Programming Languages
Dr. Gurka, Spring 2019 Assignment Cover Letter

Name: Eryn Kelsey-Adkins
Assignment: PostScript
Date submitted: 4/22/19
Total time on project: 30 hours
Programming language(s): Postscript

On time or GOOJF? On time
Is anything missing or not working correctly? Be specific. No.

Did you collaborate with any classmates on this project? No.
If yes, who was it, and what did you work together on?

Did you get any tutoring or similar help on this project, or otherwise work with someone outside class? If yes, explain. No.

What resources and references, if any, did you use? Be specific.

PostScript Language Tutorial and Cookbook (The Blue Book)
Classical Math Fractals in PostScript, Kees van der Laan,
https://www.ntg.nl/maps/44/07.pdf
Dragon curves revisited, S. Tabachnikov,
https://www.math.psu.edu/tabachni/prints/DragonCurves.pdf

Any special notes about your project (assumptions about unusual data, clarifications on specifications, information on how to run the code, etc.)?

The dragon curve PostScript code in the file fourdragons.ps is based off an example published by van der Laan (Classical Math Fractals). Van der Laan's code did not work at first, so I worked through it by hand to find the mistake. That gave me the understanding I needed to write my own dragon curve. I was then inspired by Tabachnikov to interlace four curves for my program.

Van der Laan is clearly cited in the program.

Any additional work (extra credit, fancy interfaces, etc.)?

Explain how you tested your project.

Printing the page after so many changes.

What went well?

The Blue Book was the most clearly written tutorial I have ever seen. The examples and explanations made learning PostScript relatively easy.

What problems did you have?

Working through rotation for the four dragons. It seemed simple enough on the surface, but I was making a false assumption regarding rotation. This lead to dragon curves that layered instead of nesting, and it took me a while to realize that my assumption was wrong.

Any surprises?

How much fun fractals are in PostScript

If you used a language new (or almost new) to you, explain how your learning process went.
Language: PostScript

As above, the Blue Book made learning PostScript relatively easy. I still found myself forgetting Polish notation, and if statements were a struggle, but all in all, the Tutorial and Cookbook made this a fun project.

Any remaining questions?

Other comments on the project?

Other discussions for a specific project, if any (see project specifications):
Questions below copied from Postscript specs by Dr. J. Gurka:

a. What development and execution environment(s) did you use?

   Notepad++ and PS2PDF: *https://www.ps2pdf.com/convert-ps-to-pdf*

b. What was easiest about learning and using Postscript?

   Following the Tutorial and Cookbook made learning the language realtively easy.

c. What was hardest about learning and using Postscript?

   Thinking of the page as points (I resorted to using an inch procedure in a few programs).

d. Explain your process for learning this new language. What did you do after the introduction in class? How much more code reading time did you put in? What other sample code did you run or write? How did you decide what you needed to do to learn the language better as you went along?

   I went through the Tutorial and Cookbook and rewrote 90% of its examples. I didn't bother with GhostScript beyond the first few days. I found it near impossible to make it run consistently on my Windows machine. Instead I used an online converter to view my files as .pdf files.

   After the smiles and business card programs, I looked for example code of fractal designs. That led me to van der Laan's paper *Classical Math Fractals in PostScript*. I read through his programs and ran a few with my own changes to understand what was happening. I feel I reached my highest level of understanding after I found one of van der Laan's programs that didn't work. I used pencil and paper to work through each line of the program, emulating putting data on the stack and taking it off with each operation. Not only did I find the mistake in van der Laan's code (a simple typo not obvious to a beginner), but I gained a strong enough understanding of the code to write my own version.

e. What did you find interesting, powerful, quirky, etc., about the language and programming in it?

   These days, with powerful graphics programs (often based on PostScript) with easy and intuitive user interfaces, I didn't really see the value of learning PostScript beyond being exposed to Polish notation. Then I read papers on fractal designs that either used PostScript directly, or that could use PostScript cleanly and elegantly, and I really began to see the appeal.

   Writing a business card in PostScript seems a little silly, edging on "hipster," but I can now see the power in incorporating PostScript into another application.

f. For each picture, state the required (above) Postscript capabilities you used (e.g., "save and restore").

   1. Smiles: color, translate and scale
   2. Business card: color, translate and rotate
   3. Structure (tree): translate, gsave, and grestore

4. Geometric form (rainbow triangles): translate, color, and rotate
5. Choice (four dragons): translate, color, rotate

g. Any other particulars on specific pictures (clever things you did, alternatives you tried, etc.)?

   As we had a bit more time in this project, I tried to go back and refactor my code, moving as much as possible into reusable procedures.

h. Any other comments on the language (as a new user and/or student in CS 3210)?

   As I said earlier, at first, I didn't really see the value in learning PostScript beyond learning Polish notation (which is a valuable thing to learn). However, when I started looking into the power PostScript has in displaying fractal designs, I got excited. I plan to continue playing with the language to learn to write truly recursive fractals.