Programming Assignment Checklist: Recursive Graphics

Frequently Asked Questions

What do I need to do to get full credit for 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 choose a self-referential pattern as a target output. Check out the graphics exercises in Section 2.7 for still more possibilities. See also the Famous Fractals in Fractals Unleashed for some ideas. Here are some more ideas. Here's a Pythagorean tree. Then, try to figure out a recursive scheme that will generate it. Some pictures are harder to generate than others; consult a preceptor for advice if you're unsure.

What will cause me to lose points on the artistic part? We'll deduct points if your picture is overly boring or too similar to `HTree.java`. You will also lose points if your artwork could easily be created without recursion, e.g., it is *tail-recursive*: 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.

What size window should I use? Please use 512-by-512 for both `HTree.java` and `Art.java`.

What size should I make the H's? The big H should be comprised of three line segments of length 256 each, and centered on (256, 256).

I get the error `java.lang.NullPointerException`. What am I doing wrong? Perhaps you left out `StdDraw.create(512, 512)` at the very beginning.

I get a blank gray screen. What am I doing wrong? Perhaps you left out `StdDraw.show` or `StdDraw.pause`.

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 upsetting your grader and losing a significant number of points.

- Your `Art.java` program *must* take exactly one integer command line input (expect it to be between 1 and 7).
- Your `Art.java` *must* fit within a 512-by-512 box.
- Don't call `StdDraw.save` 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 `HTree.java` program should not be very long (ours is 21 lines, not including commments and blank lines).

- Review the Koch snowflake from lecture.
- Download Draw.java and StdDraw.java to your working directory.
- Write a (nonrecursive) function `void drawH(double x, double y, double size)` that plots an H, centered on (x, y) with each of the 3 line segments having length `size`. To debug and test your function,

write `main` so that it calls `drawH` a few times, with different parameters. You will want to use this function without modification in `HTree.java`.

- Write a recursive function `htree` that plots an H-tree pattern of order n. Consider creating this function in several steps.
    - Write a recursive function `void htree(int n)` that prints the value `n`, and then calls itself four 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 `htree(N)`. Excepting the line wraps, you should get the following output when you call `htree` with N = 0, 1, 2, 3, 4, 5. Make sure you understand how this function works, and why it prints the numbers in the order it does.
    - Modify `htree` so that instead of printing `n`, it prints the size of the H to be plotted. Your function should now have the signature `void htree(int n, double size)`, and the initial call from `main` should be to `htree(N, 256.0)` since the big H has side length 256.0. Each successive level of recursion halves the length. Your function should produce the following output.
    - Modify `htree` so that it plots a recursive H pattern of order N, centered on (x, y) of the given size. Your function should now have the signature `void htree(double x, double y, int n, double size)`. Start by drawing H's with pencil and paper. Do some calculations to figure out the geometry of where the smaller H's should go.
- Now that you have the H-tree pattern drawn, it is easy to animate. Append the command `StdDraw.pause(10)` to the end of the `drawH` function.