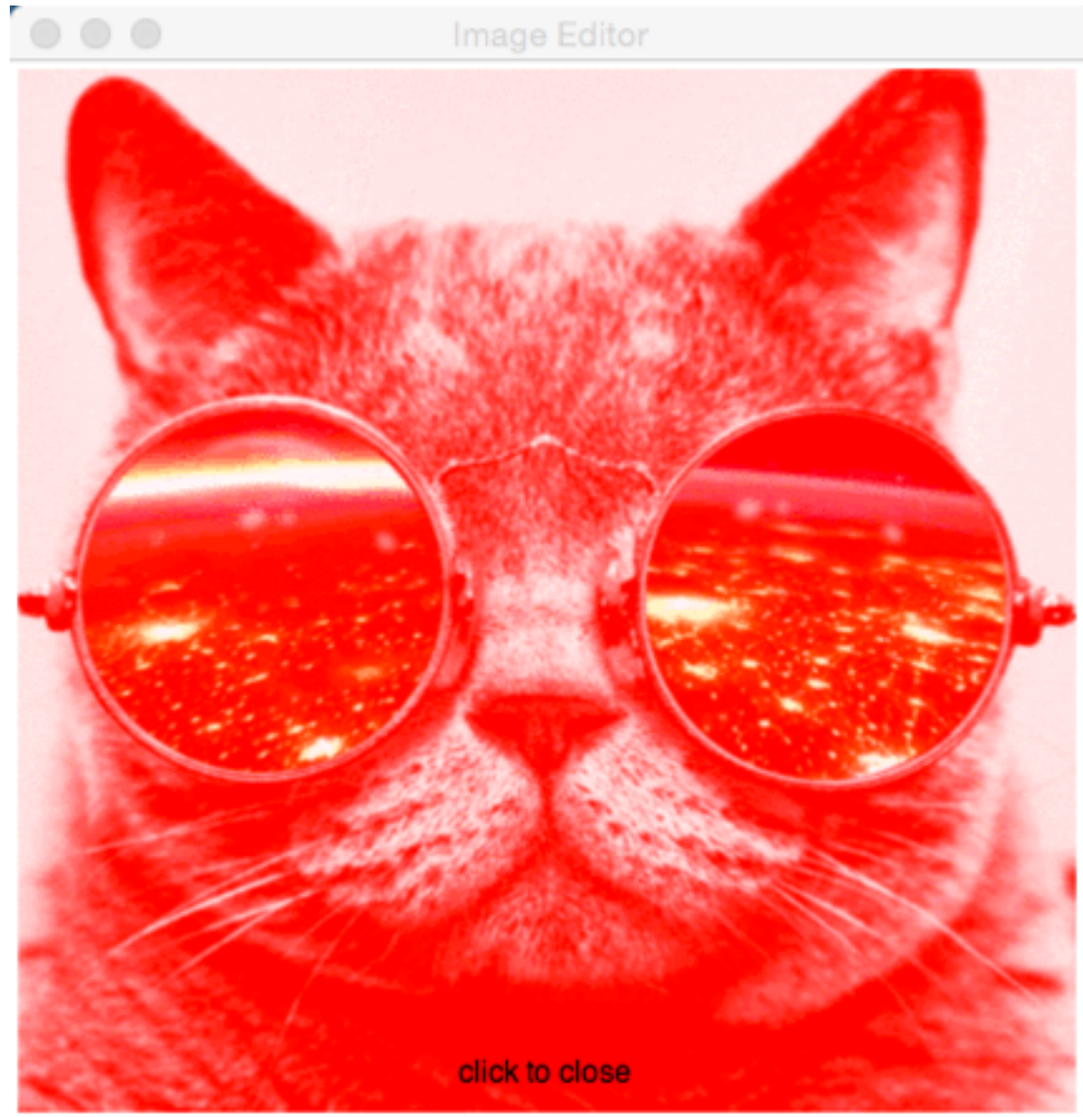
```
# imageProcessingSkel.py
# D. Thiebaut
# A skeleton program to start doing image processing
# in Python
from graphics import *

WIDTH  = 243    # width and height for catHat.gif
HEIGHT = 207

def main():
    # open the window
    win = GraphWin( "CSC111 Image Viewer", WIDTH, HEIGHT )

    # get the file name from the user
    fileName = "catHat.gif"

    # open the cat image
    cat = Image( Point(WIDTH//2, HEIGHT//2), fileName )
    cat.draw( win )

    # wait for user to click the mouse...
    win.getMouse()

    # and close the window
    win.close()

main()
```

Ln: 21 Col: 0

Processing Pixels of an Image

```
def makeRed( img, win ):
    """ set the red component of all the pixels to 255, the
    maximum value.
    """
    global WIDTH, HEIGHT
    for y in range( HEIGHT ):
        win.update()
        for x in range( WIDTH ):
            red, green, blue = img.getPixel( x, y )

            red = 255
            img.setPixel( x, y, color_rgb(red, green, blue ) )


def main():

    win = GraphWin( "CSC111 Image Viewer", WIDTH, HEIGHT )

    img = Image( Point(WIDTH//2, HEIGHT//2), IMAGEFILENAME )
    img.draw( win )

    makeRed( img, win )

    win.getMouse()
    win.close()

main()
```

Demo Time!



Transformations to Try

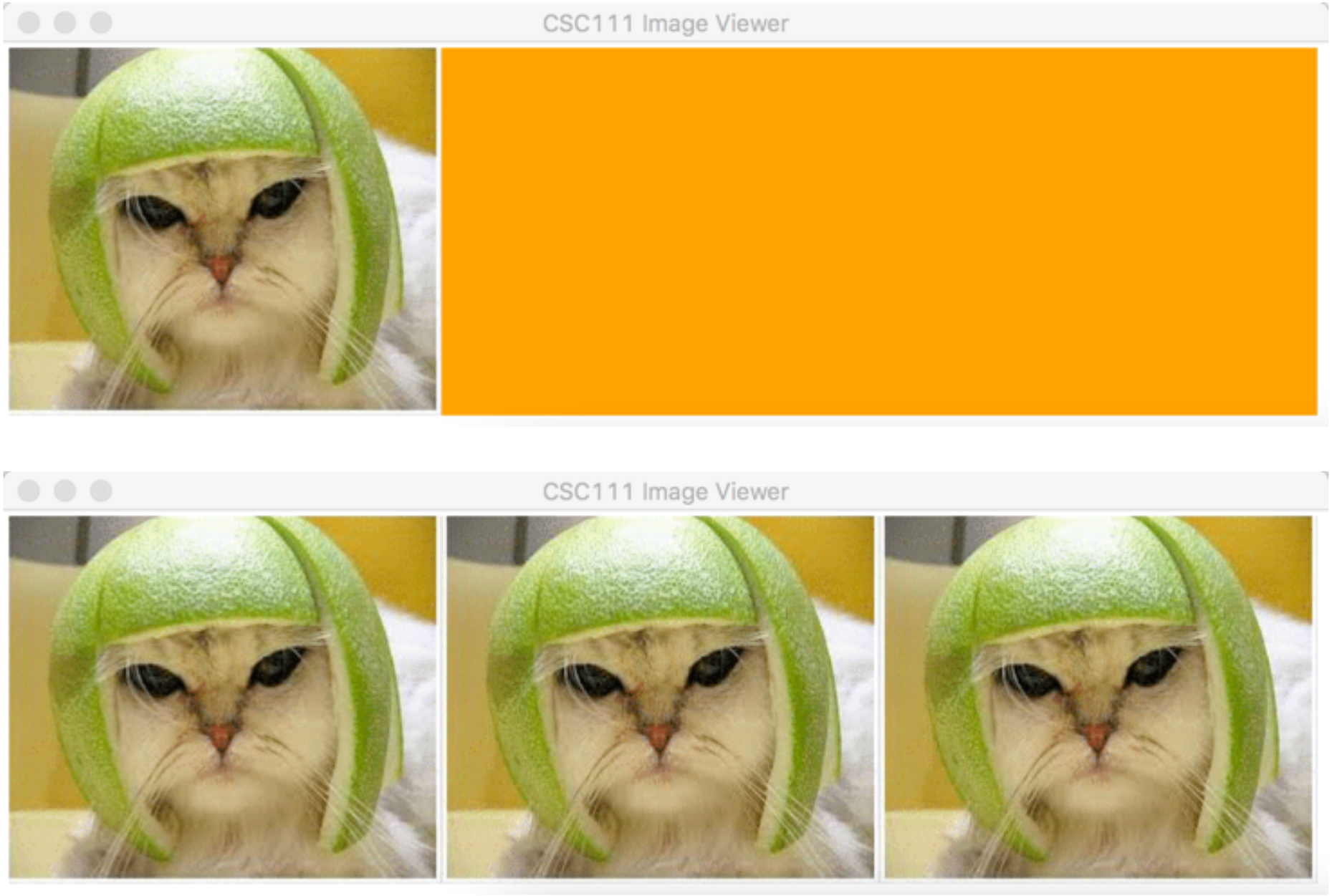
- Modify RED component (red=(blue+green)/2, red=0, red=red/2)
- swap colors (r, g, b = b, r, g)
- Try to "saturate" a color (color = 0 if less than 128, 255 otherwise)
- Draw a horizontal line (beginning of a border)

We stopped here last time…

- **Mirroring an Image**
- **Displaying a Checker Board**
    - **8x8 Grid**
    - **Alternating Colors**
- **Creating a Class for a Checkers Piece**
- **Using a Gif Image for a Piece**

Transformations to Try

- Copy Part of Photo

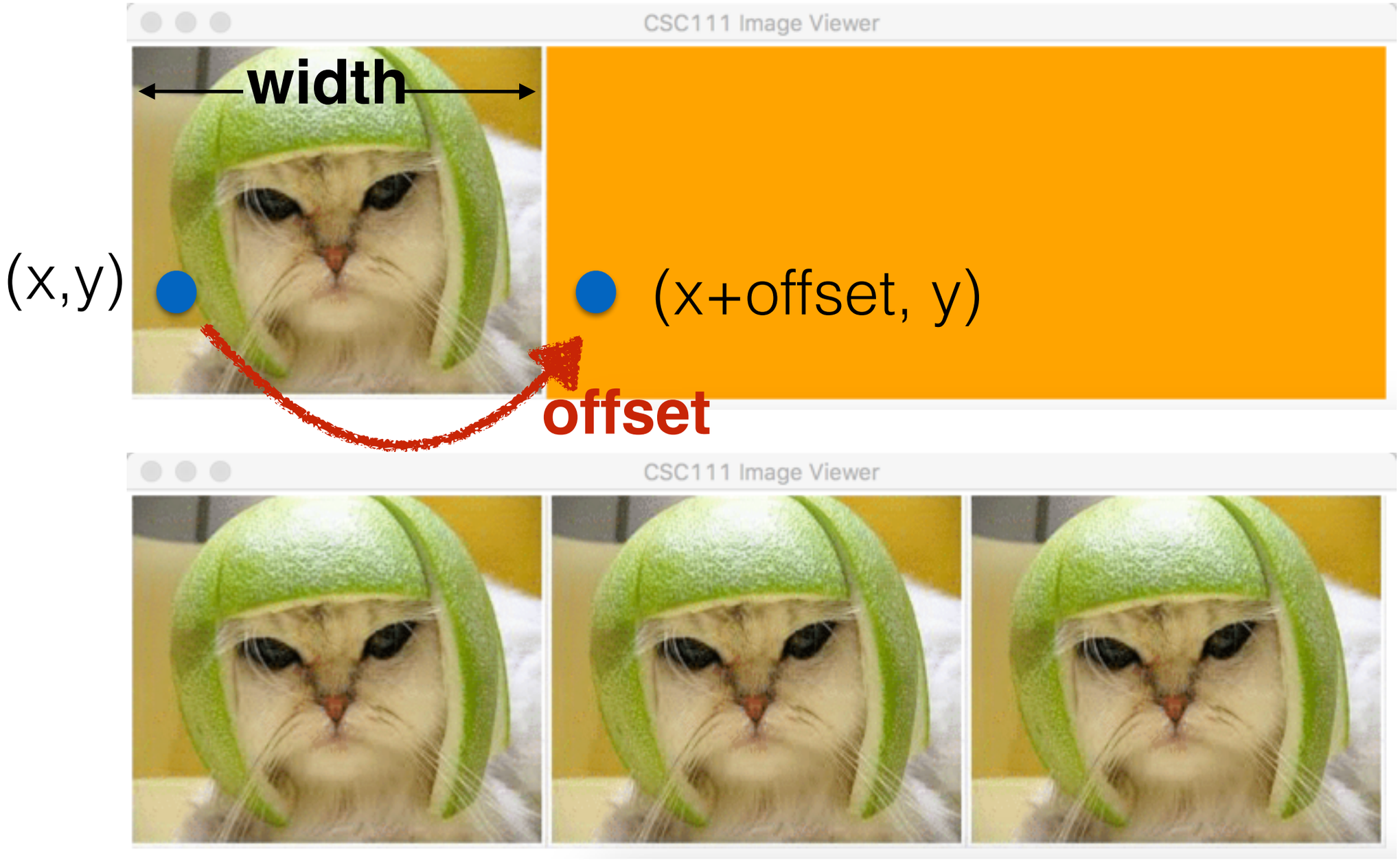


Transformations to Try

- Copy Part of Photo

Transformations to Try

- Copy Part of Photo



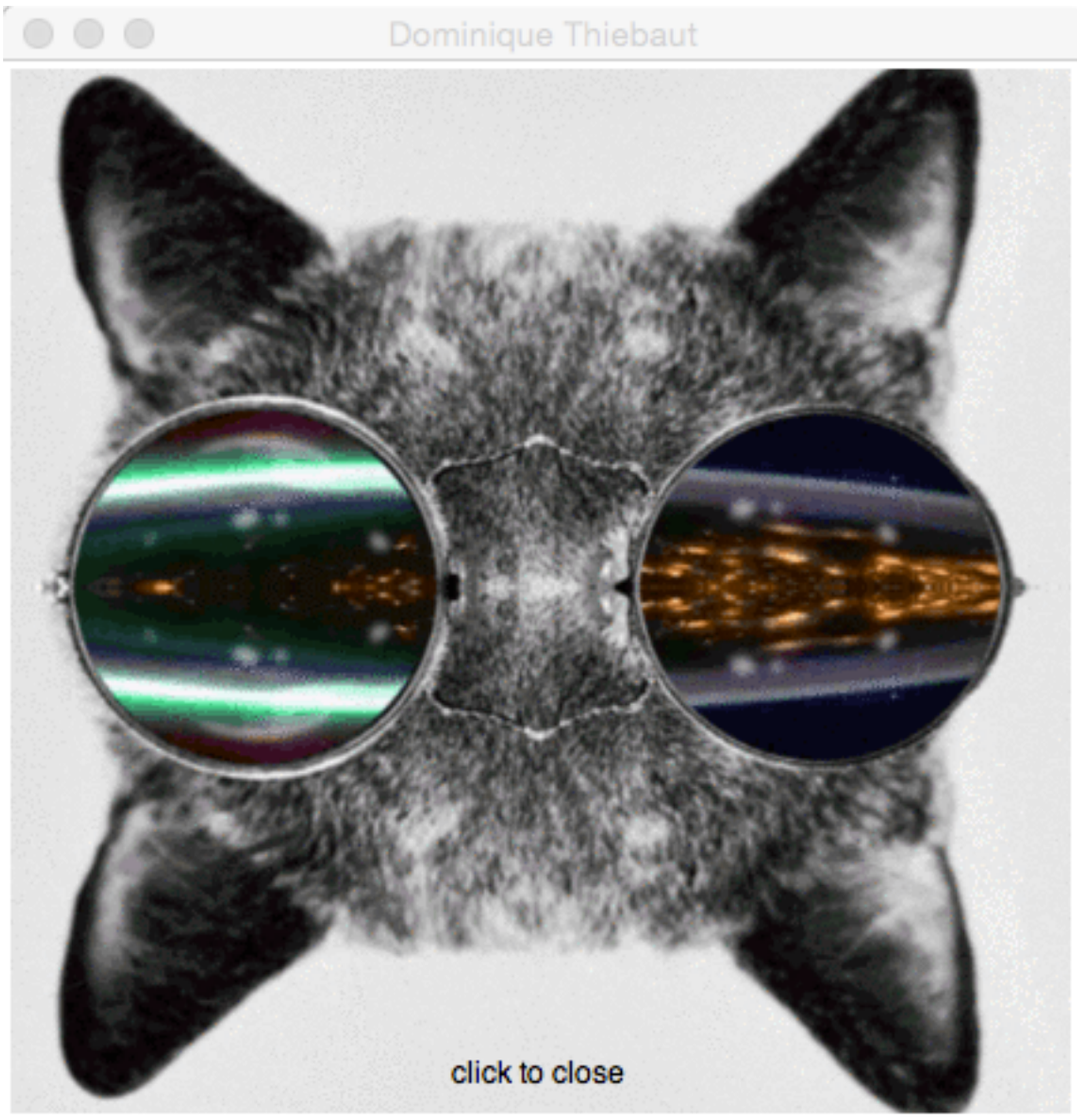
imageProcessingCopy.py - /Users/thiebaut/Desktop/Dropbox/111/imageProcessingCopy.py (3.5.4)

```
def copy( win, img, srcX, offset, width ):
    '''copy part of the image starting at srcX, width pixels
    horizontally, at srcX+offset. Copies whole height of pixels.'''
    for x in range( srcX, srcX + width ):
        win.update()
        for y in range( 0, HEIGHT ):
            r,g,b = img.getPixel( x, y )
            color = color_rgb( r,g,b )
            img.setPixel( x+offset, y, color )

def main():
    # open the window
    win = GraphWin( "CSC111 Image Viewer", WIDTH, HEIGHT )

    # get the file name from the user
    fileName = "catHatLong.gif" #input( "Image name? " )
    print()

    # open the cat image
    cat = Image( Point(WIDTH//2, HEIGHT//2), fileName )
    cat.draw( win )

    # replicate the cat, from 0 to 1 CATWIDTH away,
    # copy CATWIDTH pixels, horizontally
    copy( win, cat, 0, CATWIDTH, CATWIDTH )
    copy( win, cat, 0, CATWIDTH*2, CATWIDTH )
```

Ln: 18 Col: 34

Transformations to Try

- Mirror top half of cat image



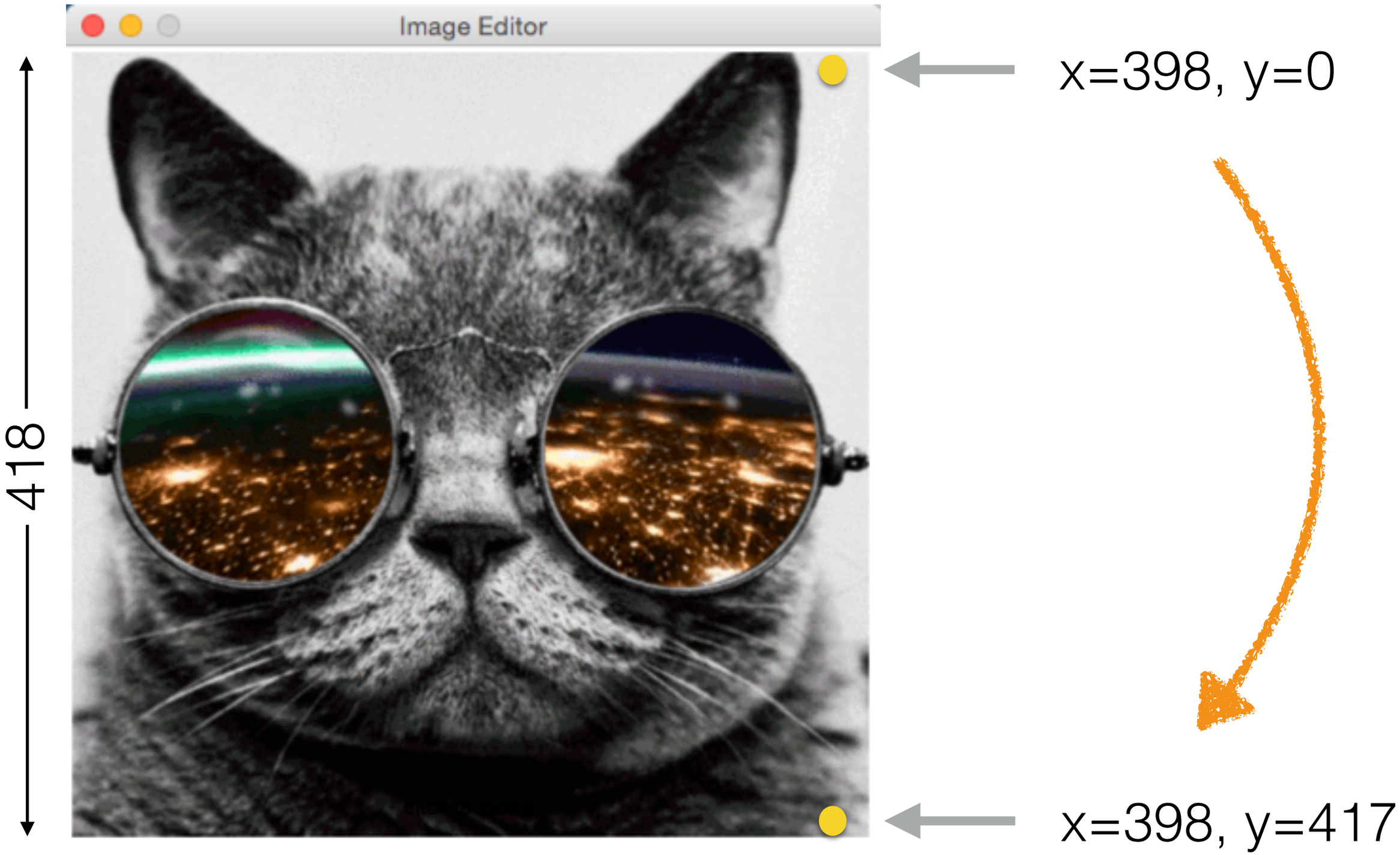


Image Editor
x=398, y=0
418
x=398, y=417

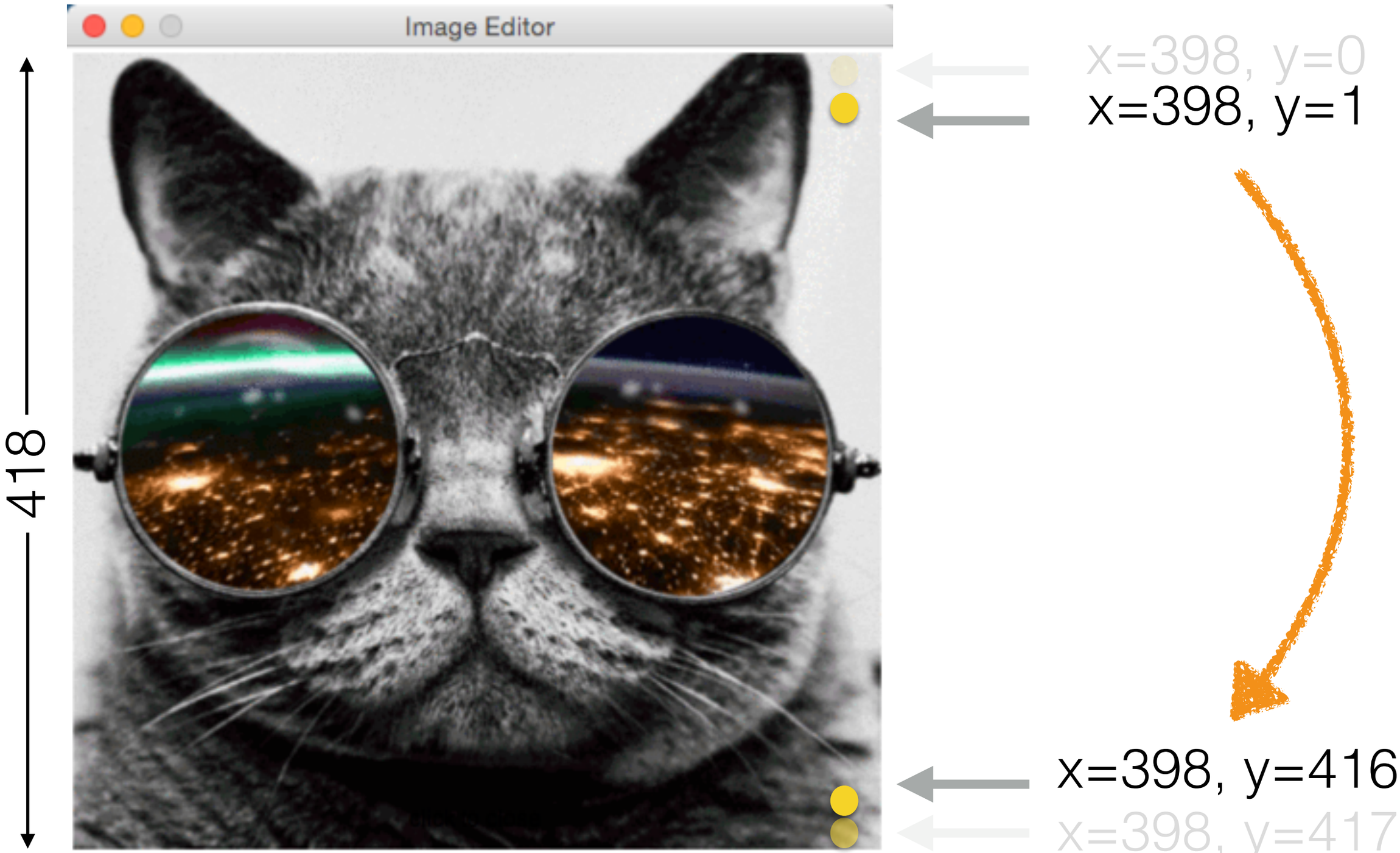


Image Editor
416
x=398, y=0
x=398, y=1
x=398, y=416
x=398, y=417

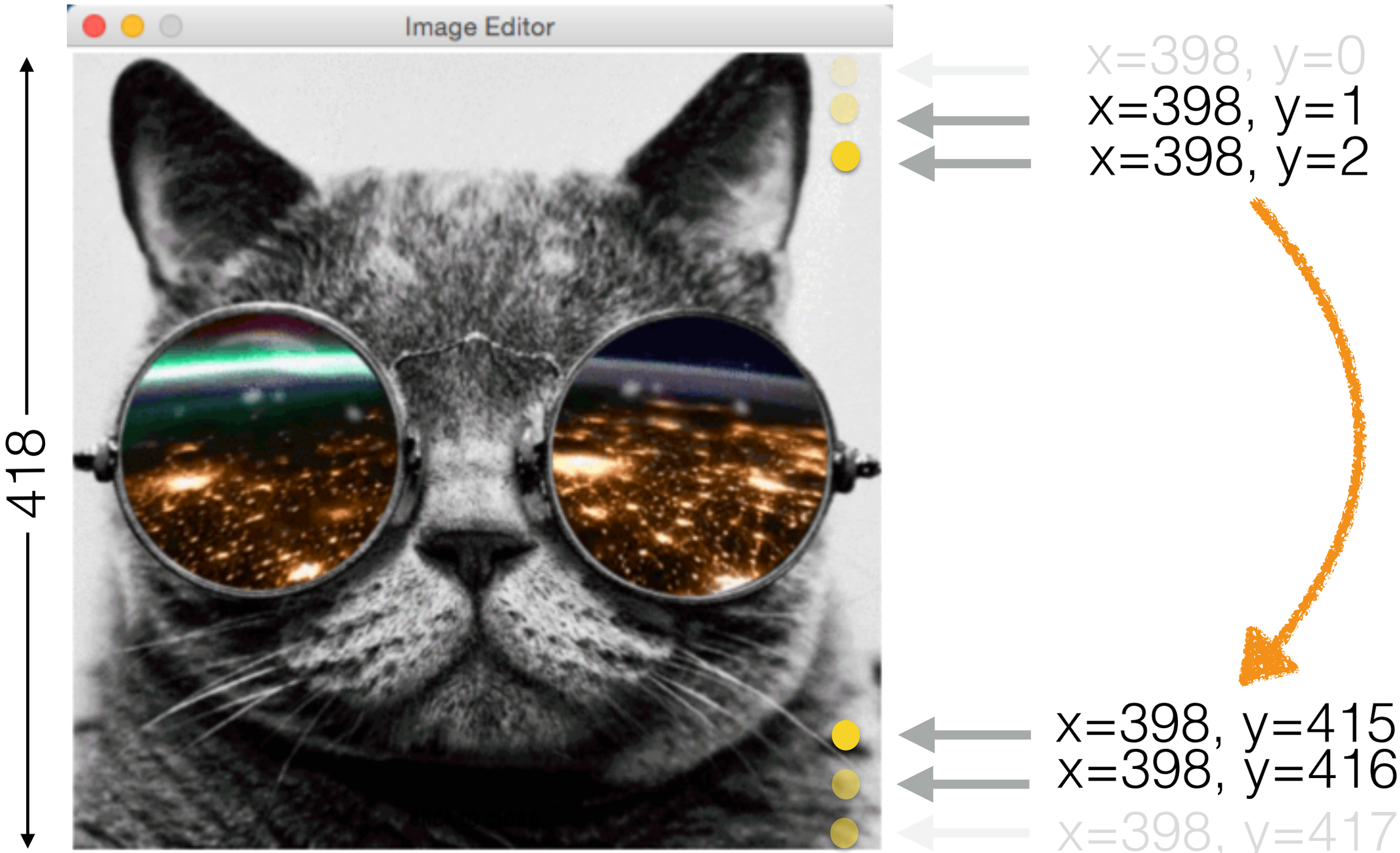


Image Editor
416
x=398, y=0
x=398, y=1
x=398, y=2
x=398, y=415
x=398, y=416
x=398, y=417

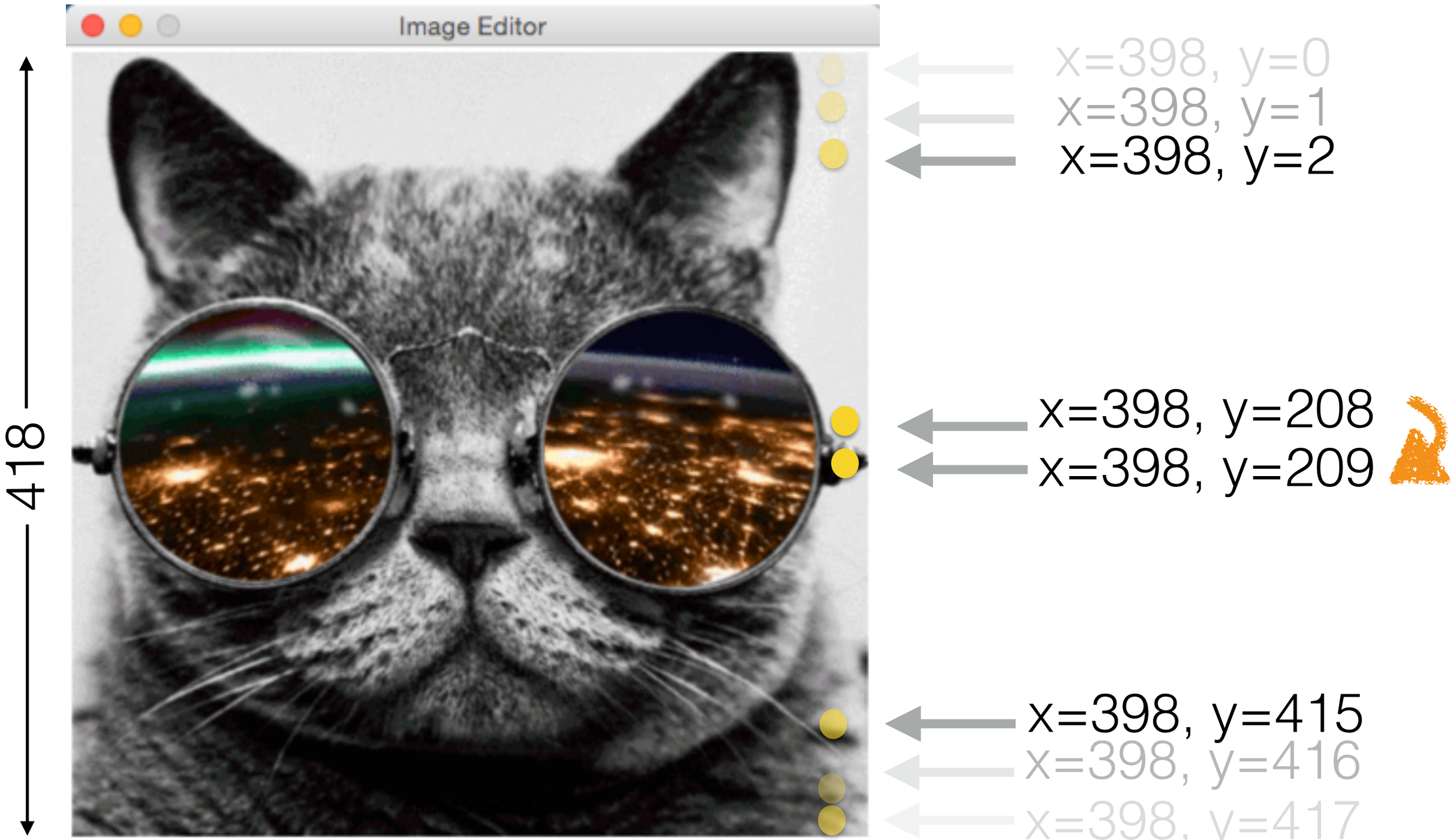
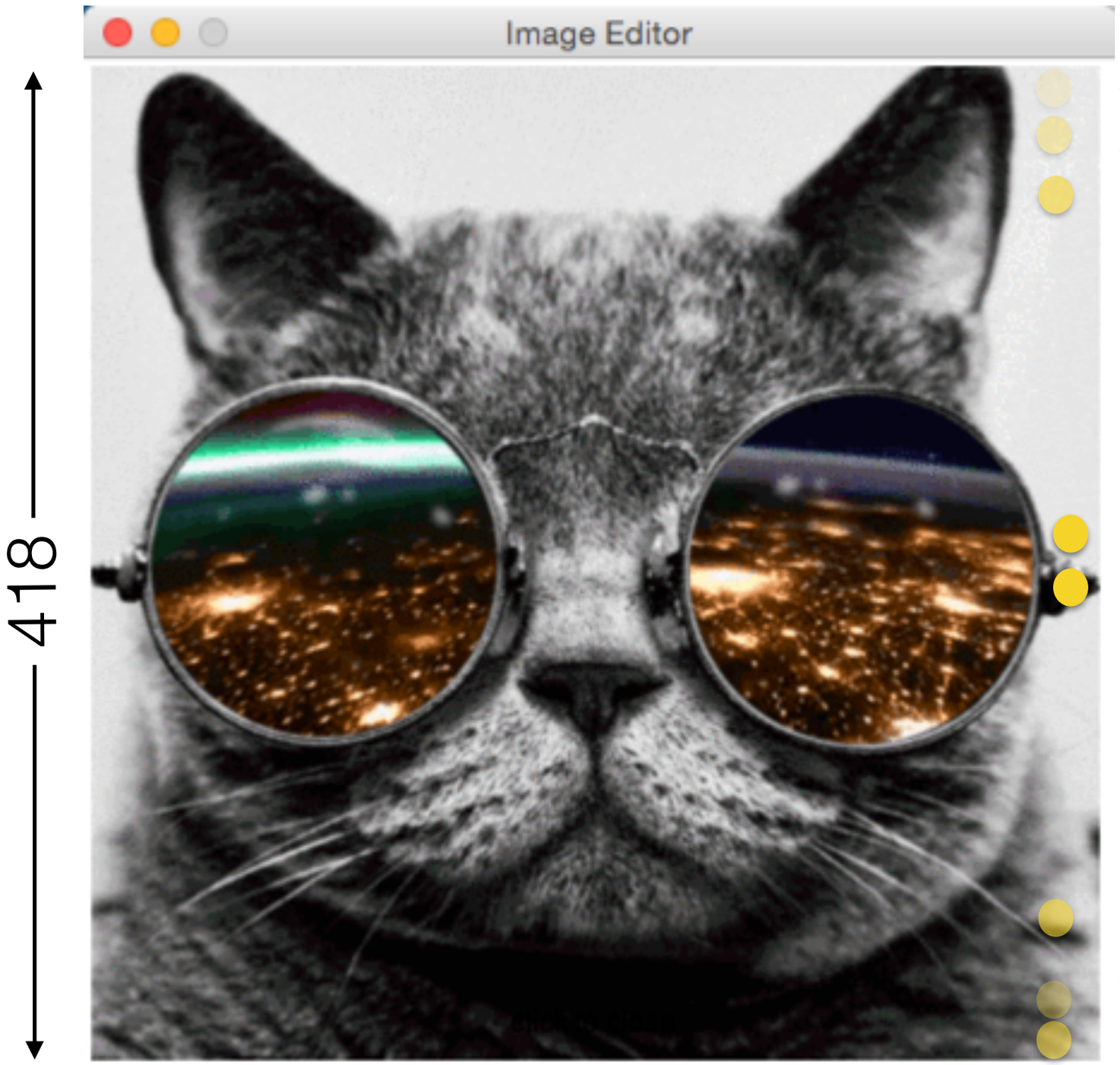
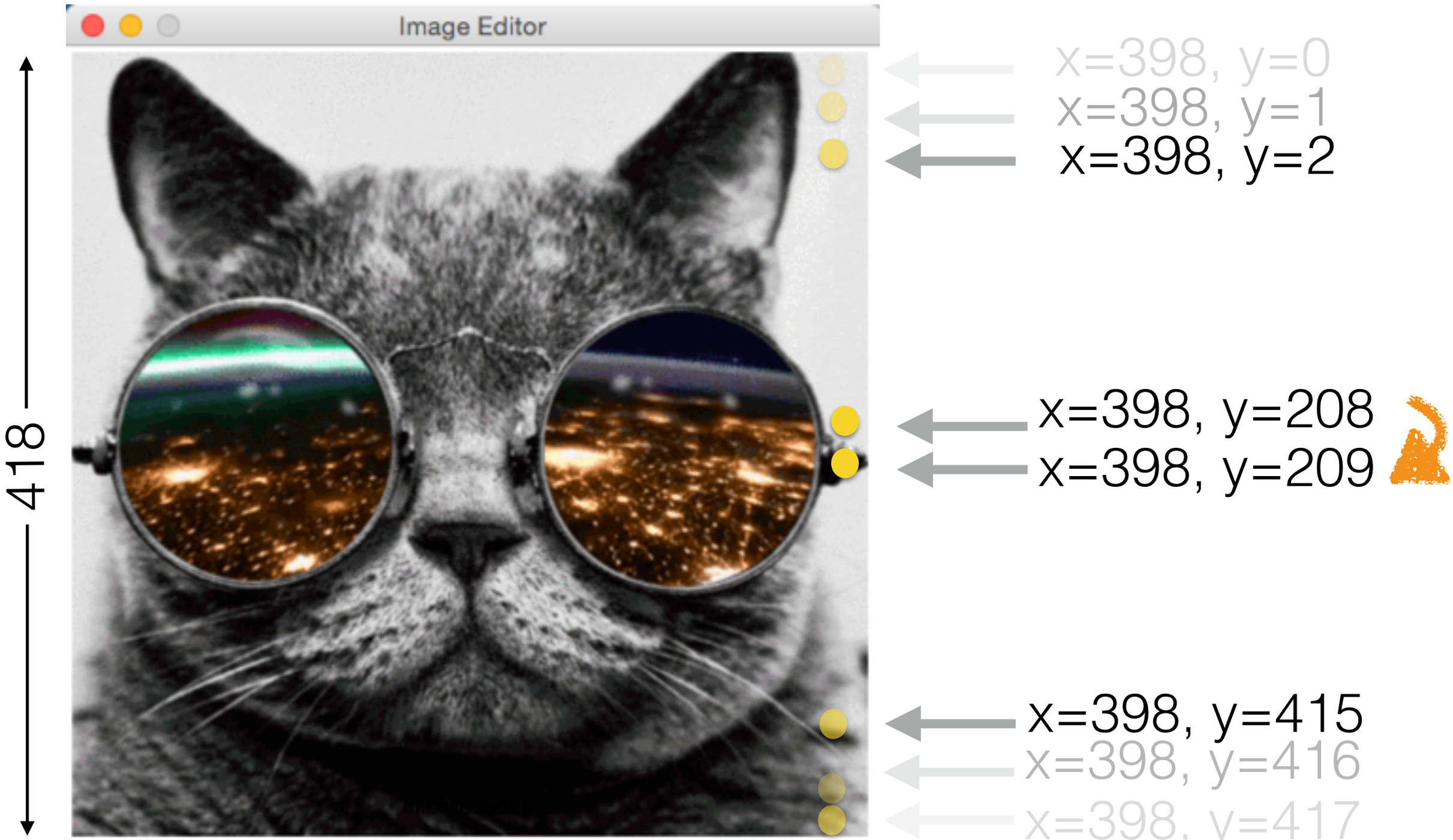


Image Editor
418
x=398, y=0
x=398, y=1
x=398, y=2
x=398, y=208
x=398, y=209
x=398, y=415
x=398, y=416
x=398, y=417



Observation 1: y_source + y_destination = 417, always



Observation 2: 417 = image height - 1

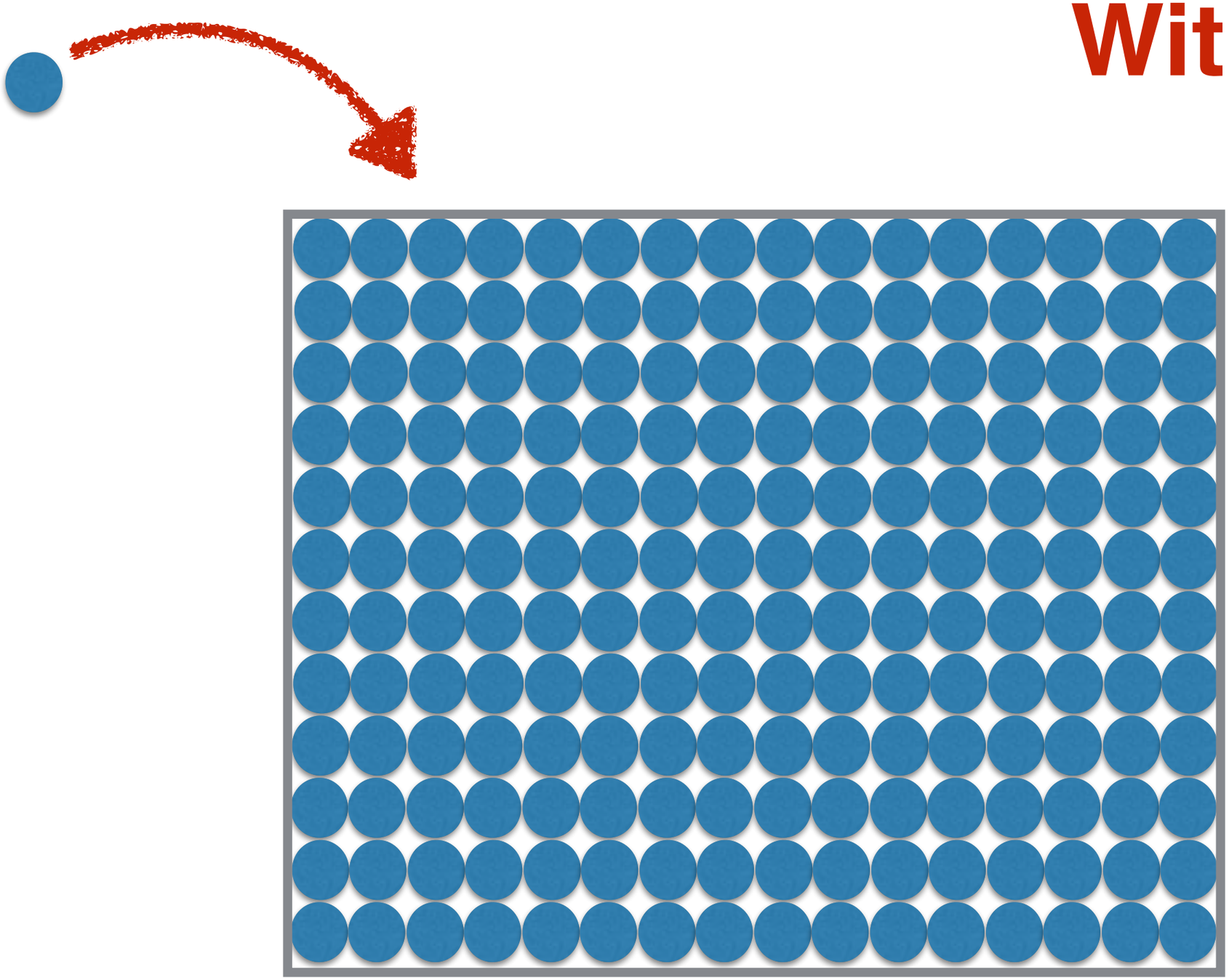
Demo Time!

imageProcessingMirror.py - /Users/thiebaut/Desktop/Dropbox/111/imageProcessingMirror.py (3.5.4)

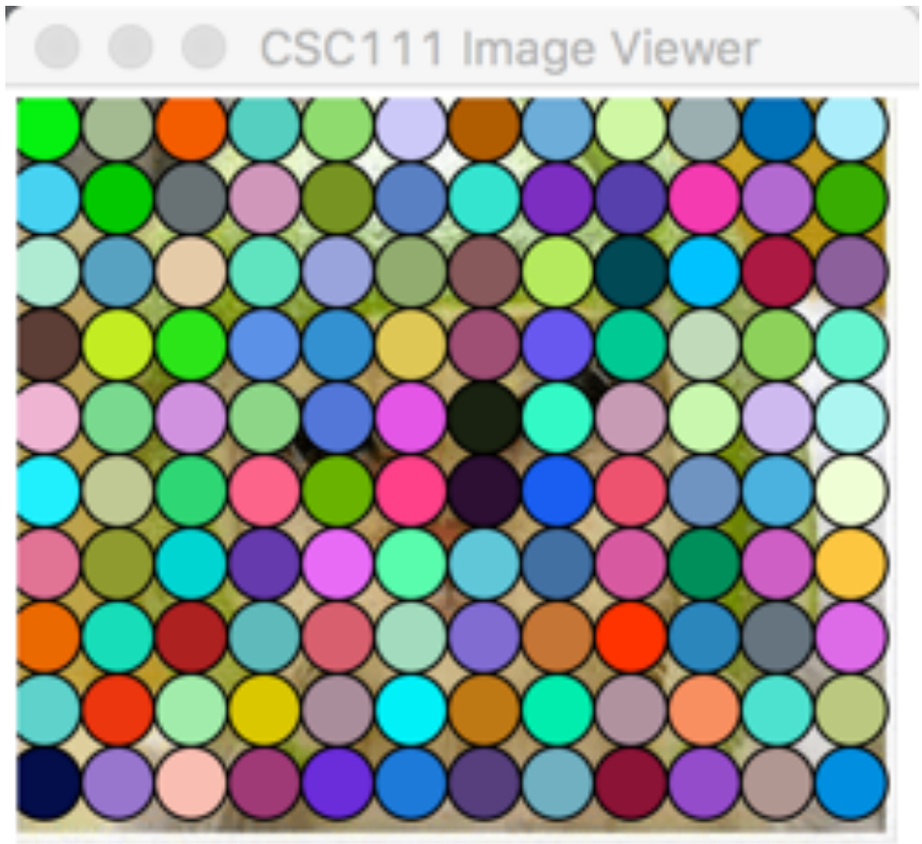
```
def mirror( img, win ):
    '''mirror an image: top horizontal half is replicated,
    upside down'''
    global WIDTH, HEIGHT
    for y in range((HEIGHT-1)//2 ):
        win.update()
        for x in range( WIDTH ):
            r, g, b = img.getPixel( x, y )
            img.setPixel( x, HEIGHT-1-y, color_rgb(r,g,b) )

def main():
    # open the window
    win = GraphWin( "CSC111 Image Viewer", WIDTH, HEIGHT )

    # open the cat image
    cat = Image( Point(WIDTH//2, HEIGHT//2), FILENAME )
    cat.draw( win )

    # mirror the cat
    mirror( cat, win )

    # wait for user to click the mouse...
    win.getMouse()
```

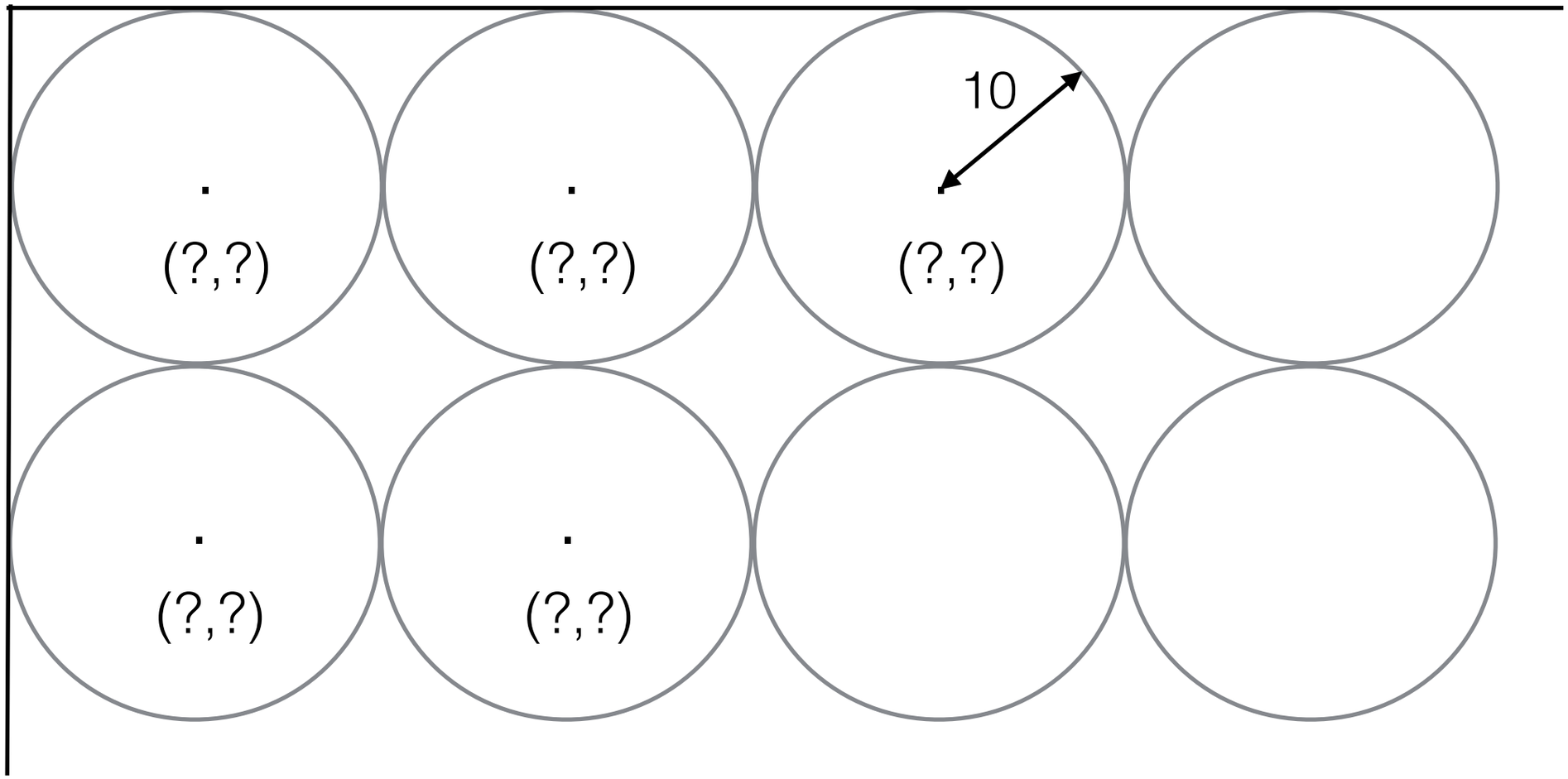
Ln: 18 Col: 0

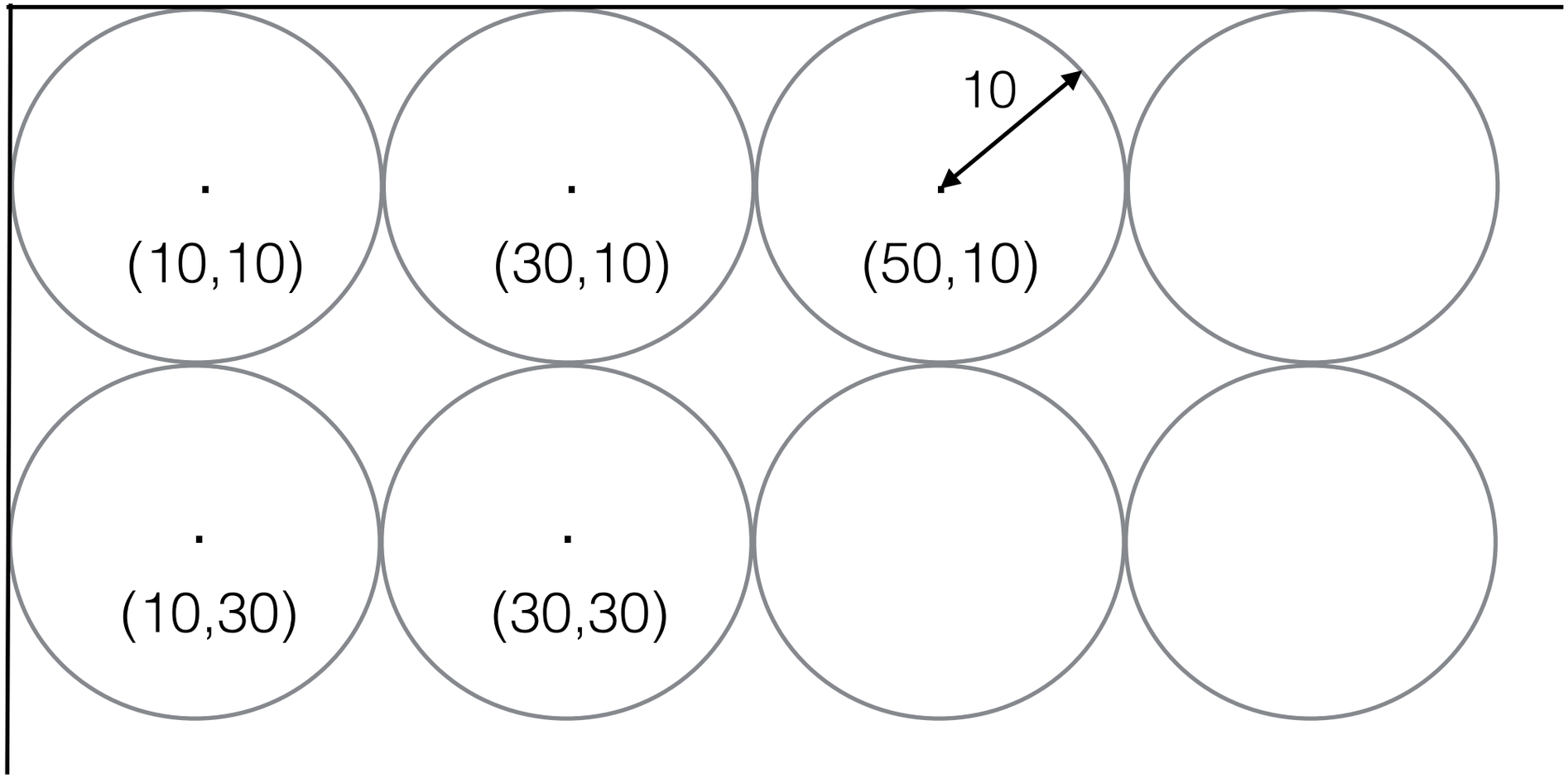
Covering Up Window With Shapes



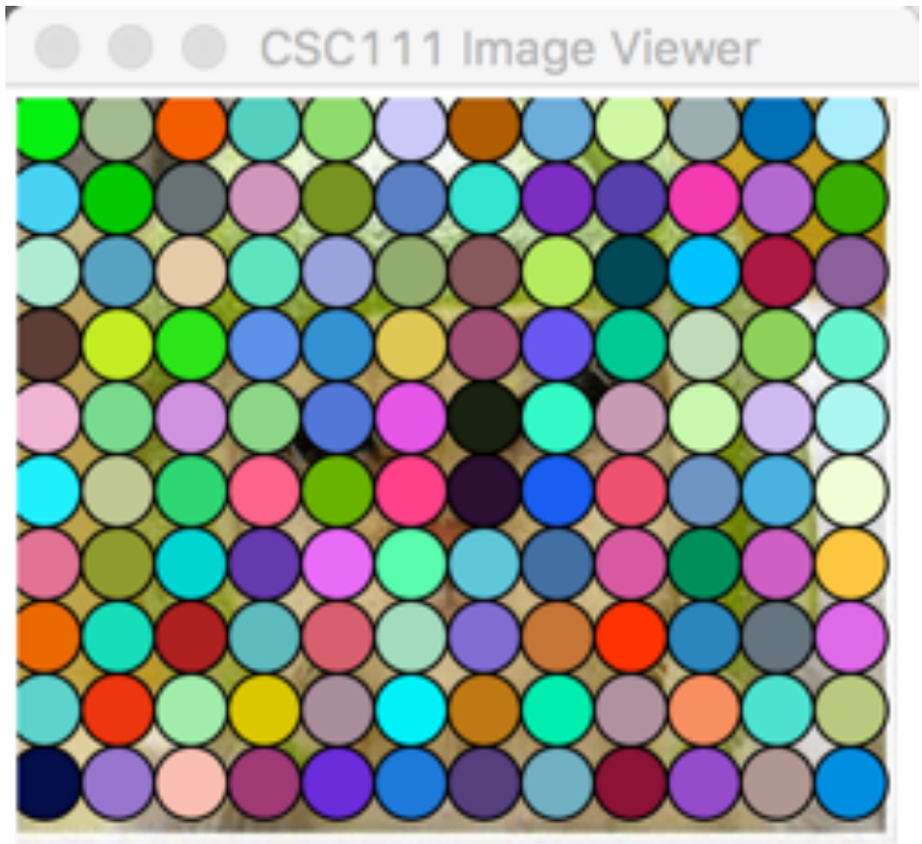
Covering Up Window With Shapes




10
(?,?)
(?,?)
(?,?)
(?,?)
(?,?)

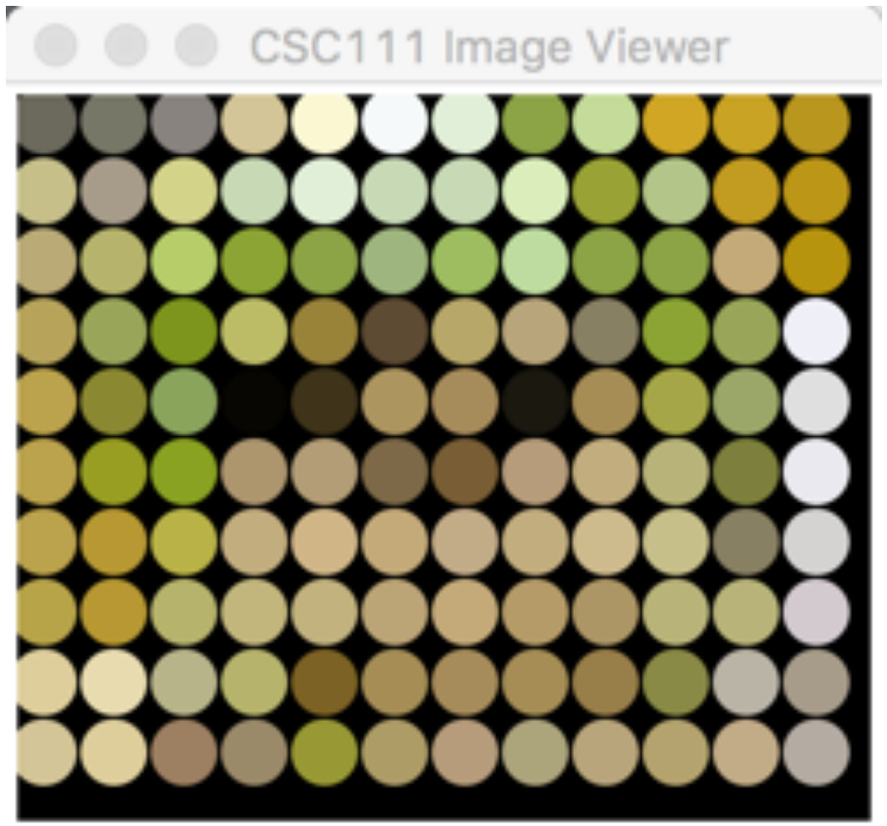

10
(10,10)
(30,10)
(50,10)
(10,30)
(30,30)

Covering Up Window With Shapes



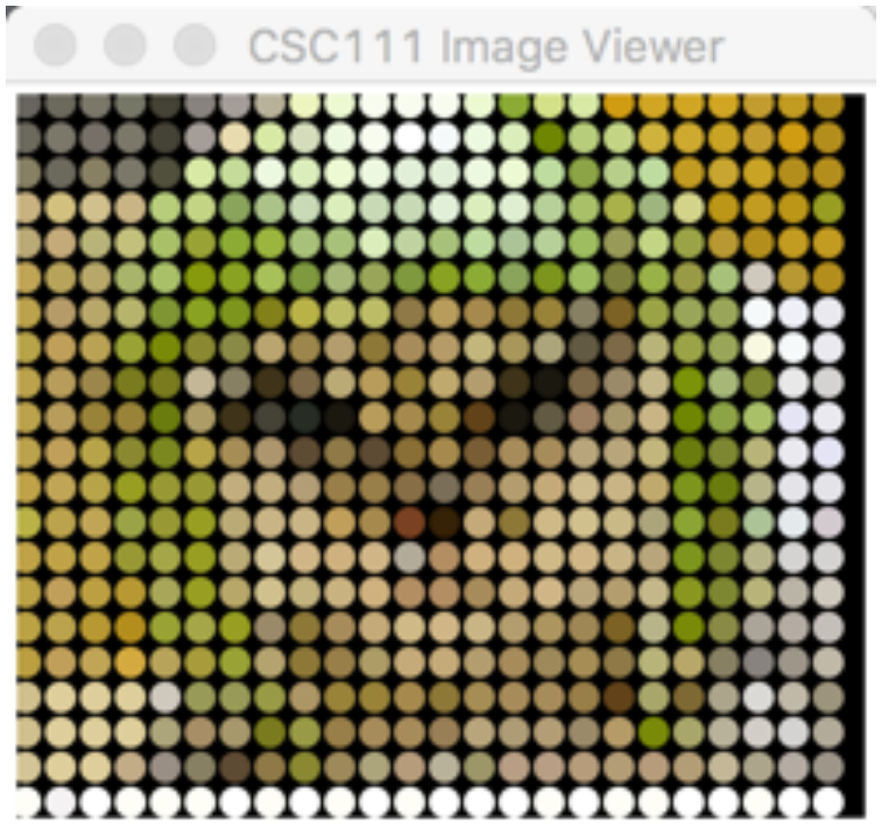
Let's code it!

Covering Up Window With Shapes



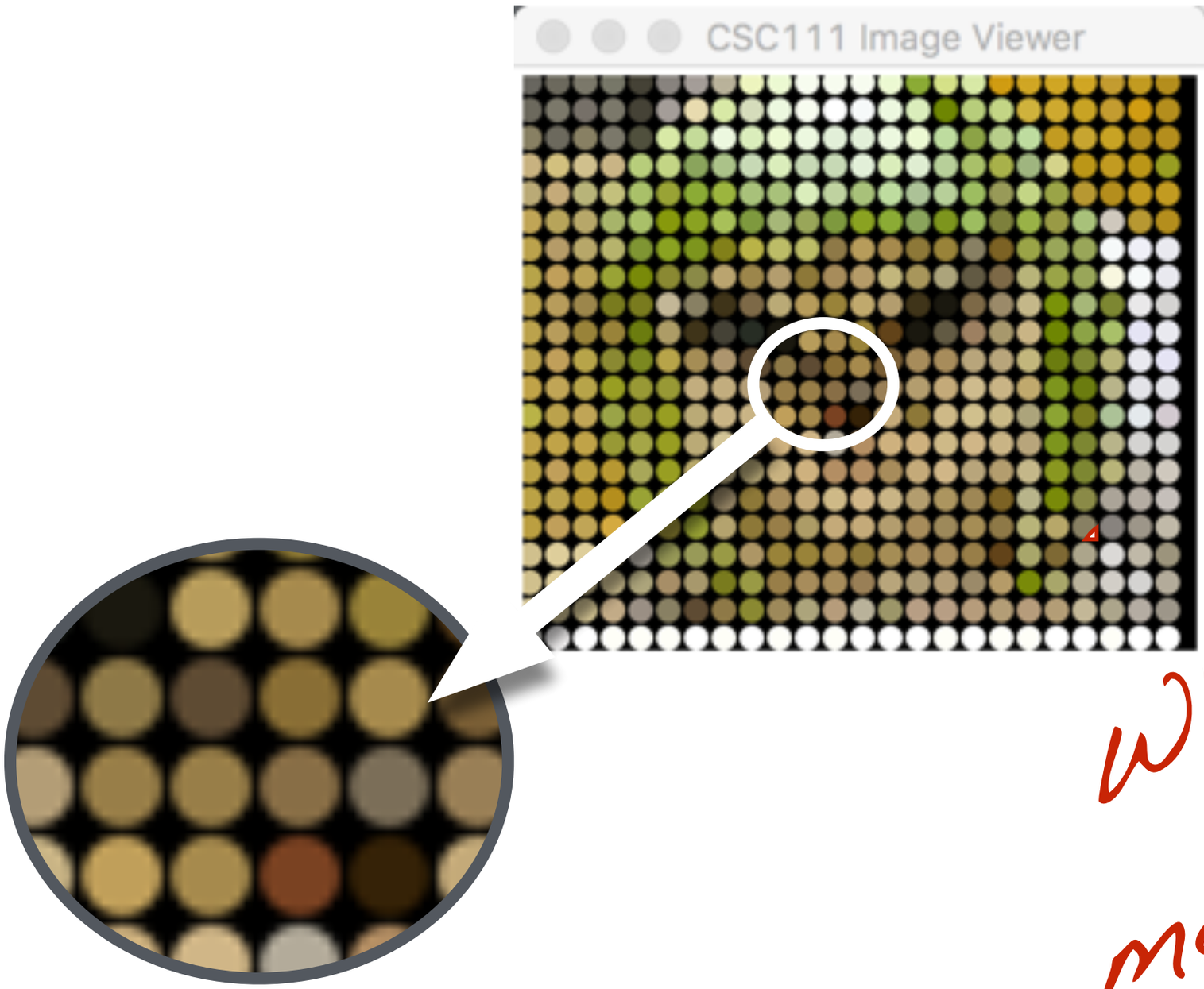
With some more coding...

Covering Up Window With Shapes



With some more coding...

Covering Up Window With Shapes



with some more coding...

- Mirroring an Image
- Displaying a Checker Board
  - **8x8 Grid**
  - **Alternating Colors**
  - **Creating a Class for a Checkers Piece**
  - **Using a Gif Image for a Piece**

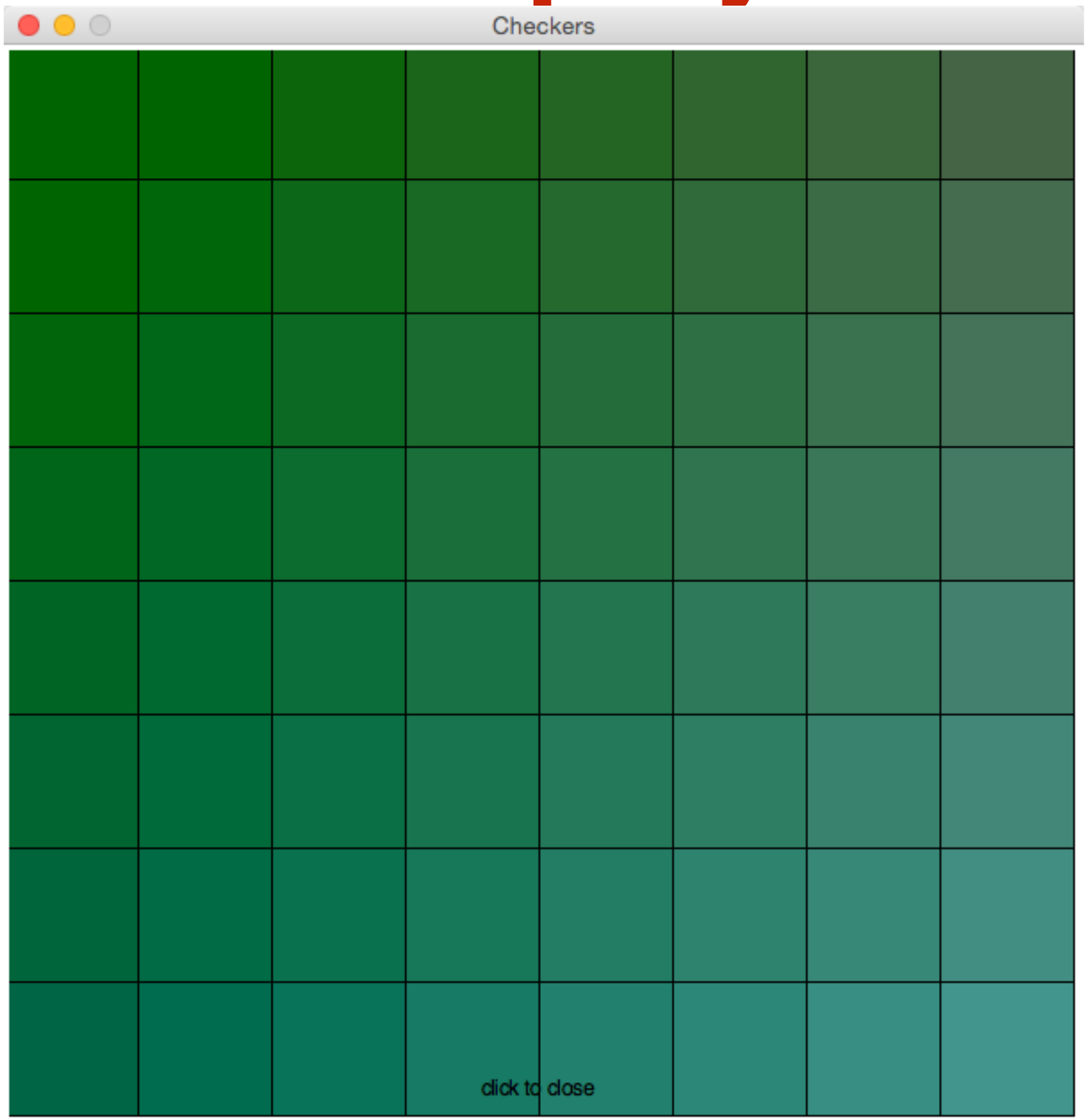
Graphic Problem of the Day: “Playing” Checkers

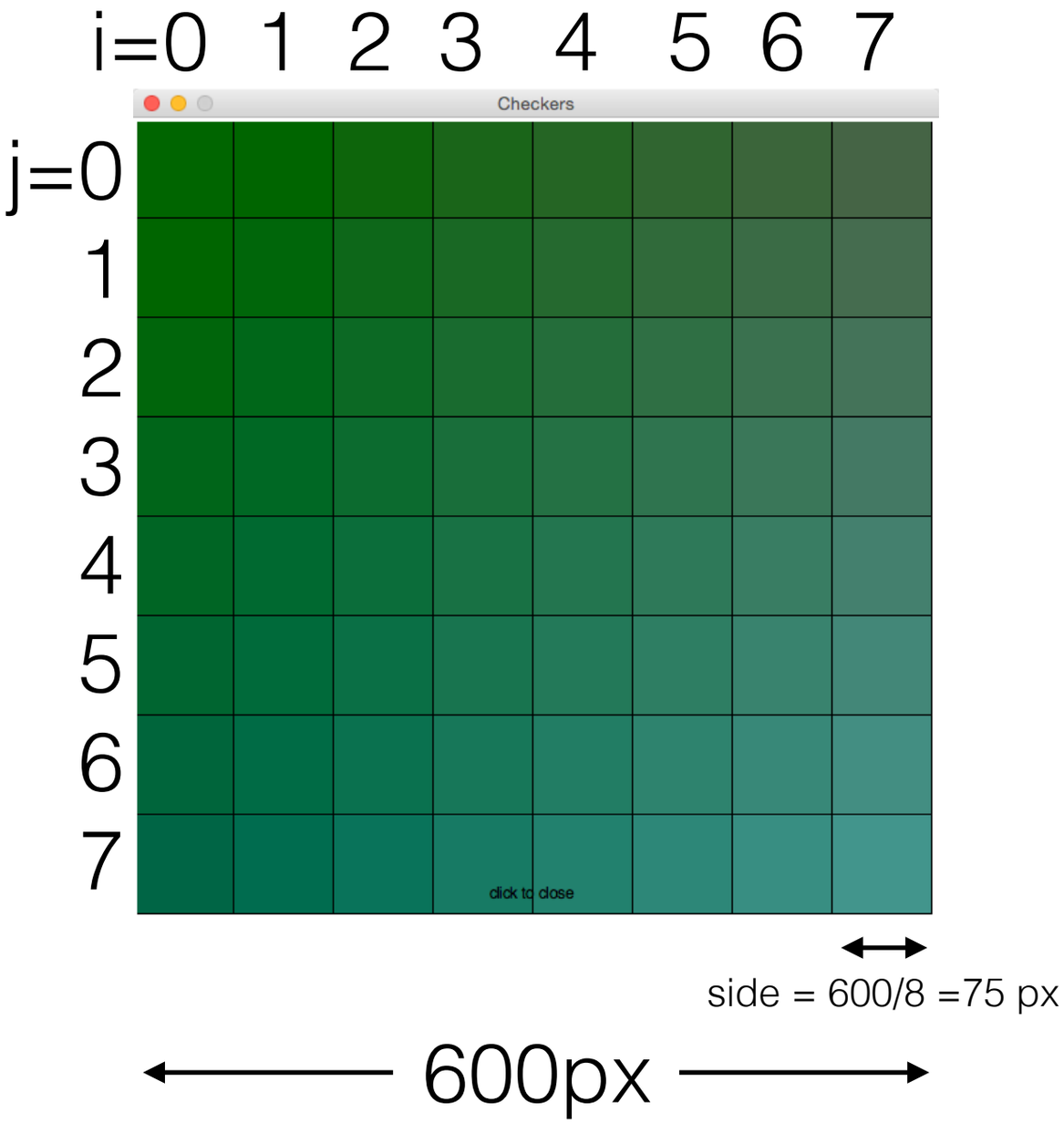
Image credit: http://www.freeimageslive.co.uk/free_stock_image/checkersjpg

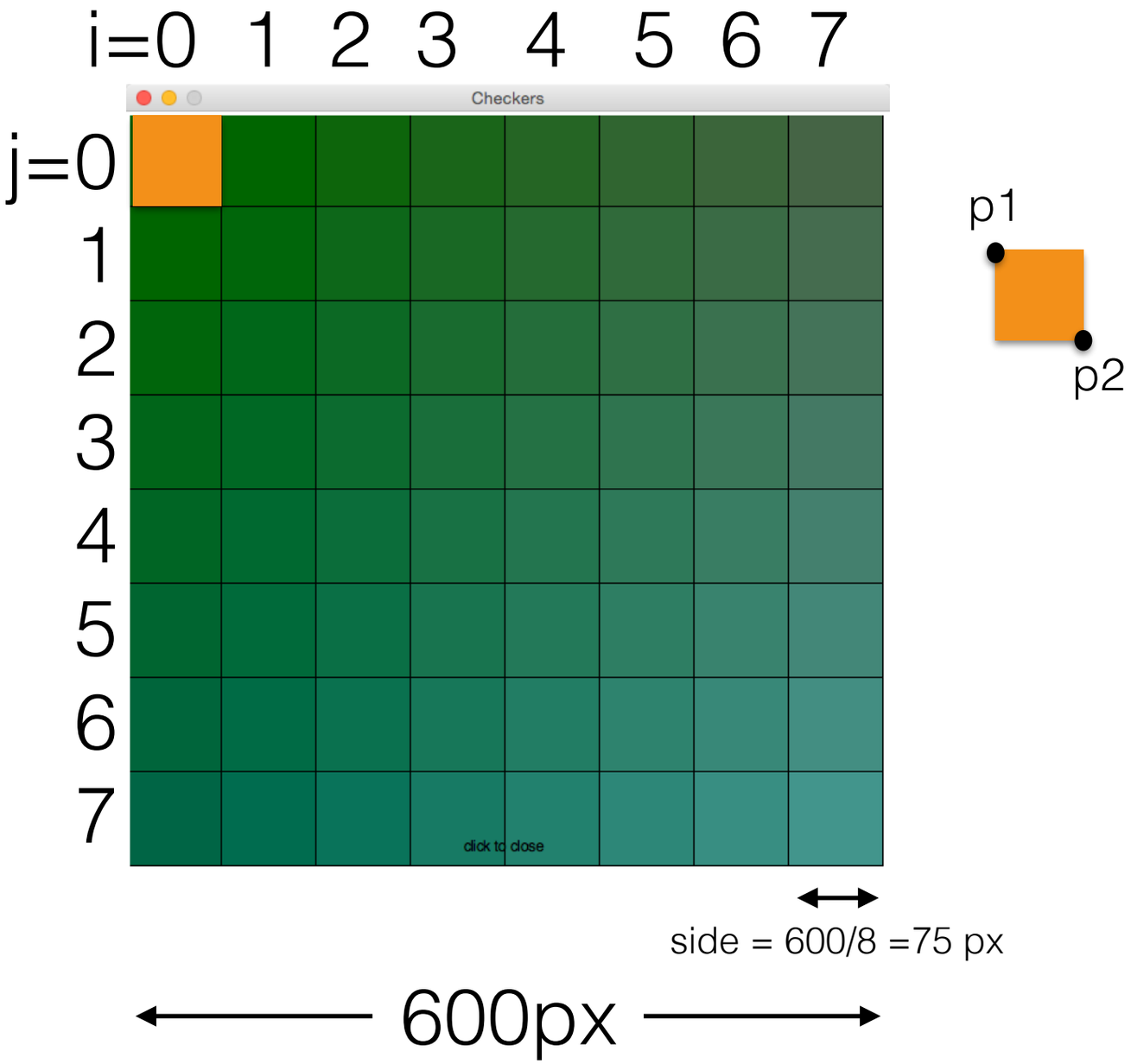
Problems to Solve:

- **Display** 8x8 **board** with alternating colors
- Generate the graphics for a **piece** (white and black circular shapes)
- **Display** the board with the black and white **pieces**
- Improve game with **real images**

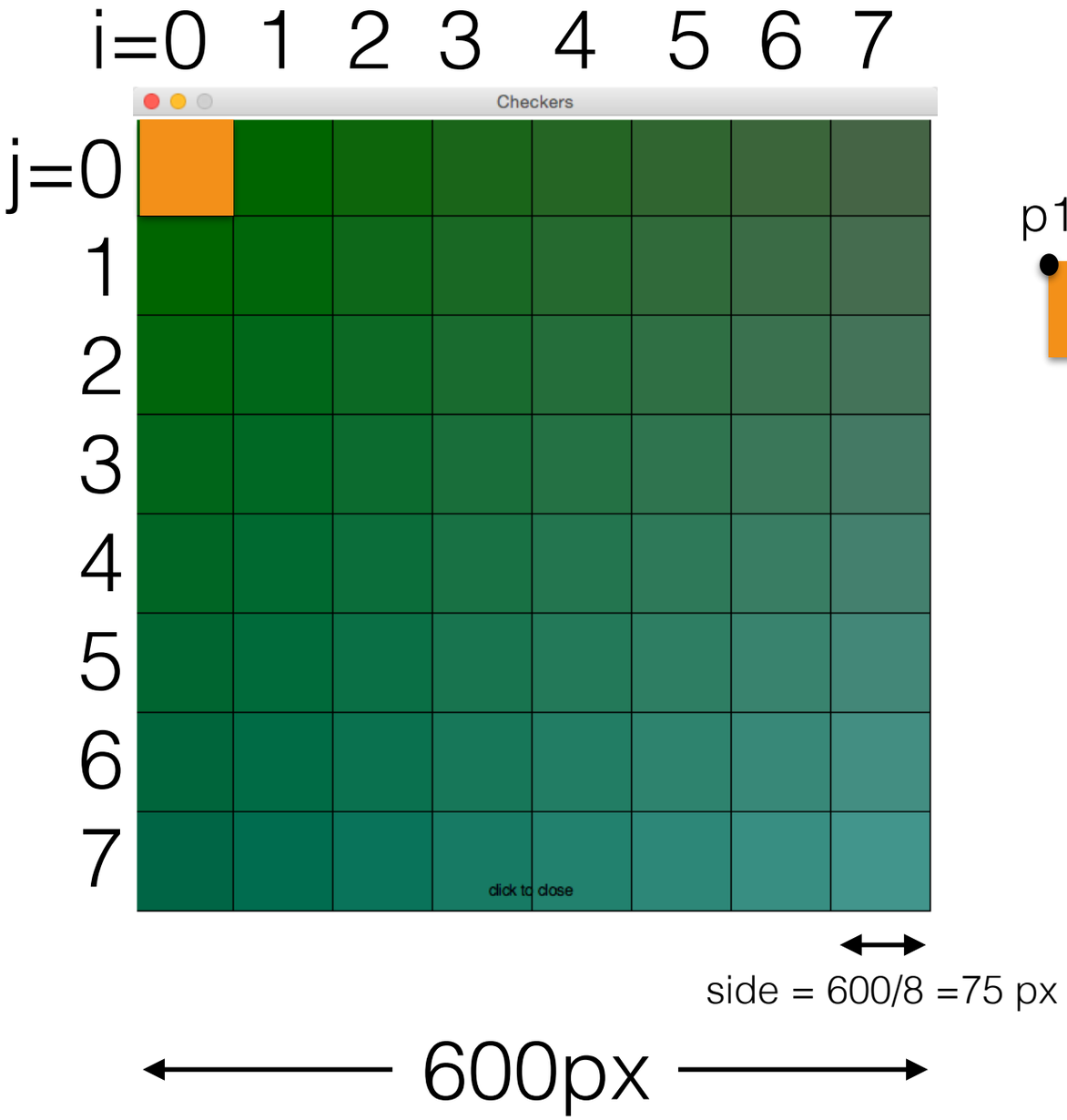
Display 8x8 Board

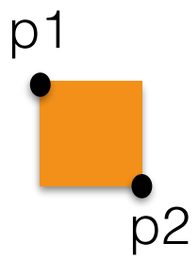



i=0 1 2 3 4 5 6 7
Checkers
j=0
1
2
3
4
5
6
7
click to close
side = 600/8 =75 px
600px

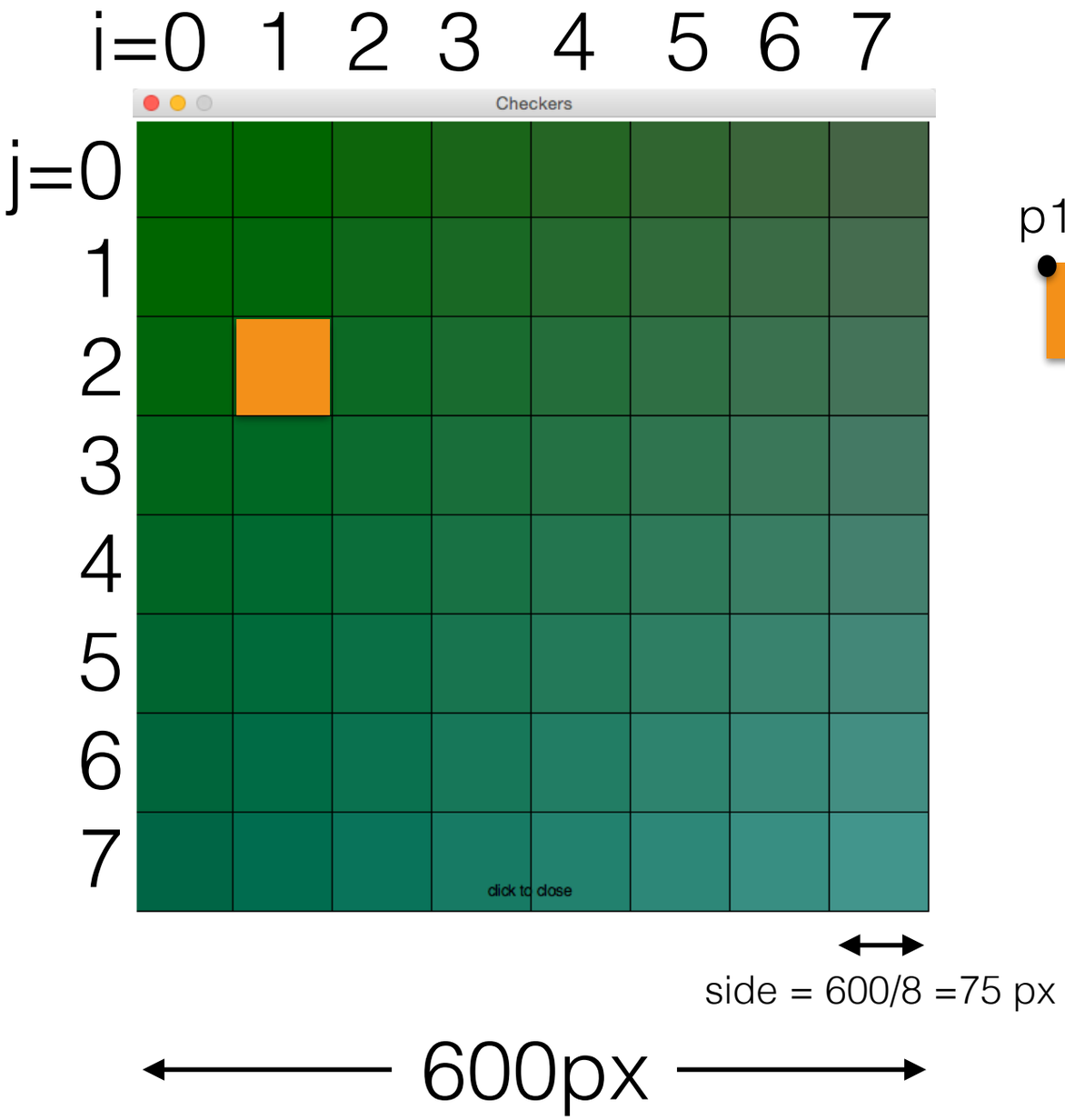

i=0 1 2 3 4 5 6 7
Checkers
j=0
1
2
3
4
5
6
7
click to close
side = 600/8 =75 px
600px

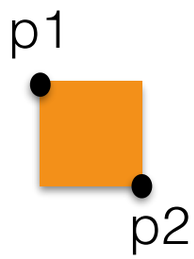
p1
p2



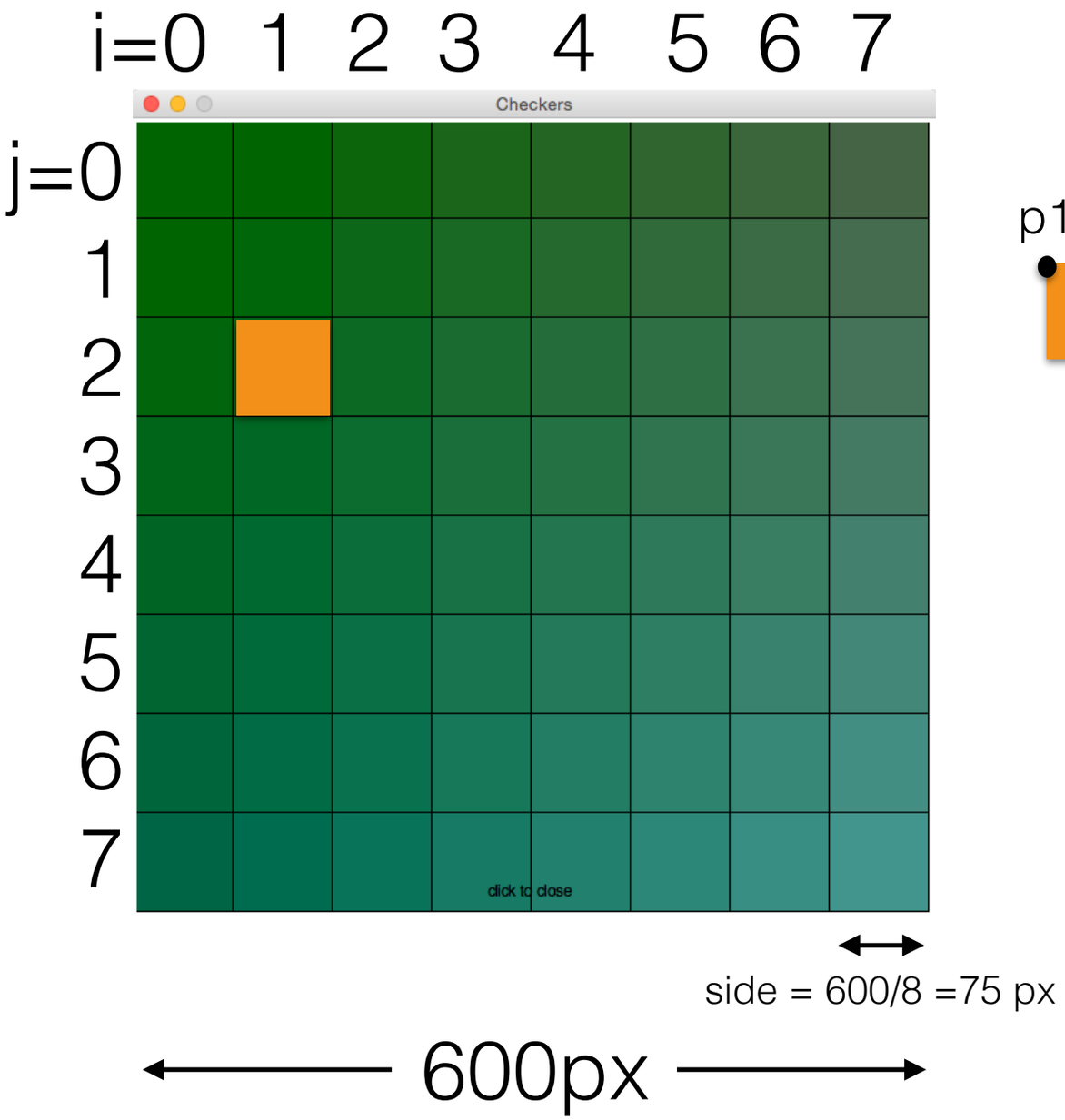


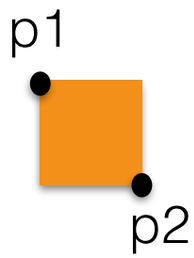

```
# i is 0, j is 0

p1 = Point( ?, ? )
p2 = Point( ?, ? )

rect = Rectangle( p1, p2 )
rect.setFill( "green" )
rect.draw( win )
```



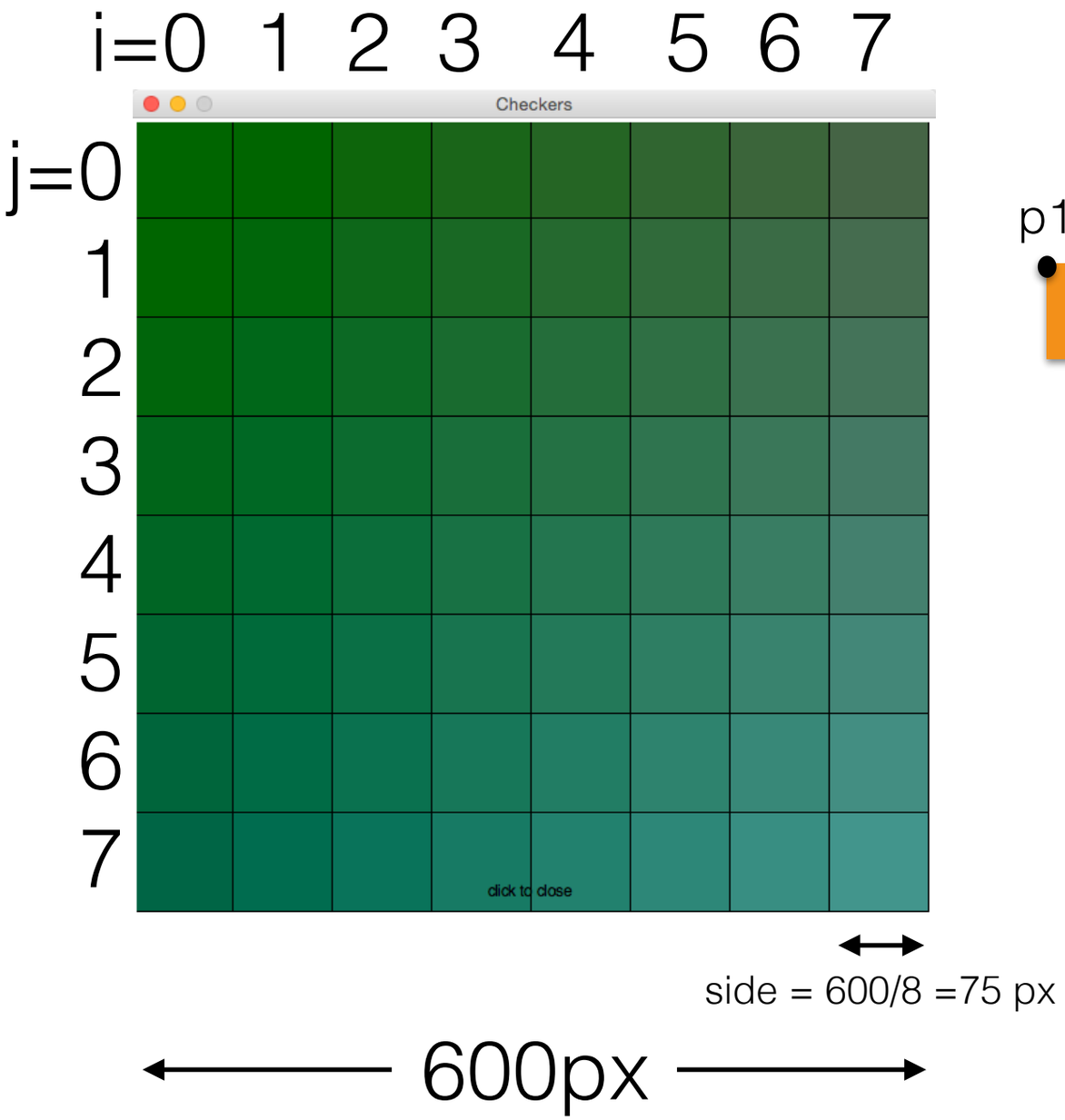


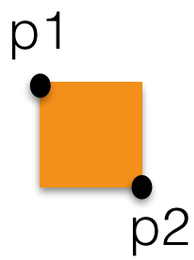

```
# i is 1, j is 2

p1 = Point( ?, ? )
p2 = Point( ?, ? )

rect = Rectangle( p1, p2 )
rect.setFill( "green" )
rect.draw( win )
```





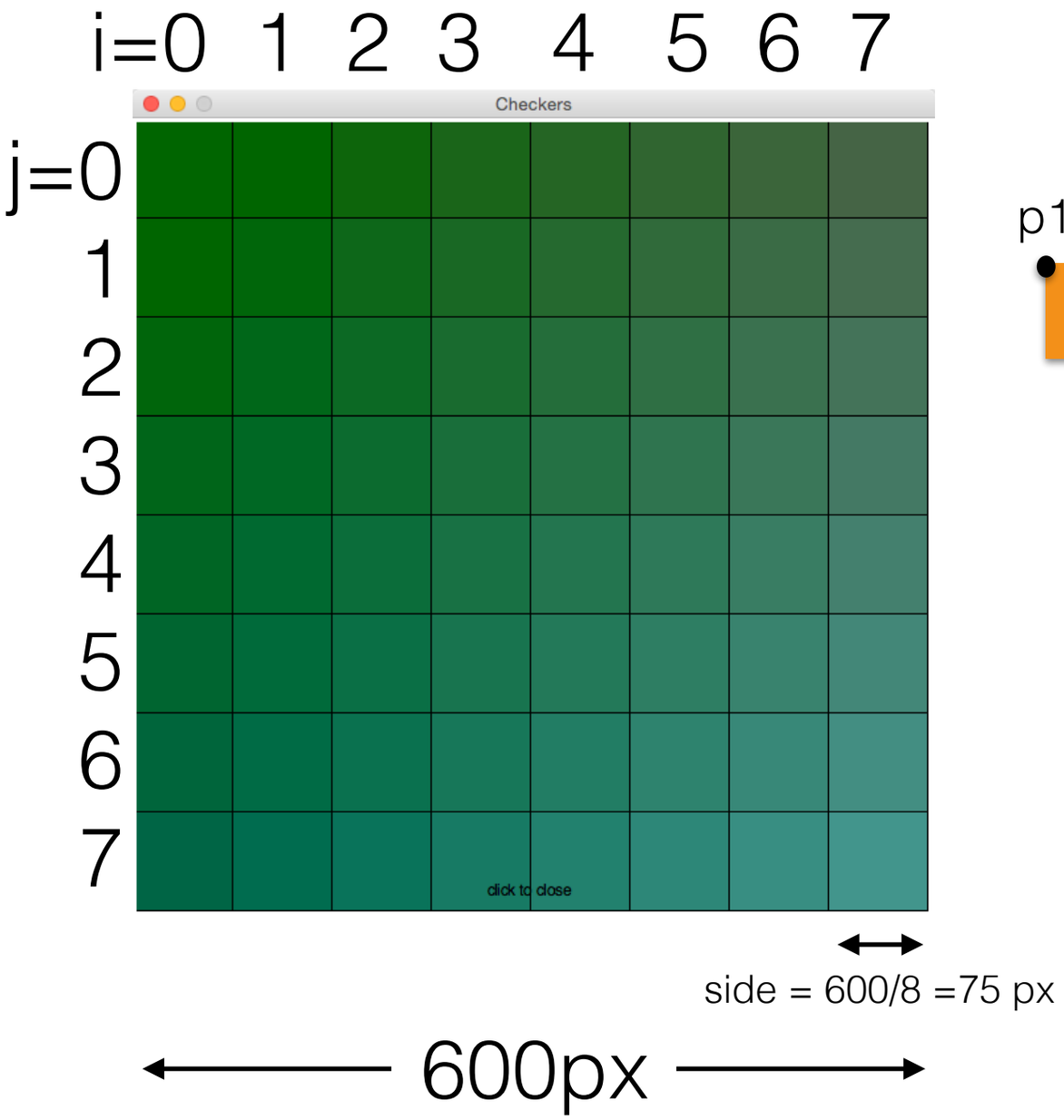

```
# i is 1, j is 2
x = i * side
y = j * side
p1 = Point( x, y )
p2 = Point( x+side, y+side )

rect = Rectangle( p1, p2 )
rect.setFill( "green" )
rect.draw( win )
```

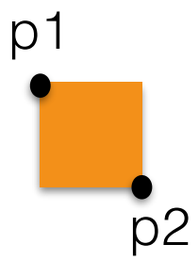





```
for i in range( 8 ):
    for j in range( 8 ):
        x = i * side
        y = j * side
        p1 = Point( x, y )
        p2 = Point( x+side, y+side )

        rect = Rectangle( p1, p2 )
        rect.setFill( "green" )
        rect.draw( win )
```

checkers0.py






```
for i in range( 8 ):
    for j in range( 8 ):
        x = i * side
        y = j * side
        p1 = Point( x, y )
        p2 = Point( x+side, y+side )

        rect = Rectangle( p1, p2 )
        rect.setFill( "green" )
        rect.draw( win )
```

For fun, replace with:
rect.setFill(color_rgb(255-(i+j)*10, i*j*3, 255-i*j*3))

- Mirroring an Image
- Displaying a Checker Board
  - **8x8 Grid**
  - **Alternating Colors**
  - **Creating a Class for a Checkers Piece**
  - **Using a Gif Image for a Piece**

Alternating Black and White Cells

Checkers

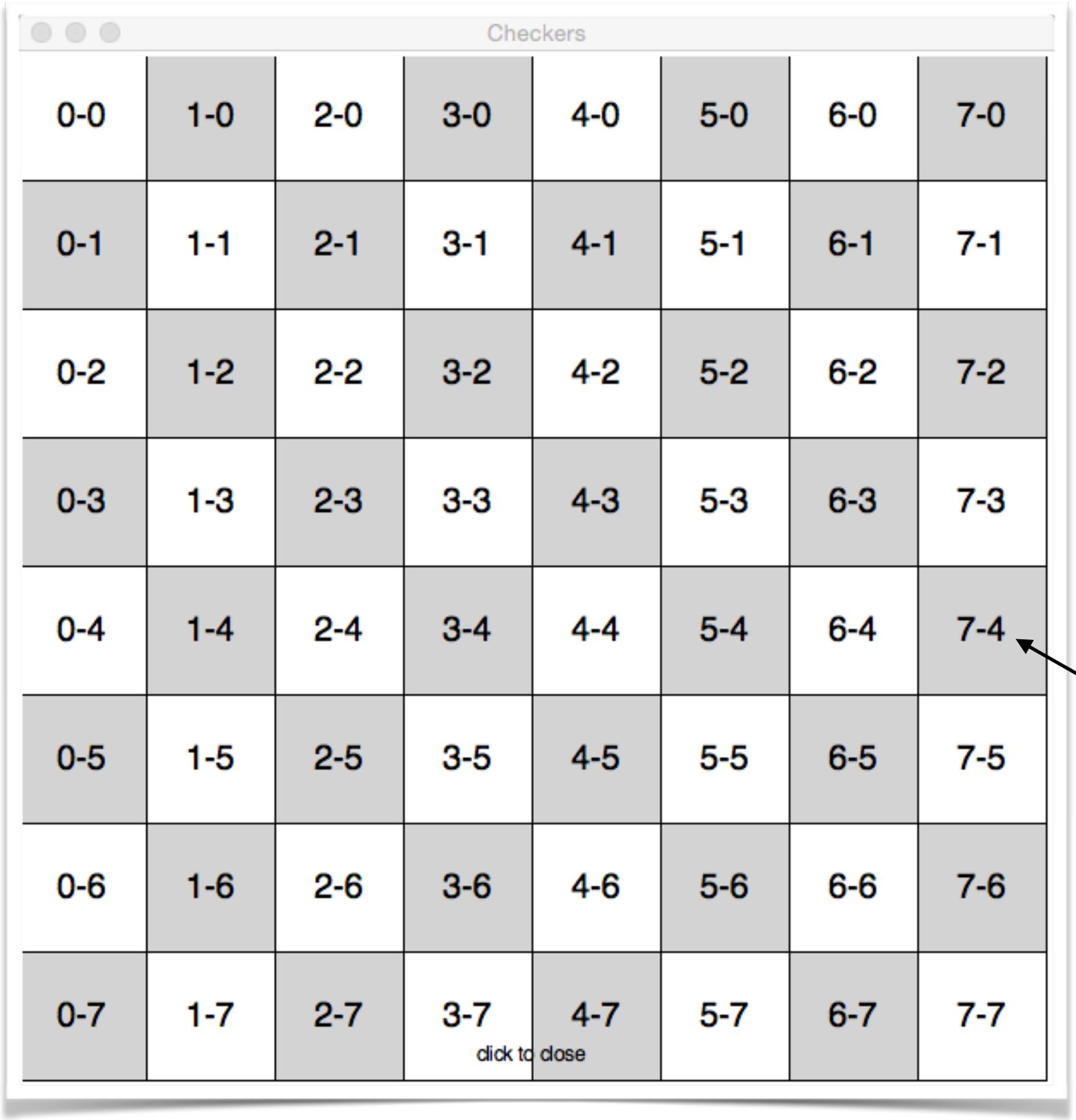
0-0	1-0	2-0	3-0	4-0	5-0	6-0	7-0
0-1	1-1	2-1	3-1	4-1	5-1	6-1	7-1
0-2	1-2	2-2	3-2	4-2	5-2	6-2	7-2
0-3	1-3	2-3	3-3	4-3	5-3	6-3	7-3
0-4	1-4	2-4	3-4	4-4	5-4	6-4	7-4
0-5	1-5	2-5	3-5	4-5	5-5	6-5	7-5
0-6	1-6	2-6	3-6	4-6	5-6	6-6	7-6
0-7	1-7	2-7	3-7	4-7	5-7	6-7	7-7

click to close

i - j

"i dash j"

Alternating Black and White Cells



checkers1.py - /Users/thiebaut/Desktop/Dropbox/111/checkers1.py (3.5.4)

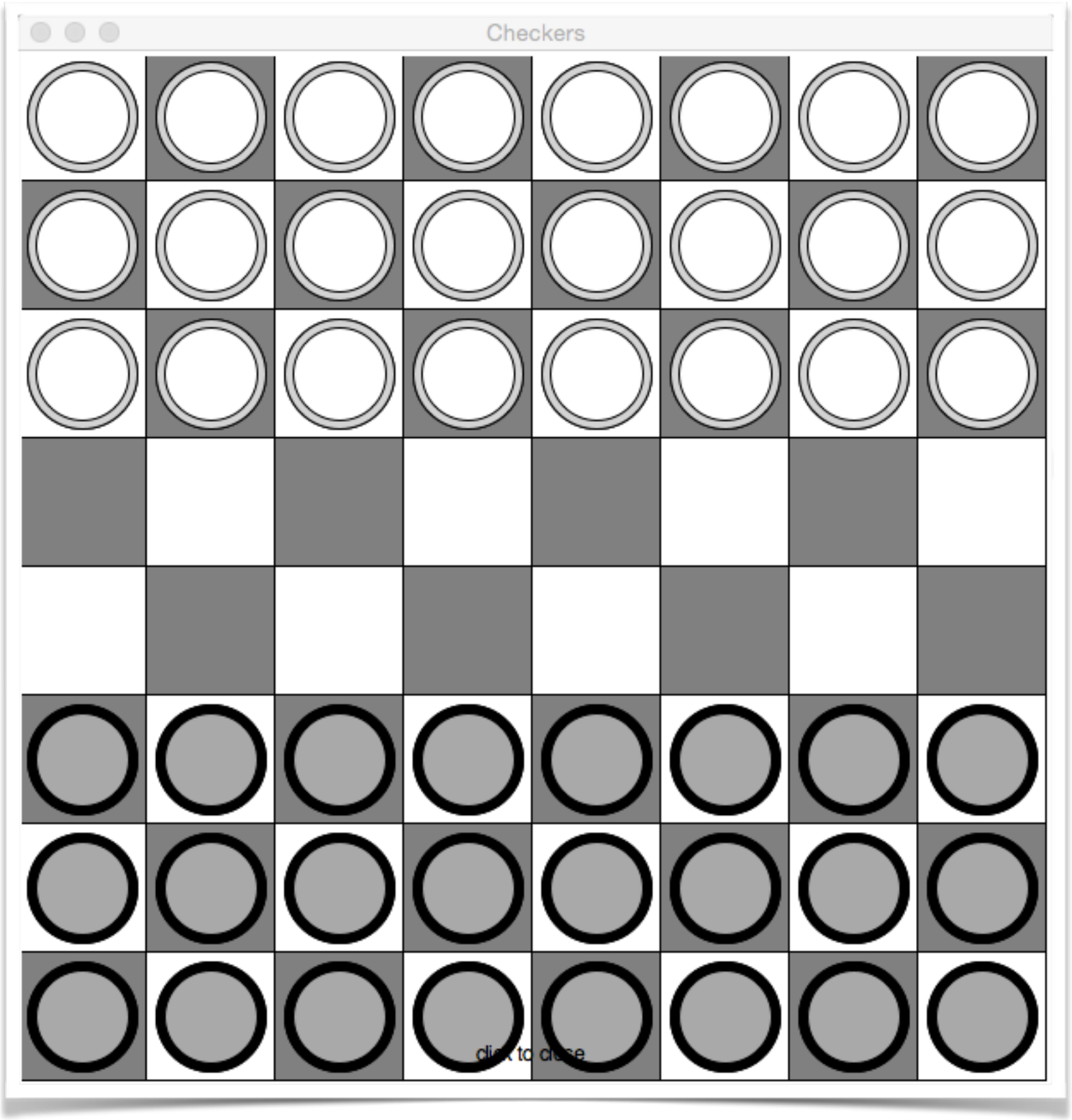
```
def displayBoard( win ):
    """display 8x8 board with alternating white and black cells"""
    for i in range( 0, 8 ):
        for j in range( 0, 8 ):
            x = i * SIDE
            y = j * SIDE
            rect = Rectangle( Point( x, y ), Point( x+SIDE, y+SIDE ) )

            if isWhite( i, j ):
                rect.setFill( "white" )
            else:
                rect.setFill( "lightgrey" )

            rect.draw( win )
            #msg = Text( Point( x+SIDE//2, y+SIDE//2 ),
            #       "{0:1}-{1:1}".format( i, j ) )
            #msg.setSize( 20 )
            #msg.draw(win)
```

Ln: 64 Col: 41

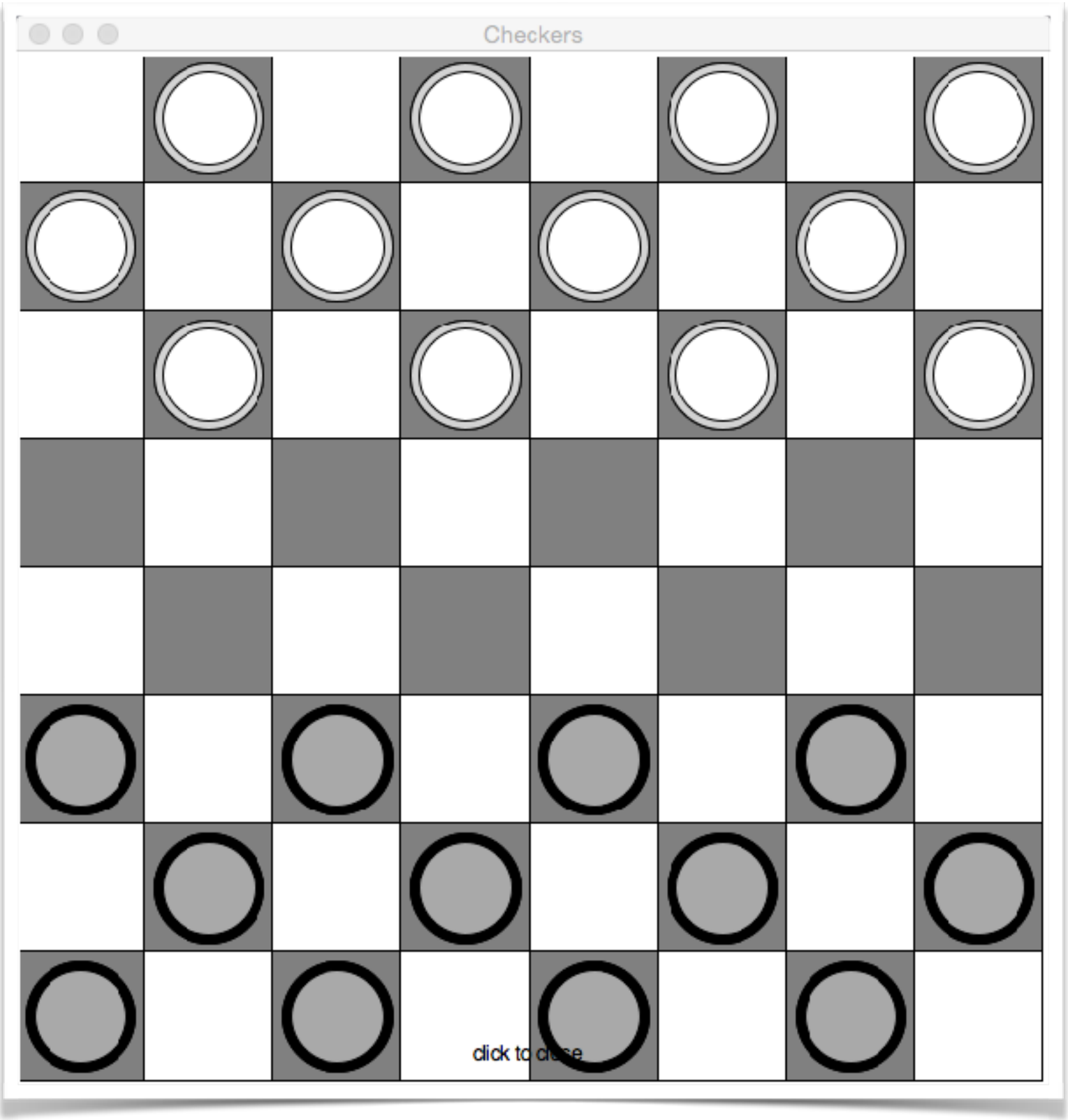
We stopped here last time…

- Mirroring an Image
- Displaying a Checker Board
  - **8x8 Grid**
  - **Alternating Colors**
  - **Creating a Class for a Checkers Piece**
  - **Using a Gif Image for a Piece**

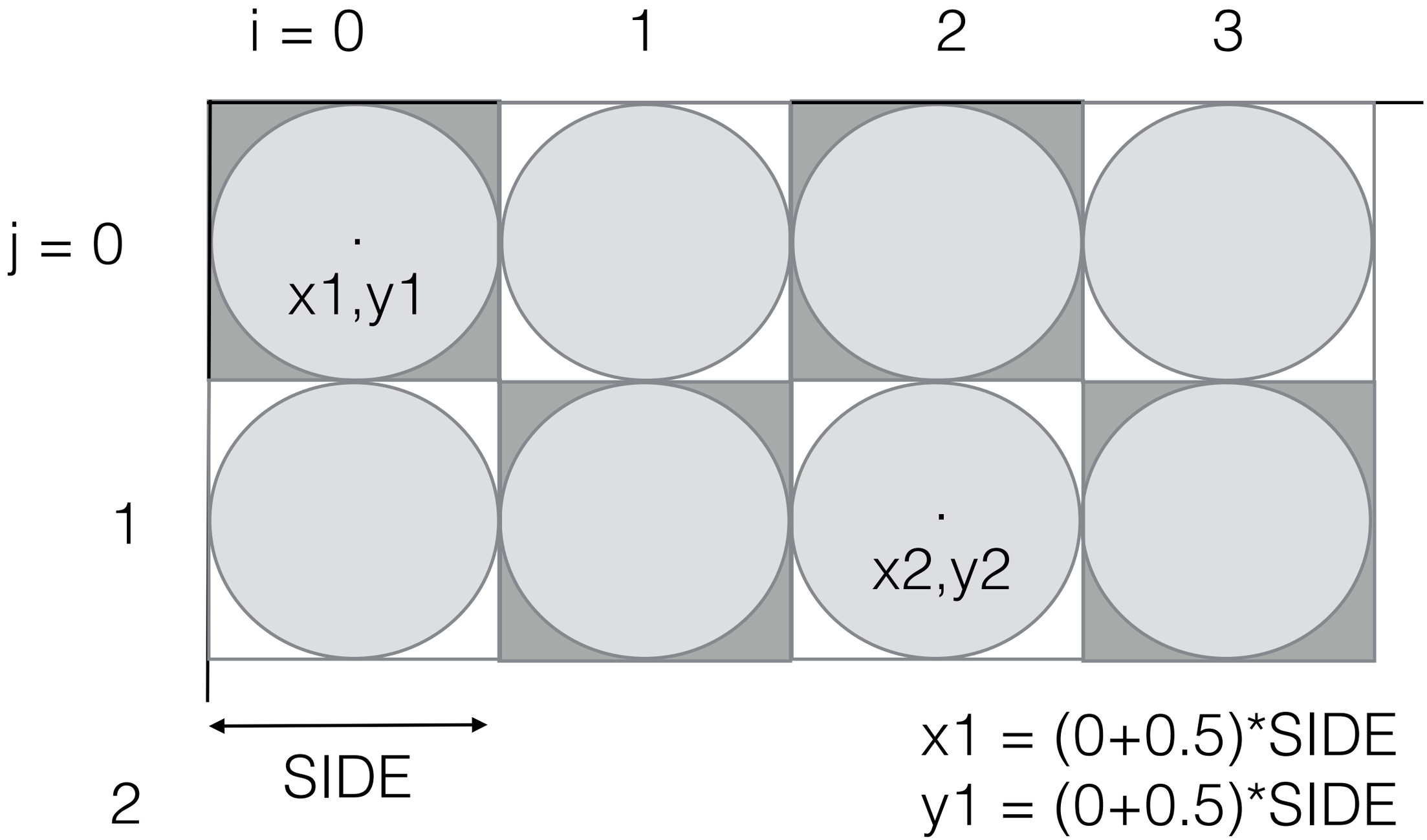
Checkers on Board



Checkers on Board



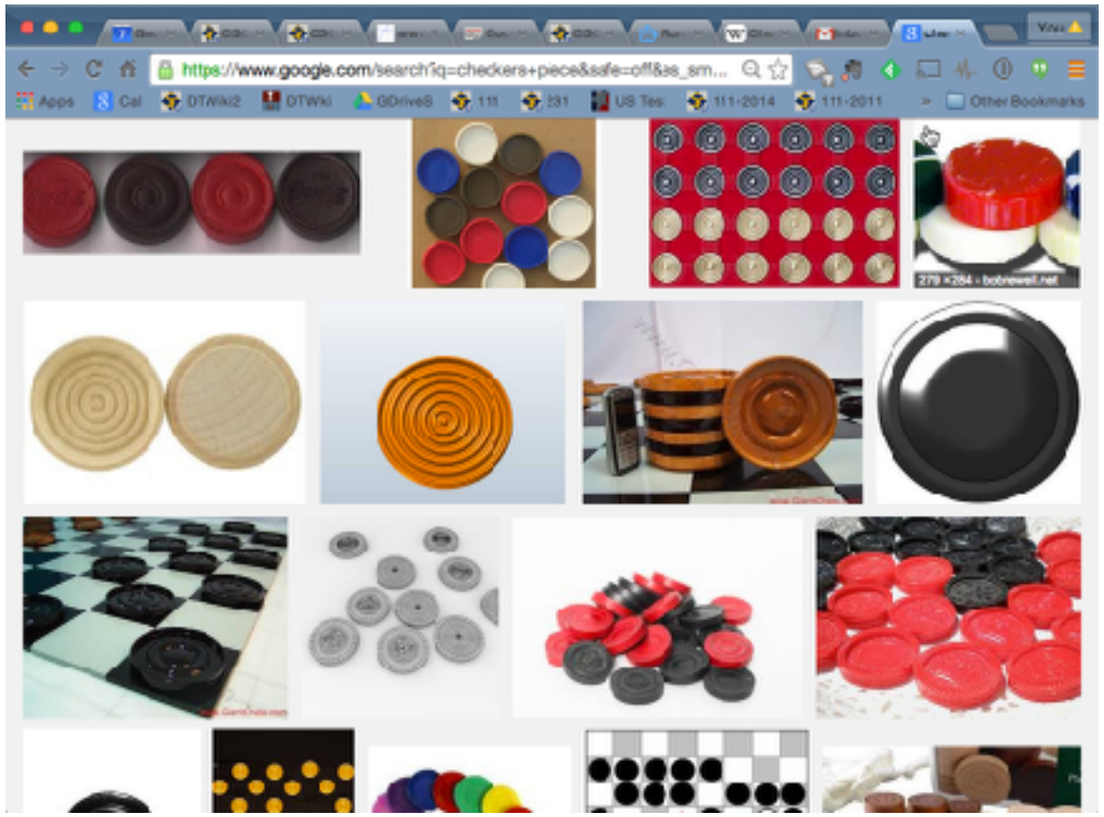
checkers2_incomplete.py



x1 = (0+0.5)*SIDE
y1 = (0+0.5)*SIDE
x2 = (2+0.5)*SIDE
y2 = (1+0.5)*SIDE
xij = (i+0.5)* SIDE
yij = (j+0.5) * SIDE

- Mirroring an Image
- Displaying a Checker Board
  - **8x8 Grid**
  - **Alternating Colors**
  - **Creating a Class for a Checkers Piece**
  - **Using a Gif Image for a Piece**
-

Using an Image

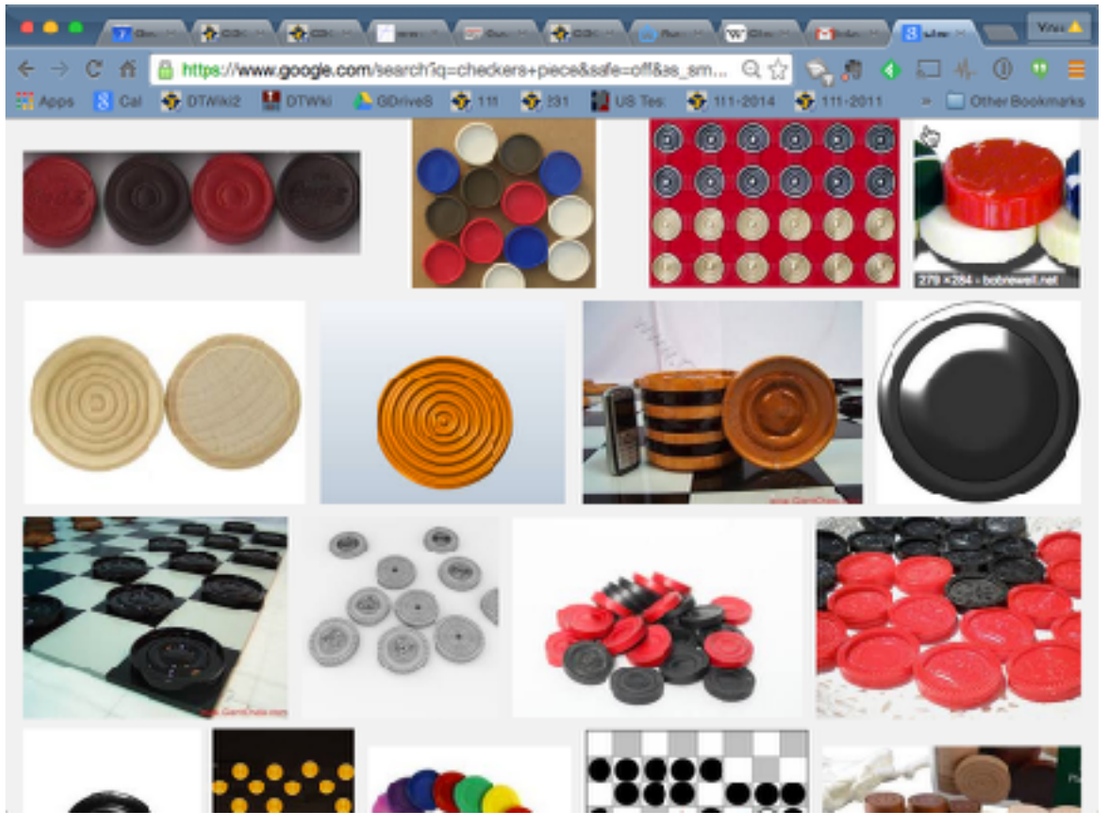


copy

Resize
Convert to gif

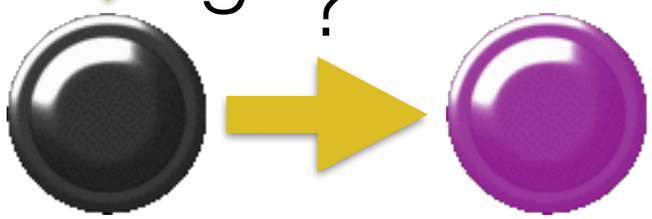
http://pixgood.com/checkers-pieces-clip-art.html

Using an Image



copy

Resize
Convert to gif?

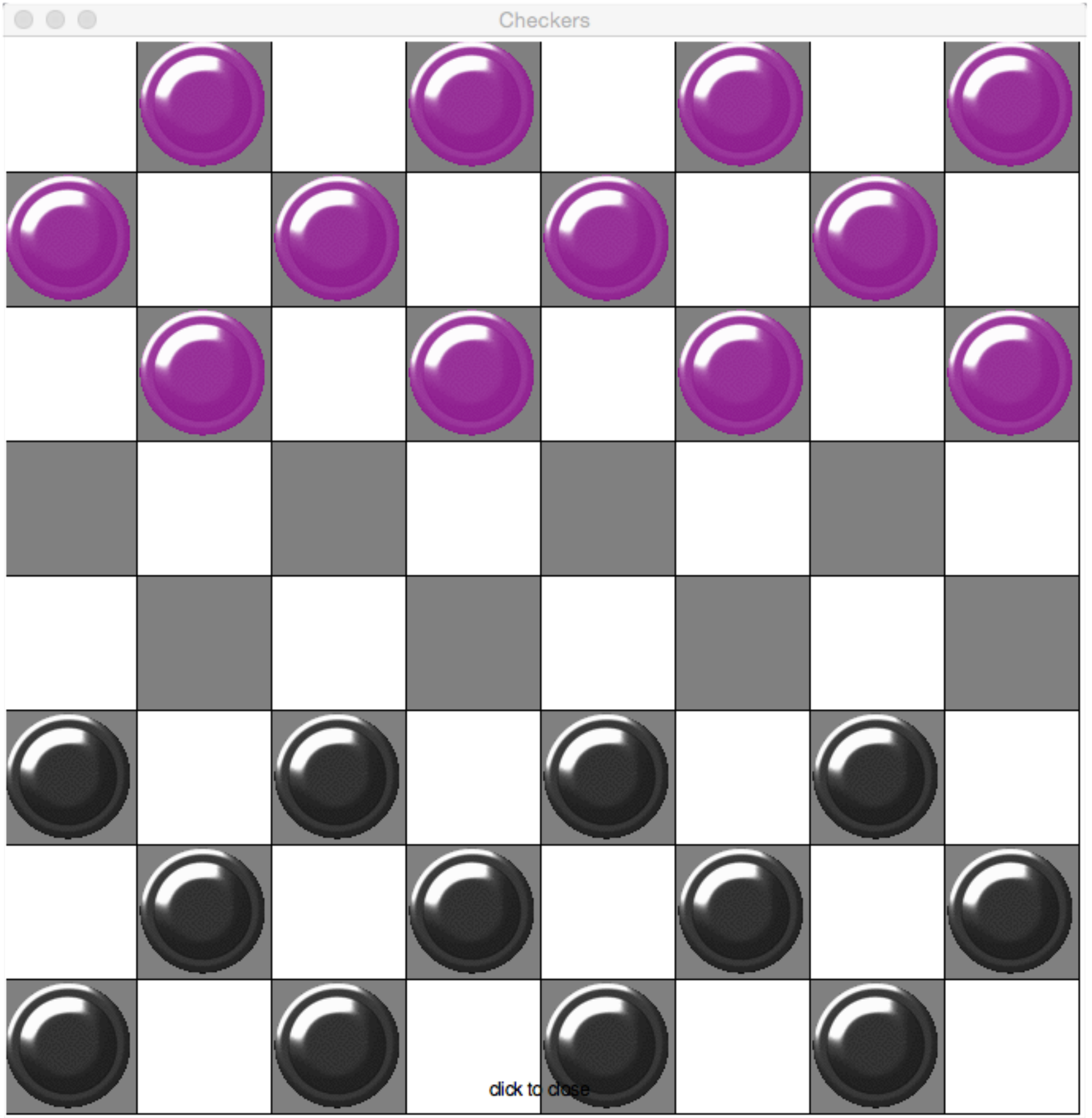


http://pixgood.com/checkers-pieces-clip-art.html

Untitled

```
def main():
    # open the window
    win = GraphWin( "Checkers", WIDTH, HEIGHT )

    # open image of black piece
    piece = Image( Point( WIDTH//2, HEIGHT//2 ),
                   "piece.gif" )
    piece.draw( win )

    # change color of all the dark pixels
    for x in range( WIDTH ):
        for y in range( HEIGHT ):
            r, g, b = piece.getPixel( x, y )
            if r>255-10 and g>255-10 and b>255-10:
                continue
            newColor = color_rgb( r + (255-r)//2, g,
                                  b+(255-b)//2 )
            piece.setPixel( x, y, newColor )

    # save modified image into new file
    piece.save( "piece2.gif" )

    # close window
    win.close()
```

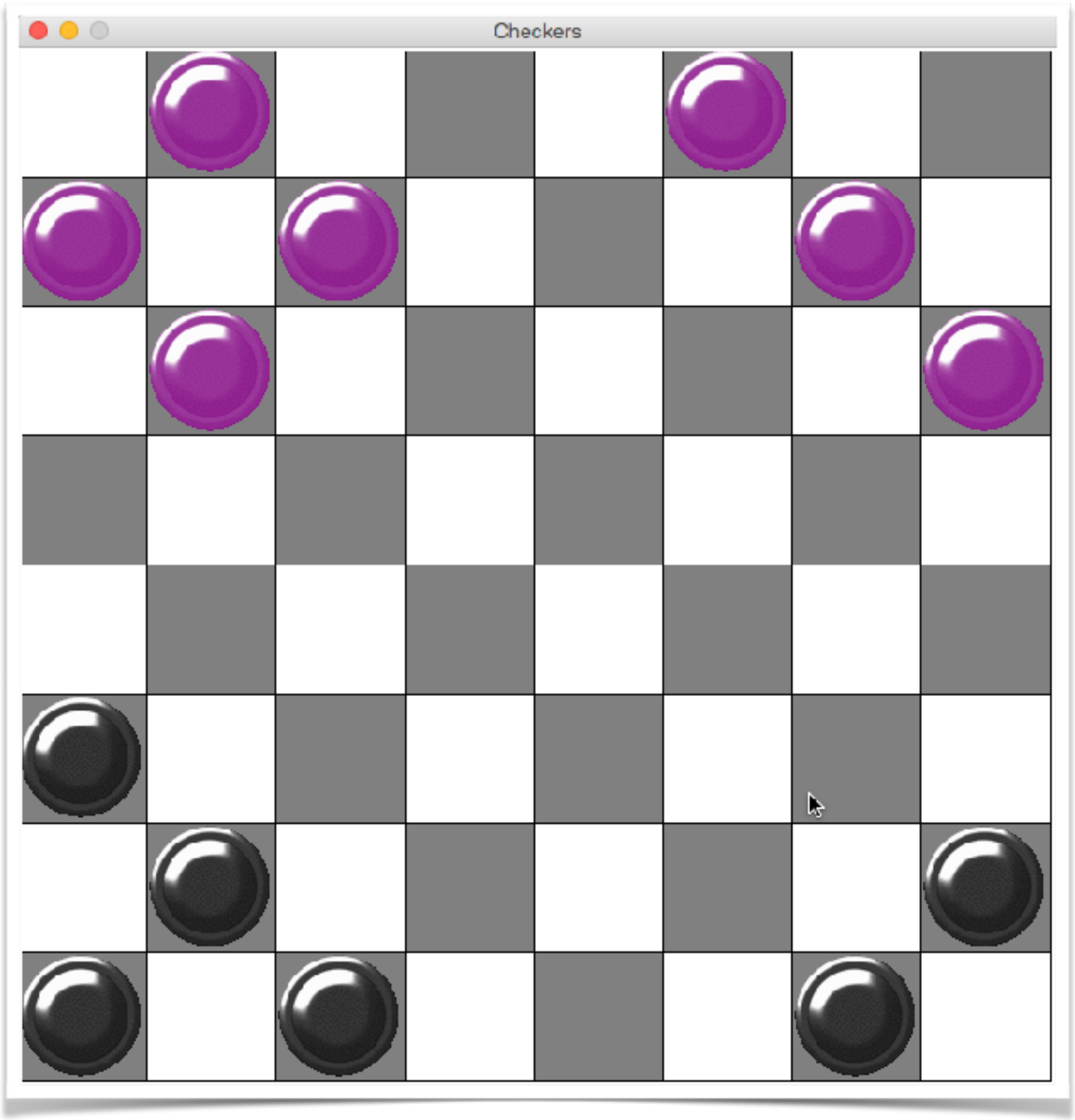
Ln: 16 Col: 12

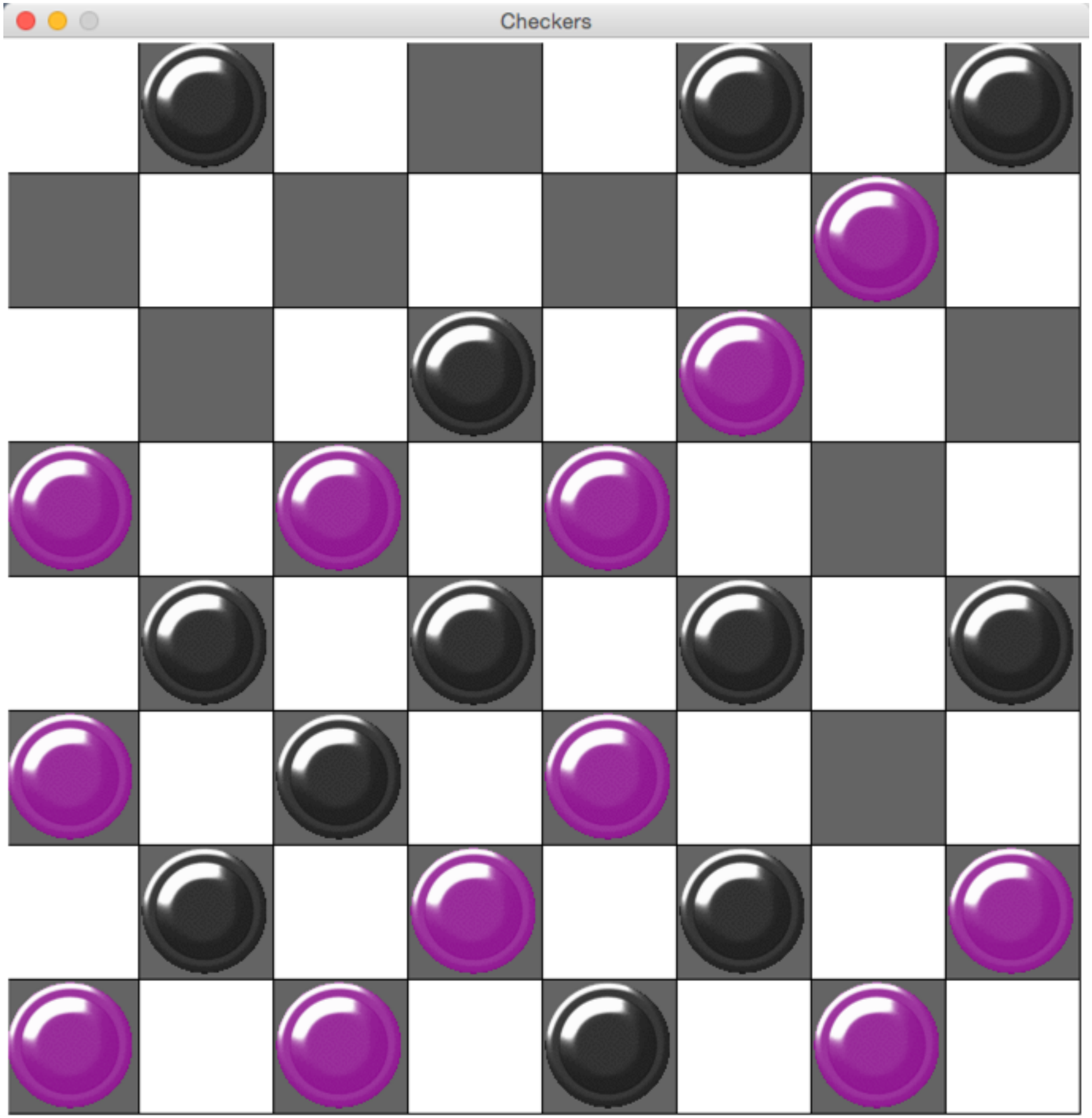


checkers4.py

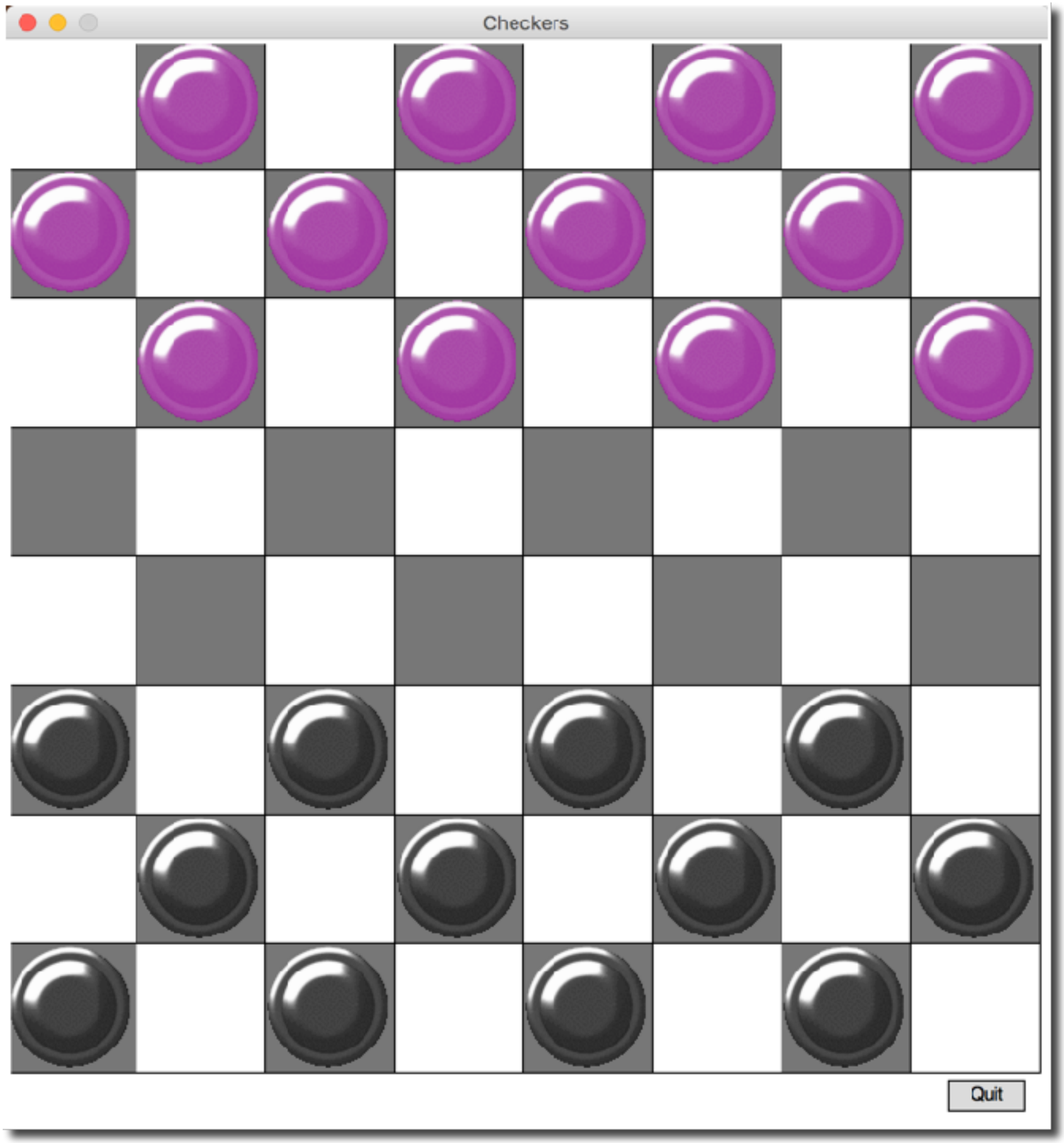
Building an Animation Loop Adding Interactivity

Removing Checkers with the Mouse





Moving Checkers with the Mouse



Adding A QUIT Button

Lists of Lists (Chapter 11 — *Designing with Lists and Classes*)

Two Types of Lists

Useful List Operations

Sorting out cats

Two Approaches to Filtering Data

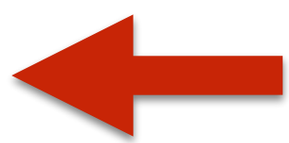
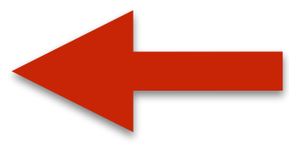
Examples

Two Types of Lists

- someList1 = **[** 1, “hello”, 6.5 **]**

- someList2 = **(** 1, “hello”, 6.5 **)**

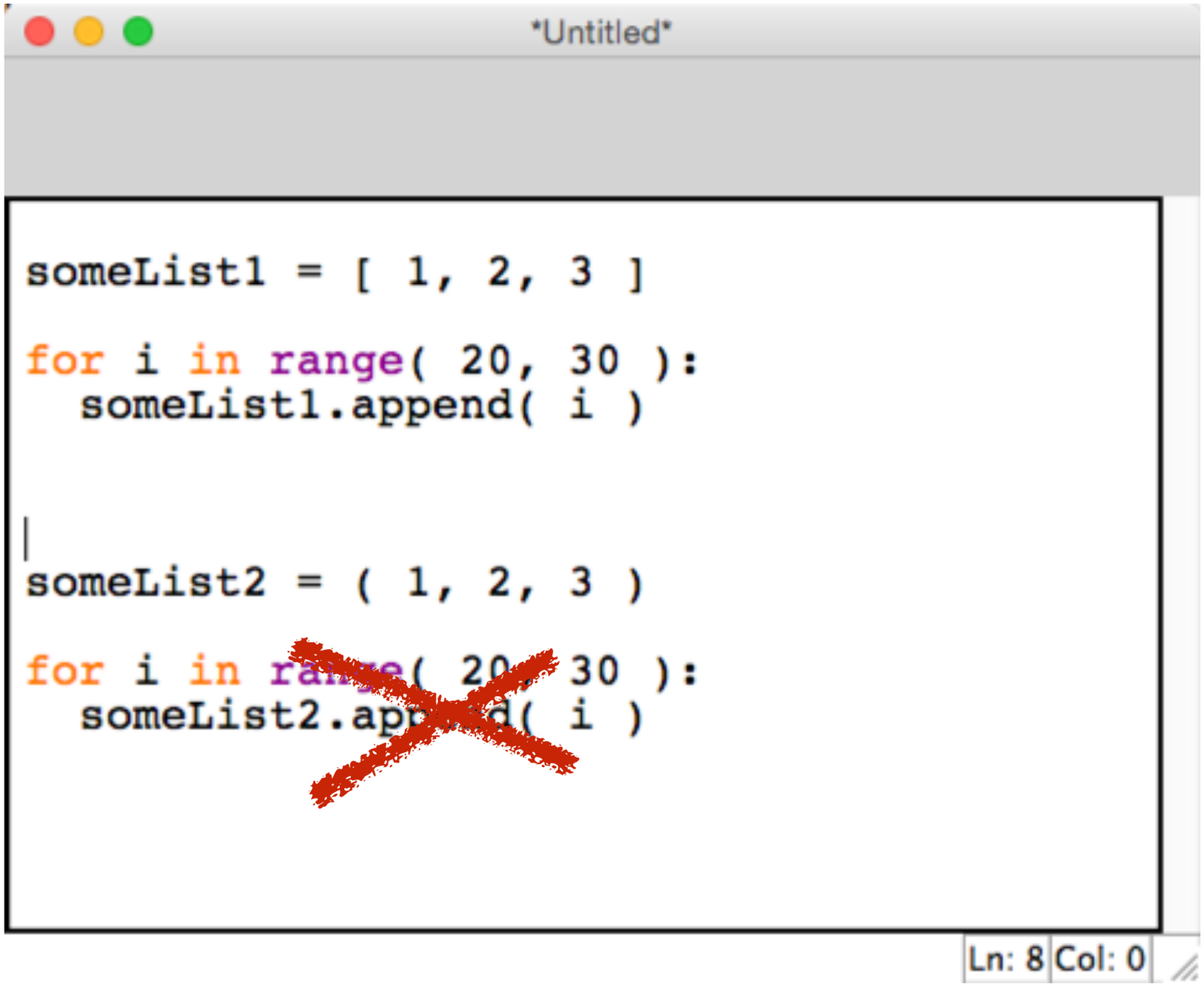
Two Types of Lists

- someList1 = **[** 1, “hello”, 6.5 **]**  **mutable**
- someList2 = **(** 1, “hello”, 6.5 **)**  **immutable**

Untitled

```
someList1 = [ 1, 2, 3 ]

for i in range( 20, 30 ):
  someList1.append( i )


someList2 = ( 1, 2, 3 )

for i in range( 20, 30 ):
  someList2.append( i )
```

Ln: 8 Col: 0


Untitled
someList1 = [1, 2, 3]
for i in range(20, 30):
someList1.append(i)
someList2 = (1, 2, 3)
for i in range(20, 30):
someList2.append(i)
Ln: 8 Col: 0

Notation

Untitled

```
someList1 = [ 1, 2, 3 ]

for i in range( 20, 30 ):
  someList1.append( i )


someList2 = ( 1, 2, 3 )

for i in range( 20, 30 ):
  someList2.append( i )
```

Ln: 8 Col: 0

Tuple

Two Types of Lists

Useful List Operations

Sorting out cats

Two Approaches to Filtering Data

Examples

Useful List Operations

Python Shell

```
Python 3.1.1 (r311:74543, Aug 24 2009, 18:44:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> L = [3, 10, 3, 5, 1, -1, 0, 6]
>>> L
[3, 10, 3, 5, 1, -1, 0, 6]
>>> L.sort()
>>> L
[-1, 0, 1, 3, 3, 5, 6, 10]
>>> L.reverse()
>>> L
[10, 6, 5, 3, 3, 1, 0, -1]
>>> L[0]
10
>>> L[0:3]
[10, 6, 5]
>>> L[-3:]
[1, 0, -1]
>>> S = set( L )
>>> S
{0, 1, 3, 5, 6, 10, -1}
>>> L = list( S )
>>> L
[0, 1, 3, 5, 6, 10, -1]
>>>
```

Ln: 25 Col: 4

Useful List Operations
Sorting Tuples

Python Shell

```
Python 3.1.1 (r311:74543, Aug 24 2009, 18:44:04)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> L = [ (10, "Smith"), (1, "Amherst"), (3, "Umass"), (5, "Hampshire" )]
>>> L
[(10, 'Smith'), (1, 'Amherst'), (3, 'Umass'), (5, 'Hampshire')]
>>> L.sort()
>>> L
[(1, 'Amherst'), (3, 'Umass'), (5, 'Hampshire'), (10, 'Smith')]
>>>
>>>
>>> L2 = [ ("Smith", 10), ("Amherst", 1), ("Umass", 3 ), ("Hampshire", 5 ) ]
>>> L2.sort()
>>> L2
[('Amherst', 1), ('Hampshire', 5), ('Smith', 10), ('Umass', 3)]
>>>
```

Ln: 16 Col: 4

Two Types of Lists

Useful List Operations

Sorting out cats

Two Approaches to Filtering Data

Examples

sortingCats.py - /Users/thicbaut/Desktop/Dropbox/111/sortingCats.py (3.5.4)

```
    def __str__( self ):
        if self.vaccinated == True:
            vacc = "vaccinated"
        else:
            vacc = "not vaccinated"
        return "{0:20}:==> {1:1}, {2:1}, {3:1} yrs old".format(
            self.name, self.breed, vacc, self.age )


def main():
    cats = [ ]
    cats.append( Cat( "Minou", 3, True, "stray" ) )
    cats.append( Cat( "Max", 1, False, "Burmese" ) )
    cats.append( Cat( "Gizmo", 2, True, "Bengal" ) )
    cats.append( Cat( "Garfield", 2, False, "Orange Tabby" ) )

    print( "\nComplete list: " )
    for cat in cats:
        print( cat )

    print( "\nCats sorted by age: " )
    cats.sort()
    for cat in cats:
        print( cat )
main()
```

Ln: 57 Col: 20

Python 3.5.4 Shell

```
Complete list:
Minou                 :==> stray, vaccinated, 3 yrs old
Max                   :==> Burmese, not vaccinated, 1 yrs old
Gizmo                 :==> Bengal, vaccinated, 2 yrs old
Garfield              :==> Orange Tabby, not vaccinated, 2 yrs old

Cats sorted by age:
Traceback (most recent call last):
  File "/Users/thiebaut/Desktop/Dropbox/111/sortingCats.py", line 58, in <modul
e>
    main()
  File "/Users/thiebaut/Desktop/Dropbox/111/sortingCats.py", line 55, in main
    cats.sort()
TypeError: unorderable types: Cat() < Cat()
>>>
```

Ln: 170 Col: 64

Default < > == != Operators

sortingCats.py - /Users/thiebaut/Desktop/Dropbox/111/sortingCats.py (3.5.4)

```
# Cats.py
# D. Thiebaut
# Minou, 3, vac, stray
# Max, 1, not-vac, Burmese
# Gizmo, 2, vac, Bengal
# Garfield, 4, not-vac, Orange Tabby

class Cat:
    def __init__( self, na, ag, vacc, bre ):
        self.name       = na
        self.age        = ag
        self.vaccinated = vacc
        self.breed      = bre
        return

    def __gt__( self, otherCat ):
        return self.age > otherCat.age

    def __lt__( self, otherCat ):
        return self.age < otherCat.age

    def getName( self ):
        return self.name
```

Ln: 16 Col: 0

https://docs.python.org/3/reference/datamodel.html?highlight=__gt__#object.__gt__

Two Types of Lists

Useful List Operations

Sorting out cats

Filtering Data (Everyday Python)

Examples

Two Types of Filtering Problems…

The Problem at Hand

Textual Info.

The Problem at Hand

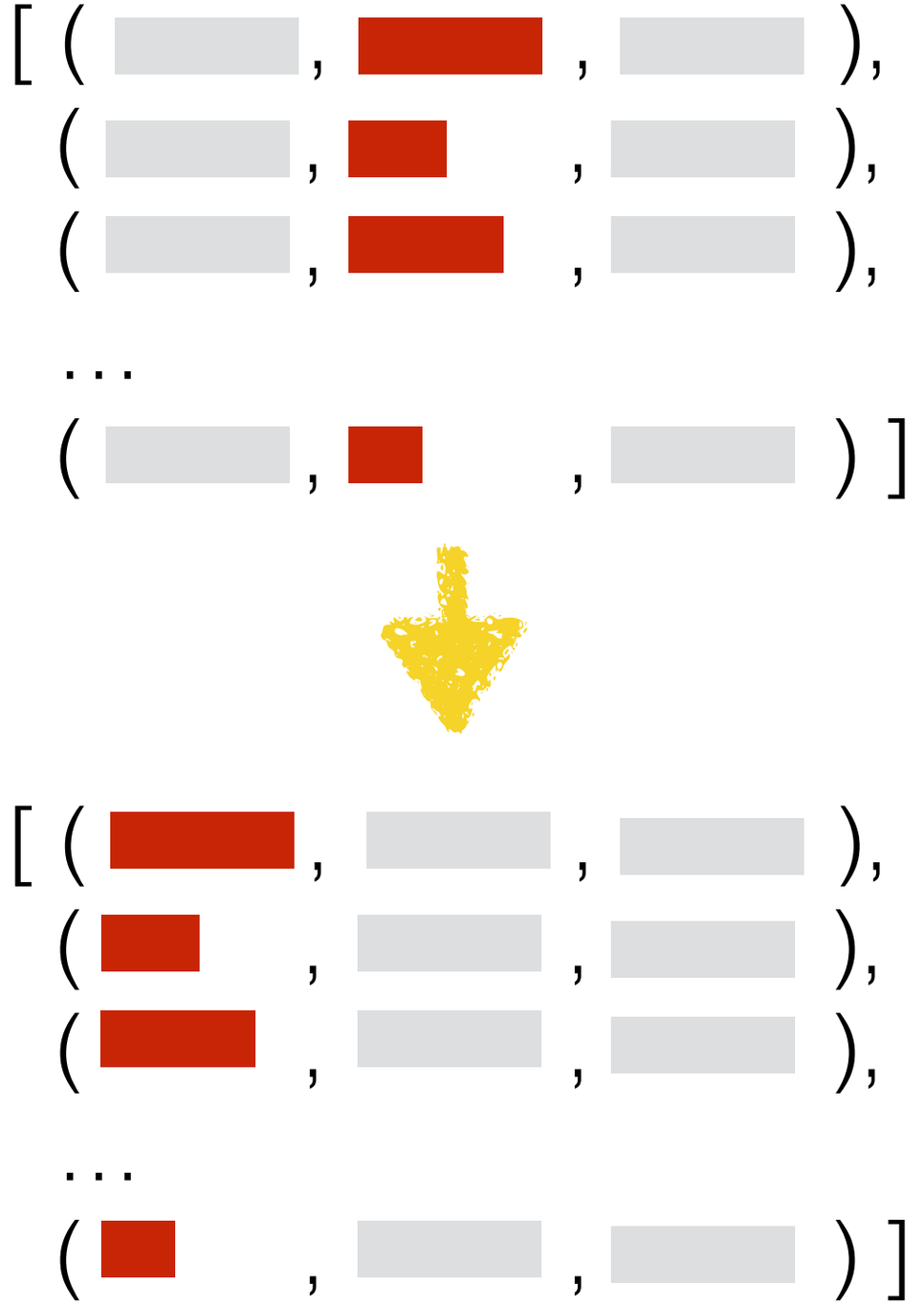
Textual Info.

OPTION 1:

We are only interested in the red information, and only the smaller or larger items…

The Problem at Hand

Textual Info.



The Problem at Hand

Textual Info.

[(, ,),
(, ,),
(, ,),
...
(, ,)]

SORT

[(, ,),
(, ,),
(, ,),
...
(, ,)]

The Problem at Hand

Textual Info.

OPTION 2:

We are only interested in the lines that contain the red information

The Problem at Hand

Textual Info.

[(, ,),
(, ,),
(, ,),
...
(, ,)]

FILTER

[(, ,),
...
(, ,)]

Two Types of Lists

Useful List Operations

Two Approaches to Filtering Data

Examples

Example 1

10 **Rainiest** Months In **Cambridge**, U.K.?

http://cs.smith.edu/~dthiebaut/UKTemperatures/

Example 2

Ammie@hampshire.edu
Bessie@smith.edu
Carylon@smith.edu
Cheryll@smith.edu
Cordelia@smith.edu
Illa@smith.edu
Lisbeth@smith.edu
Mackenzie@smith.edu
Maryellen@smith.edu
Matha@smith.edu
Patrica@hampshire.edu
Sanjuana@smith.edu
Sharie@smith.edu
Sonya@smith.edu
Yuko@smith.edu

Cheryll@smith.edu
Codi@smith.edu
Cordelia@smith.edu
Elenore@smith.edu
Emelia@smith.edu
Josie@smith.edu
…

List of email addresses for students enrolled in several classes.

Need a list of all Smith students without duplicates and a list of all 5-College students without duplicates

Example 3

U.S. Presidents

```
text="""Presidency ,President, Took office ,Left office ,Party , Home State
1, George Washington, 30/04/1789, 4/03/1797, Independent, Virginia
2, John Adams, 4/03/1797, 4/03/1801, Federalist, Massachusetts
3, Thomas Jefferson, 4/03/1801, 4/03/1809, Democratic-Republican, Virginia
4, James Madison, 4/03/1809, 4/03/1817, Democratic-Republican, Virginia
5, James Monroe, 4/03/1817, 4/03/1825, Democratic-Republican, Virginia
6, John Quincy Adams, 4/03/1825, 4/03/1829, Democratic-Republican/National Republican, Massachusetts
7, Andrew Jackson, 4/03/1829, 4/03/1837, Democratic, Tennessee
8, Martin Van Buren, 4/03/1837, 4/03/1841, Democratic, New York
9, William Henry Harrison, 4/03/1841, 4/04/1841, Whig, Ohio
10, John Tyler, 4/04/1841, 4/03/1845, Whig, Virginia
11, James K. Polk, 4/03/1845, 4/03/1849, Democratic, Tennessee
12, Zachary Taylor, 4/03/1849, 9/07/1850, Whig, Louisiana
13, Millard Fillmore, 9/07/1850, 4/03/1853, Whig, New York
14, Franklin Pierce, 4/03/1853, 4/03/1857, Democratic, New Hampshire
15, James Buchanan, 4/03/1857, 4/03/1861, Democratic, Pennsylvania
16, Abraham Lincoln, 4/03/1861, 15/04/1865, Republican/National Union, Illinois
17, Andrew Johnson, 15/04/1865, 4/03/1869, Democratic/National Union, Tennessee
18, Ulysses S. Grant, 4/03/1869, 4/03/1877, Republican, Ohio
19, Rutherford B. Hayes, 4/03/1877, 4/03/1881, Republican, Ohio
20, James A. Garfield, 4/03/1881, 19/09/1881, Republican, Ohio
21, Chester A. Arthur, 19/09/1881, 4/03/1885, Republican, New York
22, Grover Cleveland, 4/03/1885, 4/03/1889, Democratic, New York
23, Benjamin Harrison, 4/03/1889, 4/03/1893, Republican, Indiana
24, Grover Cleveland, 4/03/1893, 4/03/1897, Democratic, New York
25, William McKinley, 4/03/1897, 14/9/1901, Republican, Ohio
26, Theodore Roosevelt, 14/9/1901, 4/3/1909, Republican, New York
27, William Howard Taft, 4/3/1909, 4/03/1913, Republican, Ohio
28, Woodrow Wilson, 4/03/1913, 4/03/1921, Democratic, New Jersey
29, Warren G. Harding, 4/03/1921, 2/8/1923, Republican, Ohio
30, Calvin Coolidge, 2/8/1923, 4/03/1929, Republican, Massachusetts
31, Herbert Hoover, 4/03/1929, 4/03/1933, Republican, Iowa
32, Franklin D. Roosevelt, 4/03/1933, 12/4/1945, Democratic, New York
33, Harry S. Truman, 12/4/1945, 20/01/1953, Democratic, Missouri
34, Dwight D. Eisenhower, 20/01/1953, 20/01/1961, Republican, Texas
35, John F. Kennedy, 20/01/1961, 22/11/1963, Democratic, Massachusetts
36, Lyndon B. Johnson, 22/11/1963, 20/1/1969, Democratic, Texas
37, Richard Nixon, 20/1/1969, 9/8/1974, Republican, California
38, Gerald Ford, 9/8/1974, 20/01/1977, Republican, Michigan
39, Jimmy Carter, 20/01/1977, 20/01/1981, Democratic, Georgia
40, Ronald Reagan, 20/01/1981, 20/01/1989, Republican, California
41, George H. W. Bush, 20/01/1989, 20/01/1993, Republican, Texas
42, Bill Clinton, 20/01/1993, 20/01/2001, Democratic, Arkansas
43, George W. Bush, 20/01/2001, 20/01/2009, Republican, Texas"""
```

Who was president in 1939?