CS116 – Nested loops in a matrix

Purpose: We have been viewing a Picture as a list of pixels but another way to certainly view a Picture is as a two-dimensional matrix of pixels with a width and a height. There are two reason why we would need to view a Picture this way: 1) to process just a region of the picture or 2) to perform an operation that is dependent on the location of the pixel. A region will have an upper left corner position (xoffset,yoffset) and the width and height of the region. Whether dealing with the entire picture or a region, we can use a nested loop where one loop iterates through one dimension or the picture/region and the other loop iterates through the other dimension.

Skills: After completion of this module you should be able to

1. Use a two-dimensional matrix

Knowledge: This module will help you become familiar with the following content knowledge:

1. Nested loops

Schema: Process a matrix using a nested loop

```
for xindex in range(0,width to be processed):
      for yindex in range(0,height to be processed):
           process item at location (xoffset+xindex, yoffset+yindex)
```

or

```
for yindex in range(0,height to be processed):
      for xindex in range(0,width to be processed):
           process item at location (xoffset+xindex, yoffset+yindex)
```

Activity: With your group perform the following tasks and answer the questions. You will be reporting your answers back to the class in 45 minutes.

1. To motivate the concept, consider a one-dimensional case first. The following code draws a black line (one-dimensional) on a Picture at the given y-coordinate and starting at a given x-offset. The width of the line is also provided.

   ```
   def draw(self, xoffset, y, width):
       for xindex in range(0,width):
           pixel = self.getPixel(xoffset+xindex,y)
           pixel.setColor(black)
   ```

   Create a blank 500 by 500 Picture object p. Predict what `p.draw(100,100,300)` will do before showing the altered picture.

   How will the result change with `p.draw(100,100,350)`?

If we draw a bunch of lines with different y values we will get a black rectangle. Alter the method with extra parameters: `draw(self,xoffset,yoffset,width,height)`. Enclose the entire body of the method in another `for` loop which will draw the lines multiple times starting at the `yoffset` and producing a box with the `height`.

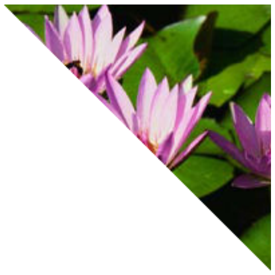
Predict what will happen if you switch the two `for` statements so the `xindex` loop is first and the `yindex` loop is next.

2. Suppose we want to crop a picture so that we return only a rectangular region of the picture. The region can be specified by providing the coordinate of the upper left corner (xoffset,yoffset) and the width and height of the rectangle. We set the colors of the resulting picture to the colors within the region of the given picture.

   Complete the `crop` method:

```
   def crop(self, xoffset, yoffset, width, height):
           result = Picture(width,height)
           for xindex in range(0,width):
                for yindex in range(0,height):
                      pixel1 = self.getPixel(xoffset+xindex,yoffset+yindex)
                     # complete the code to get the correct pixel pixel2 from result
                      pixel2.setColor(pixel1.getColor())
          return result
```

   What minor changes are needed to change the crop so that it gives us a triangular region inside a square of the given height like this?



3. We will look at an example which uses the entire picture rather than a region but requires the locations of the pixels for processing:

   ```
   def flip(self):
       result = Picture(self.getWidth(),self.getHeight())
       for xindex in range(0,self.getWidth()):
           for yindex in range(0,self.getHeight()):
               pixel = self.getPixel(xindex,yindex)
               resultPixel = result.getPixel(self.getWidth()-1-xindex,yindex)
               resultPixel.setColor(pixel.getColor())
       return result
   ```

   What does this method do to a Picture object?

   Why do we need to process the Picture using a matrix rather than just a list of pixels?

   Why are we modifying the pixel at `(self.getWidth()-1-xindex,yindex)` rather than at `(self.getWidth()-xindex,yindex)`?

4. Consider the following Picture method which performs edge detection:

   ```
   def edges(self,threshold):
       result = Picture(self.getWidth(),self.getHeight())
       for x in range(0,self.getWidth()-1):
           for y in range(0,self.getHeight()-1):
               pixel = self.getPixel(x,y)
               pixelRight = self.getPixel(x+1,y)
               pixelDown = self.getPixel(x,y+1)
               pixelColor = pixel.getColor()
               rightColor = pixelRight.getColor()
               downColor = pixelDown.getColor()
               if (pixelColor.colorDistance(rightColor)> threshold or
                   pixelColor.colorDistance(downColor)>threshold):
                   result.getPixel(x,y).setColor(black)
       return result
   ```

   Try this with various threshold values until you get a good result.

   What is the `if` statement doing in this method?

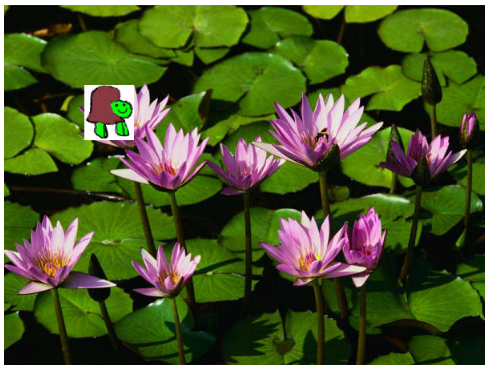
   Why are the two `for` loops going from 0 to the dimension length -1 rather than from 0 to the dimension length?

Complete each of the following assignments to be submitted for grading. Each should be done individually but you can consult with a classmate to discuss your strategies or if you get an error message that you do not understand.
For nested loop problems it will help if you ask yourself the following questions:

1. What is the width and height of the region in the matrix?
2. What offsets if any are needed?
3. What processing is needed for each item in the region?

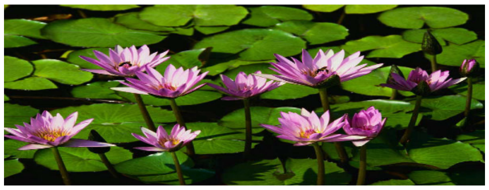
Assignment 1:
Write Picture method `overlay(self,background,xoffset,yoffset)` that overlays the picture on top of a background picture with its upper left corner at (xoffset,yoffset).

Criteria of Success: Here is the turtle on the waterlilies background at offset (100,100)



Assignment 2:
Write a Picture method `squish(self)` which squishes a picture vertically by removing every other row of the picture.

Criteria of Success: Here is the squished waterlilies:

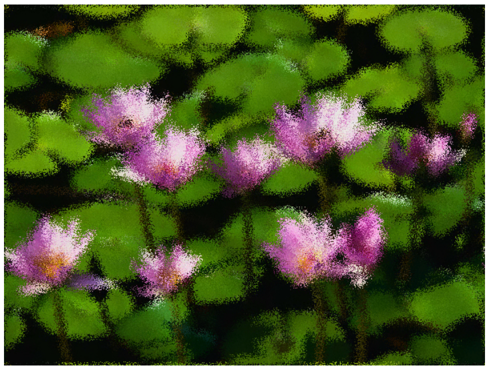


Assignment 3:
Write a Picture method `glass(self,distance)` which implements a glass filter on the picture. The filter works as follows:

1. For every pixel in the picture compute two random numbers, `randomx` and `randomy`, each in the range `-distance` to `distance`.
2. Use these two random numbers to select a pixel that is `randomx` away in the x-dimension and `randomy` away in the y-dimension from the pixel location you are processing.
3. Make sure your new pixel does not fall off the edge of the picture by using the modulo (%) operator in both dimensions.
4. Use this new neighboring pixel color as the color in the result instead of the real color at that position.

You will want to import the `random` module at the top of your file. Here is an example of generating a random number from 0 to 10: `random.randint(0,10)`

Criteria of Success: Here is the glass filter with distance 5 on waterlilies:



Submit your python files with your methods in Canvas for grading.