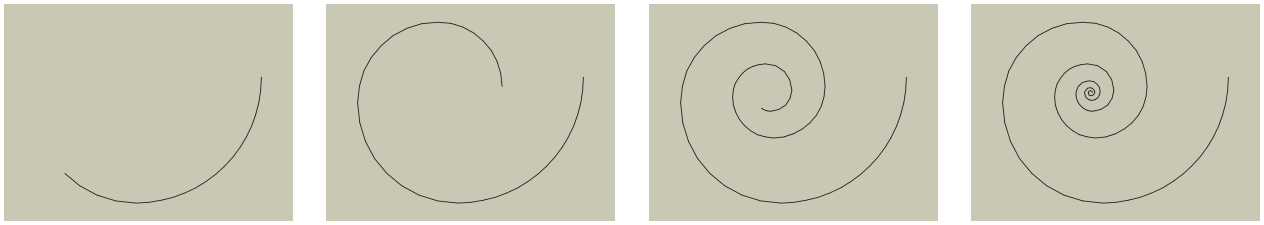
CS 116x Winter 2015
Craig S. Kaplan

Assignment 1: Input/Output

Due date: Tuesday, 20 January, 12:00pm

In this assignment you will develop sketches that receive input from files and produce output to new files.

Question 1: Connect-the-dots



Create a Processing sketch that reads a list of line segments from a text file and draws them one at time until a complete picture is revealed. The sketch should be able to present a sequence of files one after the other in a "connect-the-dots slide show". Here are the specific requirements for this sketch:

- The sketch should run in a window of size 800×600.
- You should assume that the sketch folder contains a sequence of text files with names `drawing_0.txt`, `drawing_1.txt`, ... `drawing_`n.txt (where n is the highest numbered drawing).
- Each text file consists of a sequence of lines of four floats (floating point numbers). The floats give the x and y coordinates of the start and end of a line segment to be drawn.
- Draw one segment per frame until the drawing is complete, and then stop. In the interests of efficiency, don't clear the window and redraw earlier segments every frame—just add the one new line segment to the current window.
- When the user presses any key, clear the screen and proceed to drawing the next file in the list. It should be possible to press a key at any time, interrupting a partially completed drawing if necessary.
- When there are no more files to be read, the program can either stop (and keep on displaying the final drawing) or cycle back to the first drawing.

Here are a few hints:

- I recommend using the function `loadStrings()` to read the contents of one of these files. (Note that this function returns `null` if the file doesn't exist. That will be useful here.) Other functions

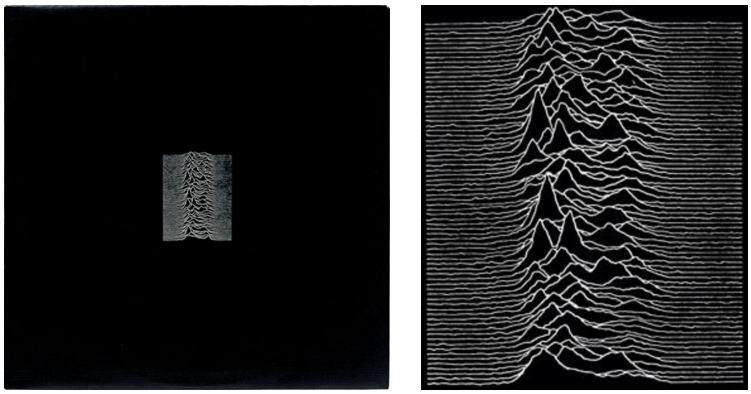
will help you turn each line of input into four floats. See the functions listed under "Conversion" and "String Functions" on the Processing reference page.

- You should assume that the input is perfectly formed. The files are named correctly, and every line of every file contains precisely four floats and nothing else.
- You may not assume that there is any limit on how many files you may have to draw. That means that you can't have a predefined array of file names to check for. You need to keep an integer counter and create strings representing new file names as needed.

On the course web page you will find a ZIP file containing four drawings that you can place in your sketch folder as tests. **You must add at least one drawing of your own**, in its own text file. Your drawing can be very simple if you want, but it can't be empty. We may award bonus marks for especially novel or creative drawings. One way to generate interesting arrangements of lines might be to create a second Processing sketch that writes a sequence of coordinates to a file!

What to submit: On LEARN, you should submit a single sketch entitled `A01_1`. Submit the entire sketch folder, including all drawing files (even the ones that we provided).

Question 2: Unknown Pleasures



Left: the original album cover. Right: a (not quite identical) close-up.

The cover of the 1979 album *Unknown Pleasures* by UK band Joy Division is generally considered to be one of the most iconic rock album covers of all time. In this question you will write a Processing sketch that can generate similar drawings. The sketch must read a description of the curves to draw from an *image*, and must be able to output a vector illustration as a PDF file, using Processing's PDF library.

The requirements are as follows:

- When the sketch starts, it looks for a file in the sketch folder called `input.jpg` (you can assume the file is present). The rows of pixels in that image will become the "graphs" that make up the white lines of the album cover. You will use the brightness of each pixel in a row to control the height of the row's graph at every point.

- Every graph in the final drawing has a "baseline", its vertical position in the sketch window when not affected by the input image. The layout of the drawing as a whole is controlled by two numbers (best stored in global variables): the maximum displacement of each graph from its baseline (i.e., how high above the baseline a maximally bright pixel will be), and the spacing between adjacent baselines.
- You should set the size of the sketch to be large enough to fit the final drawing, based on the size of the input image and the two parameters mentioned above. (That means you'll probably want to wait until after loading the image to make a call to `size()`.)
- When the user presses any key, you should create a vector illustration matching the on-screen drawing, and save it to a file called `output.pdf`. Be sure to consult the documentation on the Processing PDF library at `https://processing.org/reference/libraries/pdf/index.html`.

It may take some thought to figure out how to draw this image, especially in a way that "hides" parts of graphs behind other graphs. You should definitely consult the set of functions under "Vertex" in the Processing reference for ideas on how to draw each graph.

On the course web page you will find a test image that can be used as input. (Note that the image is wide and short—the spacing between lines in the drawing makes it much taller than the input image.) Feel free to experiment with other images. In particular, it may help to prepare a few test images that are much less random, so that you can convince yourself that your sketch is drawing things correctly.

There are many opportunities for enhancements with this sketch, aside from the obvious possibility of experimenting with different input images. It might be interesting to vary the spacing and maximum height interactively, in order to find a best composition for a given input image. You could vary the colours of the graphs (possibly including transparency). You are welcome to include any such enhancements in your sketch, *as long as it supports at least the functionality above.* If you do add enhancements, please include a comment at the top of your sketch that says what they are and how they work. We may award bonus points to enhancements that are especially creative or interesting.

What to submit: On LEARN, you should submit a single sketch entitled `A01_2`. Submit the entire sketch folder. Your sketch folder must contain at least one `output.pdf`, corresponding to the test image file that we provided. If you generate other interesting output PDFs, rename them and leave them in the folder.