CS 1315 Exam 2, Version A
Fall 2015

ANSWER KEY

Section: _____

TA Name: _____

Question	Points	Score	Grading TA
Section and TA	2		
1	5		
2	10		
3	10		
4	5		
5	5		
6	10		
7	4		
8	10		
Total:	61		

Honor Pledge

"In accordance with the Georgia Tech Honor Code, I have neither given nor received unauthorized aid on this test."

Sign your name on the line above

1. *(5pts)*
Pretend you are the Jython interpreter. Evaluate each of the expressions below. The **result** column is the value of the expression that would be provided by JES. The **type** column is where the data type of the result should be listed. If the expression is not valid Jython syntax, or errors simply write ”Error” in both columns. The first line has been provided as an example.

For purposes of this table, the folder directory you will be accessing via setMediaPath() is 'C:\\Users\Student\\GT\\CS1315\\images'. This directory contains a picture of buzz.jpg which is 100 X 200 pixels.

Expression	Result	Type
`5 + 5`	`10`	`int`
`getWidth(pic)`	`100`	`integer`
`range(3,12,4)`	`[3,6,9]`	`list`
`getMediaPath()`	`'C:\\Users\Student\\GT\\CS1315\\images'`	`string`
`“I love ” + “CS ”*3`	`I love CS CS CS`	`string`
`“abc” > ”cba”`	`False`	`Boolean`

2. *(10 pts)*

You just moved into your new apartment and have a ton of pictures to display so you go to Ikea and buy some photo frames. But when you get home you realize all your photos are too big for the frames! Fill in the blanks below for a function that takes a picture as a parameter, and shrinks it to half of its original width and height.

```
def scalingHalf(pic):
  w = getWidth(pic)
  h = getHeight(pic)
  canvas = makeEmptyPicture(w/2, h/2) #creates the new picture
  for x in range(w/2):
    for y in range(h/2):
      srcPix = getPixelAt(pic, x*2, y*2)  #pixels from the orig. picture
      color = getColor(srcPix)
      tarPix = getPixelAt(canvas, x, y) #pixels from the new picture
      setColor(srcPix, color)
  return canvas                              #returns the new picture
```

3.*(10 pts)*

Write a function that creates a black square 60 x 60 pixels in the center of the picture. This function should take in one parameter, the picture, and show the edited image at the end. When writing this function you may use either nested for loops or conditionals.

Test Case:

```
>>> pic = makePicture(pickAFile())
>>> centerSquare(pic)
```

Original Picture:

Edited Picture:

```
def centerSquare(pic):
  w = getWidth(pic)
  h = getHeight(pic)
  for x in range(w/4, 3*w/4):
    for y in range(h/4, 3*h/4):
      pix = getPixelAt(pic, x, y)
      setColor(pix, black)
  show(pic)
```

```
def centerSquareTwo(pic):
  w = getWidth(pic)
  h = getHeight(pic)
  pixels = getPixels(pic)
  for pix in pixels:
    x = getX(pix)
    y = getY(pix)
    if x > w/4 and x < 3*w/4 and y > h/4 and y
< 3*h/4:
      setColor(pix, black)
  show(pic)
```

4. *(5 pts)*

What will be **printed** when the following function is called?

```
def funForLoops():
  for numOne in range(2,4):
    for numTwo in range(5,7):
      print numOne + numTwo
  print numOne * numTwo

>>> funForLoops()

7
8
8
9
18
```

5. (*5 pts)*

What will be **printed** when the following function is called?

```
def rangeFun(x):
  for num in range(0, x, 2):
    if num < 2:
      print num
    elif num < 4:
      print num + 1
    else:
      print num - 2
    if num > 4:
      print "The final number is " + str(num)

>>> rangeFun(7)
0
3
2
4
The final number is 6
```

6. (*10 pts)*

Fill in the blanks to complete the function that places a small image onto a large background image starting at location (100, 100). If the small image's pixel has a color that is a distance of more than 200 from white, then the small image's pixel should be copied onto the background image.

```
def chroma(background, foreground):
#background in the larger picture, foreground is the smaller picture
  w = getWidth(foreground)
  h = getHeight(foreground)
  for x in range(w):
    for y in range(h):
      forePx = getPixelAt(foreground, x, y)
      foreColor = getColor(forePx)
      if distance(color, white) > 200:
        backPx = getPixelAt(background,  x + 100,  y + 100)
        setColor(backPx  , foreColor)
  show(background)
```

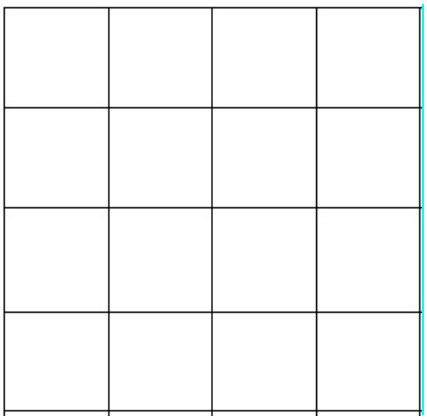
7. (*4 pts)*

The two functions below produce the same image, but they access and edit the pixels differently. Use the empty pictures below to indicate how each function accesses and edits the pixels. On the first picture, labeled EXAMINED PIXELS, place X's on the squares that are accessed by the function. On the second picture, labeled EDITED PIXELS, color the squares that are EDITED by the function.

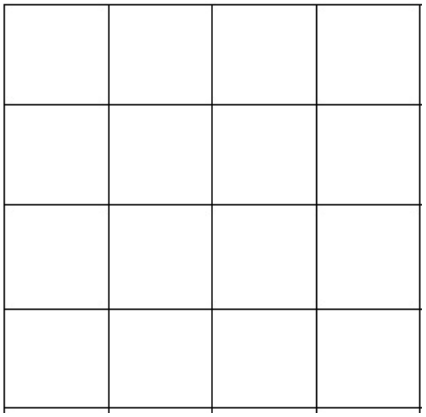
```
def targetOne(pic):
  w = getWidth(pic)
  h = getHeight(pic)
  pixels = getPixels(pic)
  for pix in pixels:
     x = getX(pix)
     y = getY(pix)
     if x > w/2 and y < h/2:
        setColor(pix, black)
```

```
def targetTwo(pic):
  w = getWidth(pic)
  h = getHeight(pic)
  for x in range(w/2, w):
    for y in range(0, h/2):
      pix = getPixelAt(pic, x, y)
      setColor(pix, black)
```

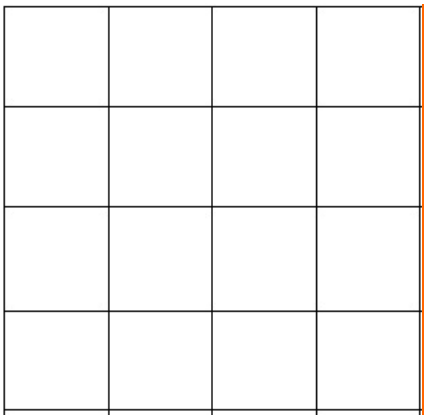
EXAMINED PIXELS



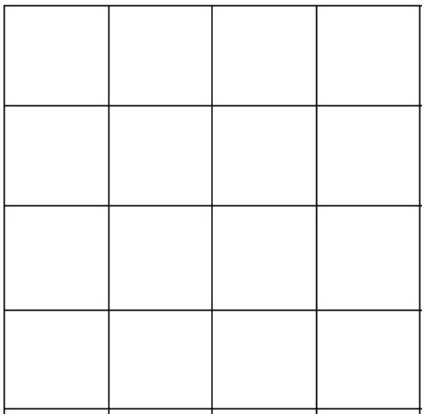
EXAMINED PIXELS



EDITED PIXELS



EDITED PIXELS



8. (*10 pts)*

Write a function that mirrors the left half of the picture onto the right half of a picture. This function should take in one parameter, a picture, and show the edited picture at the end. YOU MUST USE NESTED FOR LOOPS WHEN WRITING THIS FUNCTION.

Test Case:

```
>>> pic = makePicture(pickAFile())
>>> mirrorMe(pic)
```

Original Picture:

Edited Picture:

```
def mirror(pic):
  w = getWidth(pic)
  h = getHeight(pic)
  for x in range(w/2):
    for y in range(h):
      srcpix = getPixelAt(pic, x, y)
      color = getColor(srcpix)
      destpix = getPixelAt(pic, w - x -1, y)
      setColor(destpix, color)
  show(pic)
```