Project 5

CS 1323, Fall 2017

# Learning Objectives

1. Declare and construct a two dimensional array. (10 points)
2. Use nested for loops to set the elements of a two dimensional array. (10 points)
3. Write out an image file in PGM format. (10 points)
4. Use constants everywhere they are appropriate in this program. (10 points)
5. Create images with horizontal and vertical stripes. (10 points)
6. Create a synthetic image with diagonal stripe pattern. (20 points)
7. Create a synthetic image with a checkerboard pattern. (20 points)

10 points will be awarded for the documentation of your program. That means using good names for variables, sufficient explanatory comments, proper and consistent indentation of code, and meaningful use of whitespace.

When you submit your program for grading the image sizes should be 10 x 10, the size of the stripes and checkerboard should be 2 pixels wide. These must be constants in your program.

Submit the finished programon Canvas by 11:59 p.m. on Friday, November 3.

# Description

Images can be separated into two categories: natural and synthetic. Natural images are captured from cameras. In this project we will generate synthetic images—images that are written by computer programs. These images will be greyscale (i.e. black, white, and shades of grey). By convention, the value black is stored as 0 (since black is the absence of light). The value white is stored as 255. The shades of grey get lighter as their value increases.

Images are stored in a two dimensional array of integers.

Two dimensional arrays are similar to the one dimensional arrays we’ve used previously, except that they represent a rectangular grid of storage spaces instead of a linear one.

A one dimensional array is shown below, with the indices shown as the contents.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| [0] | [1] | [2] | [3] | [4] | [5] |

A two dimensional array is shown below, also with the indices shown as contents. This array has 3 rows and 5 columns. Row are horizontal and columns are vertical (just like the columns on the front of a building).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| [0,0] | [0,1] | [0,2] | [0,3] | [0,4] |
| [1,0] | [1,1] | [1,2] | [1,3] | [1,4] |
| [2,0] | [2,1] | [2,2] | [2,3] | [2,4] |

To declare a two dimensional array, you use two sets of square brackets.

int[][] image;

To construct a two dimensional array, you need to tell Java how many rows and how many columns should be in the array. The array constructed below has ten rows and three columns.

image = new int[10][3];

To set the values in an image, you use a nested for loop. Since images are rectangular, these for loops will be independent.

The loop below would create an image array that is medium grey.

for (int rows = 0; rows < ROWS; ++rows)

{

for(int cols = 0; cols < COLS; ++cols)

{

image[rows][cols] = 128;

}

}

To complete this project, you need to figure out how to create images that have vertical, horizontal, and diagonal stripes, and a checkerboard pattern. The picture below gives you a hint for how you can create vertical stripes (these are just one row wide—and should be wider to make them more visible).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| (0,0) | (0,1) | (0,2) | (0,3) | (0,4) | (0,5) |
| (1,0) | (1,1) | (1,2) | (1,3) | (1,4) | (1,5) |
| (2,0) | (2,1) | (2,2) | (2,3) | (2,4) | (2,5) |
| (3,0) | (3,1) | (3,2) | (3,3) | (3,4) | (3,5) |
| (4,0) | (4,1) | (4,2) | (4,3) | (4,4) | (4,5) |
| (5,0) | (5, 1) | (5,2) | (5,3) | (5,4) | (5,5) |

The method signature for vertical stripes is below:

public static int[][] createVerticalStripes(int height, int width, int stripeWidth)

To be able to see images on your screen, you need to store them in a file. When images are stored in files, they must have a fixed format. We’re going to use a really simple old format called PGM (portable grey map).

You should figure out the signatures for the other methods on your own.

The format of a PGM image is:

P2<newline>

rows cols<newline>

maximum grey value<newline>

data <write out one row at a time, left to right, then put newline>

For example, a 5x7 image that has one pixel wide vertical stripes looks like this:

P2

5 7

255

0 255 0 255 0

0 255 0 255 0

0 255 0 255 0

0 255 0 255 0

0 255 0 255 0

0 255 0 255 0

0 255 0 255 0

Write a method to write out files in this format.

PGM is not used by many modern image programs, so to see your files as images, you’ll need to use specific software. For Windows PC, this software is called Irfanview. Irfanview is available from <http://www.irfanview.com/>. Install the software on your PC (it’s in Download). In eclipse, when you can see the file name, right click on the file and Open With> Irfanview PGM file. Irfanview does not currently work on Macintosh systems without a lot of fuss. To see the images on a Mac install ToyViewer (<https://itunes.apple.com/us/app/toyviewer/id414298354?mt=12>). Images viewed in ToyViewer can be blurry—but they should still let you see whether you’re doing the right things. If you have trouble viewing the images on your computer, you can read the files and see that the contents are correct.

I suggest using small image sizes initially (like 10 x 10).

# Hints

When you write out files in eclipse, eclipse does not automatically see that they are there. To get eclipse to recognize the file, you’ll need to go to the package explorer, go to the top level folder for your project, and right click and choose refresh.

The vertical and horizontal stripes are fairly simple to write. The checkerboard is easier to write if you think of it as vertical stripes BLACK first then WHITE for NUM\_STRIPES rows, then vertical stripes with the WHITE and BLACK reversed for NUM\_STRIPES rows, etc.

Writing an image with diagonal lines requires an observation. Take a look at the relationship between rows and columns on the diagonal in the array below. The indices always sum to 5! If you look at the diagonal rows above and below the main diagonal, you’ll see a pattern that will make it possible for you to make the diagonal stripes.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| (0,0) | (0,1) | (0,2) | (0,3) | (0,4) | (0,5) |
| (1,0) | (1,1) | (1,2) | (1,3) | (1,4) | (1,5) |
| (2,0) | (2,1) | (2,2) | (2,3) | (2,4) | (2,5) |
| (3,0) | (3,1) | (3,2) | (3,3) | (3,4) | (3,5) |
| (4,0) | (4,1) | (4,2) | (4,3) | (4,4) | (4,5) |
| (5,0) | (5, 1) | (5,2) | (5,3) | (5,4) | (5,5) |