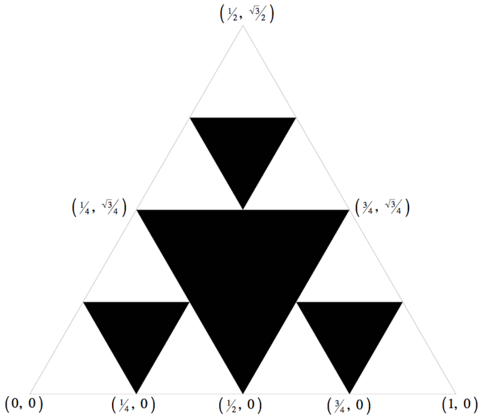
Programming Assignment Checklist: Recursive Graphics

Frequently Asked Questions

I forget how to do geometry. Any hints? Here are the coordinates of the critical endpoints. Click the image for a bigger version.



How do I draw a filled triangle? Use `StdDraw.filledPolygon()`.

How should I go about doing the artistic part of the assignment? This part is meant to be fun, but here are some guidelines in case you're not so artistic. A very good approach is to first choose a self-referential pattern as a target output. Check out the graphics exercises in Section 2.3. Here are some of our favorite student submissions from Spring '08. See also the Famous Fractals in Fractals Unleashed for some ideas. Here are some more ideas. Some pictures are harder to generate than others (and some require trig); consult a preceptor for advice if you're unsure.

What will cause me to lose points on the artistic part? We will deduct points if your picture is too similar to `HTree`, `Sierpinski` or `Brownian`. To be "different enough" from those algorithms, you need to change the recursive part of the program. (E.g., it is *not* sufficient to simply change the triangles to squares in Sierpinski.) We will also deduct points if your artwork can be more easily created without recursion. This is indicated by a tail recursive function, i.e., one that contains a single recursive call at the very end of the function.

Can I use GIF or JPEG files in my artistic creation? Yes. If so, be sure to submit them along with your other files. Make it clear in your `readme.txt` what part of the design is yours and what part is borrowed from the image file.

My function for `Art.java` takes several parameters, but the assignment says that I can only read in one command-line argument N. What should I do? Choose a few of the best parameter values and do something like the following:

```
if      (N == 1) { x = 0.55; y = 0.75; n = 3; }
else if (N == 2) { x = 0.55; y = 0.75; n = 5; }
else if (N == 3) { x = 0.32; y = 0.71; n = 8; }
```

Can I use the mouse to control input values? Check with your preceptor.

Testing and Submission

readme.txt. Use the following readme file template and answer all questions.

Submission. If you don't follow these instructions, you risk annoying your grader and losing a significant number of points.

- Your `Art.java` program *must* take one integer command-line argument `N` (expect it to be between 0 and 7).
- Don't call `StdDraw.save()` in either program.
- Don't call `StdDraw.setCanvasSize()` in either program.

Possible Progress Steps

These are purely suggestions for how you might make progress. You do not have to follow these steps. Note that your final `Sierpinski.java` program should not be very long (ours is around 15 lines, not including comments and blank lines).

- Review the H-tree from the booksite. You can also find it in the lecture and the text.

- Download StdDraw.java to your working directory.

- Write a (nonrecursive) function `triangle()` that takes three real-valued inputs (`x`, `y`, and `size`), and draws an equilateral triangle (pointed downward) with bottom vertex at (`x`, `y`) and side length `size`. To debug and test your function, write `main()` so that it calls `triangle()` a few times, with different parameters. You will be able to use this function without modification in `Sierpinski.java`.

- Write a recursive function `sierpinski()` that takes 4 arguments (`n`, `x`, `y`, and `size`) and plots a Sierpinski triangle of order `n`, whose largest black triangle has side length `size` and bottom vertex (`x`, `y`).

  - Write a recursive function `sierpinski()` that takes one argument `n`, prints the value `n`, and then calls itself three times with the value `n-1`. The recursion should stop when `n` becomes 0. To test this function out, write `main()` so that it reads one integer `N` from the command line and calls `sierpinski(N)`. Excepting the spacing, you should get the following output when you call `sierpinski()` with `N` ranging from 0 to 5. Make sure you understand how this function works, and why it prints the numbers in the order it does.

  - Modify `sierpinski()` so that instead of printing `n`, it prints the size of the triangle to be plotted. Your function should now take two arguments: `n` and `size`. The initial call from `main()` should be to `sierpinski(N, 0.5)` since the largest black triangle has side length 0.5. Each successive level of recursion halves the length. Your function should produce the following output.

  - Modify `sierpinski()` so that it takes 4 arguments (`n`, `x`, `y`, and `size`) and plots a Sierpinski triangle of order `n`, whose largest black triangle has side length `size` and bottom vertex (`x`, `y`). Start by drawing Sierpinski triangles with pencil and paper. Use the picture in the Q+A above to figure out the geometry of where the smaller Sierpinski triangles should go.

Enrichment

- Here's a fractal in polymer clay or a fractal cookie.