**Outline**

Develop an understanding of how images and graphics are drawn and stored in a computer. Learn about the RGB colour space. Apply Python concepts related to lists and loops

**Level 1: RGB Color Space**

1. Create a new Repl for Python with Turtle.
   1. Copy and paste “Sample Program #1” from the listing at the end of this module.
   2. Run the program and examine the Turtle output
2. Colours can be specified by using a combination of three numbers. These three numbers together define a “Pixel” point in a graphic image.
   1. What position is the number that controls the amount of red (r) in the pixel?

-The first position controls the red amount of a pixel.

* 1. What position is the number that controls the amount of green (g) in the pixel?

-The second position controls the green amount of a pixel.

* 1. What position is the number that controls the amount of blue (b) in the pixel?

-The third position controls the blue amount of a pixel.

1. Colour number values can range from 0 to 255.
   1. What happens when the colour value is less than 255?

-If the colour value is less than 255, then the color’s brightness is less than its maximum

* 1. What happens when the colour value is close to 0?

-The closer the value to 0, the darker the color gets.

1. Other shades of colours can be created using a combination of r,g,b number values.
   1. Create a pixel containing a shade of the colour orange.

-255,70,0

* 1. Create a pixel containing a shade of the colour yellow.

-255,255,0

* 1. Create a pixel containing a shade of your favorite colour.

-0,255,0

1. Black, white, and shades of grey are created using combinations of equal r,g,b number values.
   1. Create a completely white pixel.

-255,255,255

* 1. Create a completely black pixel.

-0,0,0

* 1. Create a pixel containing a shade of middle grey.

-120,120,120

**Level 2: Resolution**

1. Download the image “Resoultion\_284x177.jpg” from Topic B folder in the class repository.
   1. Open the image in a program like Paint or Photoshop.
   2. What is the size of this image? How many pixels does it contain?

-This image contains 50268 pixels.

* 1. Describe how the image looks (e.g. Can you see the pixels?)

-The pixels of the image are not very visible unless looked at from a very close distance to the screen.

* 1. Zoom in the view to enlarge the image
  2. Describe how the image looks (e.g. Can you see the pixels?)

-The image is zoomed in, and the different pixels become more visible because they are bigger in size due to zooming in.

1. Download the image “Resoultion\_16x16.jpg” from Topic B folder in the class repository.
   1. Open the image in a program like Paint or Photoshop.
   2. What is the size of this image? How many pixels does it contain?

-This image contains 64 pixels.

* 1. Describe how the image looks (e.g. Can you see the pixels?)

-This image is bad quality and hard to understand.

* 1. Zoom in the view to enlarge the image
  2. Describe how the image looks (e.g. Can you see the pixels?)

-Zooming in reduces the quality of the image and the colors become clearly visible.

1. Create a new Repl for Python with Turtle.
   1. Copy and paste “Sample Program #2” from the listing at the end of this module.
   2. Run the program and examine the Turtle output
   3. Compare the program output to the “Resoultion\_16x16.jpg” image in question #2 above.

The output of the program is 8x8 pixels instead of 16x16, which means it has 4 times less pixels and the quality is really bad. This image is unrecognizable.

**Level 3: Draw a custom Image**

import turtle

myPen = turtle.Turtle()

myPen.speed(0)

# These variables track the position of the turtle pen

posX = 0

posY = 0

# These variables define the image information.

# Each pixel in the image has a (r,g,b) value

# The complete image is simply a list of pixels

pixelAddress = 0

pixelMemory = [(119, 173, 85), (126, 179, 89), (130, 179, 90), (109, 153, 64), (115, 160, 69), (124, 169, 76), (96, 148, 50), (119, 177, 75), (122, 187, 83), (117, 186, 80), (102, 171, 65), (106, 173, 68), (103, 161, 61), (108, 158, 61), (92, 135, 43), (129, 168, 77), (130, 173, 93), (121, 163, 81), (139, 176, 96), (138, 173, 91), (128, 162, 78), (54, 90, 3), (129, 169, 80), (101, 148, 56), (98, 152, 56), (142, 199, 102), (138, 195, 98), (128, 183, 89), (104, 153, 61), (126, 166, 78), (115, 149, 63), (130, 159, 75), (152, 177, 111), (62, 85, 17), (172, 191, 125), (138, 155, 87), (131, 146, 77), (71, 88, 17), (137, 159, 86), (54, 82, 5), (138, 173, 93), (152, 189, 109), (136, 175, 94), (155, 189, 112), (61, 89, 14), (109, 131, 59), (135, 150, 81), (63, 74, 6), (74, 76, 26), (108, 108, 58), (69, 67, 16), (83, 77, 27), (157, 151, 99), (67, 64, 9), (70, 71, 14), (70, 76, 16), (130, 140, 79), (41, 55, 0), (150, 164, 103), (71, 82, 22), (87, 92, 36), (81, 80, 26), (70, 64, 14), (114, 104, 55), (139, 119, 82), (116, 94, 57), (134, 110, 74), (166, 140, 105), (89, 63, 26), (138, 114, 76), (124, 105, 63), (109, 94, 51), (78, 67, 22), (88, 79, 36), (71, 62, 19), (120, 108, 68), (105, 90, 51), (143, 120, 86), (78, 52, 19), (119, 88, 57), (115, 78, 52), (109, 72, 46), (141, 101, 75), (141, 101, 75), (125, 85, 59), (141, 101, 75), (89, 54, 26), (91, 57, 29), (84, 55, 25), (130, 103, 73), (119, 92, 63), (84, 54, 26), (119, 85, 60), (127, 89, 66), (124, 83, 63), (177, 132, 113), (176, 130, 107), (126, 80, 57), (133, 85, 63), (125, 77, 55), (178, 130, 108), (139, 93, 70), (128, 83, 60), (180, 138, 114), (179, 138, 116), (117, 79, 56), (164, 126, 105), (183, 142, 122), (132, 89, 70), (147, 101, 85), (130, 81, 66), (155, 106, 91), (129, 79, 56), (129, 79, 56), (180, 132, 109), (179, 131, 108), (151, 103, 80), (152, 106, 82), (118, 73, 50), (129, 84, 61), (150, 108, 84), (96, 54, 32), (151, 109, 87), (154, 109, 90), (124, 79, 60), (120, 73, 55), (157, 110, 92), (151, 102, 87), (156, 109, 79), (114, 69, 40), (135, 90, 61), (155, 111, 82), (125, 84, 54), (142, 101, 73), (131, 92, 63), (98, 58, 32), (129, 89, 63), (141, 99, 74), (143, 101, 77), (144, 102, 78), (118, 76, 52), (118, 76, 52), (105, 63, 39), (121, 79, 55), (133, 90, 55), (154, 114, 78), (86, 48, 12), (127, 91, 55), (126, 92, 57), (98, 65, 30), (93, 60, 25), (116, 81, 49), (120, 85, 53), (124, 88, 56), (134, 97, 68), (101, 64, 35), (181, 144, 115), (183, 148, 118), (107, 72, 42), (151, 118, 87), (119, 81, 42), (144, 109, 69), (130, 97, 56), (163, 134, 94), (175, 147, 107), (114, 86, 47), (180, 152, 113), (139, 109, 73), (88, 55, 20), (167, 133, 98), (175, 141, 106), (94, 60, 25), (144, 111, 76), (145, 115, 77), (151, 123, 84), (127, 102, 62), (157, 119, 80), (111, 76, 36), (124, 92, 53), (144, 116, 76), (127, 104, 63), (163, 140, 99), (119, 93, 56), (134, 106, 69), (137, 104, 69), (152, 118, 83), (141, 107, 70), (122, 88, 51), (97, 65, 27), (131, 103, 63), (120, 97, 55), (75, 54, 11), (123, 83, 47), (98, 62, 26), (142, 110, 72), (119, 91, 52), (140, 114, 77), (148, 124, 86), (163, 137, 102), (132, 102, 68), (115, 80, 48), (124, 88, 56), (118, 79, 46), (120, 84, 48), (126, 94, 56), (117, 89, 49), (175, 152, 110), (166, 145, 102), (126, 81, 50), (151, 110, 80), (129, 93, 61), (116, 83, 52), (108, 79, 47), (119, 92, 62), (124, 95, 65), (140, 106, 78), (128, 88, 62), (95, 51, 24), (190, 146, 117), (100, 56, 27), (128, 92, 58), (163, 131, 93), (127, 102, 62), (142, 121, 78), (152, 102, 79), (128, 82, 58), (90, 48, 23), (177, 141, 115), (128, 94, 67), (96, 64, 39), (128, 94, 69), (97, 59, 36), (170, 125, 102), (187, 139, 117), (126, 76, 53), (148, 101, 75), (123, 79, 50), (124, 90, 55), (161, 131, 95), (131, 106, 66), (156, 104, 83), (135, 84, 63), (176, 131, 108), (149, 108, 86), (145, 109, 85), (113, 78, 56), (160, 124, 102), (126, 83, 64), (157, 110, 92), (151, 98, 80), (133, 81, 60), (133, 83, 60), (154, 108, 82), (142, 103, 72), (115, 82, 47), (89, 61, 24)]

def square(size):

for i in range(4):

myPen.fd(size)

myPen.lt(90)

# This user defined function draws a single image pixel

def drawPixel(rgb) :

global posX

myPen.down()

myPen.color(rgb)

myPen.begin\_fill()

square(18)

#myPen.circle(4.5)

myPen.end\_fill()

myPen.up()

myPen.forward(18)

posX = posX + 18

# This user defined function starts a new row of pixels

def newRow() :

global posX

global posY

myPen.up()

myPen.left(180)

myPen.forward(posX)

myPen.left(90)

myPen.forward(18)

myPen.left(90)

myPen.down()

posX = 0

posY = posY + 18

# THE MAIN PROGRAM CODE STARTS HERE

#

# Draw eight(sixty-four) rows of the image.

# Each row contains eight(sixty-four) pixels

def main(size,resolution):

global pixelAddress

for row in range (resolution) :

for column in range(resolution) :

drawPixel(pixelMemory[pixelAddress])

pixelAddress += 1

newRow()

#Done

main(15,16)

**SAMPLE PROGRAM #1**

import turtle

myPen = turtle.Turtle()

# These variables track the position of the turtle pen

posX = 0

posY = 0

# This user defined function draws a single image pixel

def drawPixel(rgb) :

global posX

myPen.down()

myPen.color(rgb)

myPen.begin\_fill()

myPen.circle(8)

myPen.end\_fill()

myPen.up()

myPen.forward(18)

posX = posX + 18

# THE MAIN PROGRAM CODE STARTS HERE

#

redColor = (255,0,0)

drawPixel(redColor)

drawPixel((128,0,0))

greenColor = (0,255,0)

drawPixel(greenColor)

drawPixel((0,128,0))

blueColor = (0,0,266)

drawPixel(blueColor)

drawPixel((0,0,128))

**SAMPLE PROGRAM #2**

import turtle

myPen = turtle.Turtle()

# These variables track the position of the turtle pen

posX = 0

posY = 0

# These variables define the image information.

# Each pixel in the image has a (r,g,b) value

# The complete image is simply a list of pixels

pixelAddress = 0

pixelMemory = [

(15,15,5),(13,13,6),(8,10,3),(23,21,10),(32,33,16),(33,52,22),(32,54,21),(25,42,17),

(21,19,17),(20,18,9),(7,7,6),(58,65,11),(42,47,7),(11,8,6),(24,25,8),(21,28,10),

(25,19,5),(16,13,8),(28,28,12),(191,192,18),(205,202,21),(42,42,14),(11,11,4),(16,11,3),

(34,59,10),(35,47,15),(24,35,12),(156,139,26),(154,140,22),(28,43,10),(9,12,1),(19,22,5),

(42,88,15),(48,94,18),(98,120,49),(213,195,123),(109,134,66),(44,91,15),(52,86,22),(43,85,18),

(50,95,13),(63,104,39),(224,213,156),(255,225,140),(120,153,92),(41,99,17),(58,103,28),(42,98,17),

(35,86,13),(71,105,42),(223,208,144),(216,204,146),(907,132,79),(28,87,3),(39,83,12),(32,80,12),

(49,102,29),(57,109,33),(92,125,53),(66,103,36),(29,66,13),(32,76,17),(48,91,26),(47,93,23)

]

# This user defined function draws a single image pixel

def drawPixel(rgb) :

global posX

myPen.down()

myPen.color(rgb)

myPen.begin\_fill()

myPen.circle(8)

myPen.end\_fill()

myPen.up()

myPen.forward(18)

posX = posX + 18

# This user defined function starts a new row of pixels

def newRow() :

global posX

global posY

myPen.up()

myPen.left(180)

myPen.forward(posX)

myPen.left(90)

myPen.forward(18)

myPen.left(90)

myPen.down()

posX = 0

posY = posY + 18

# THE MAIN PROGRAM CODE STARTS HERE

#

# Draw eight rows of the image.

# Each row contains eight pixels

for row in range (8) :

for column in range(8) :

drawPixel(pixelMemory[pixelAddress])

pixelAddress += 1

newRow()