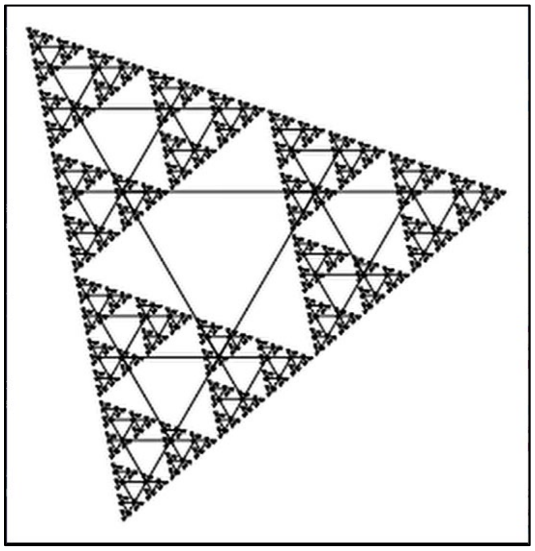
PS5: Recursive Graphics (Triangle Fractal)

In this assignment you will write a program that plots a Triangle Fractal as illustrated below.



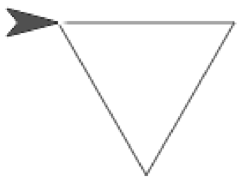
It is a variation of the *Sierpinski triangle*. The Polish mathematician Wacław Sierpiński described the pattern in 1915, but it has appeared in Italian art since the 13th century.

API specification

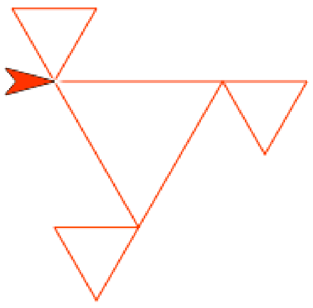
Your task is to write a program `TFractal.cpp` with a recursive function `fTree()`, and a `main()` program that calls the recursive function.

Your program shall take two command-line arguments *L* and *N*:

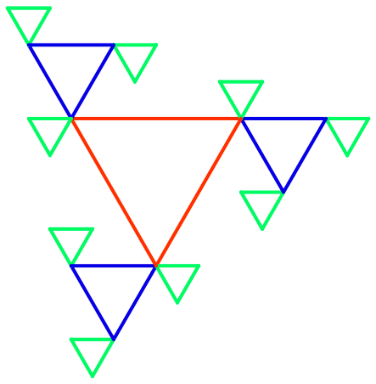
L	length of the side of the base equilateral triangle (double)
N	the depth of the recursion (int)



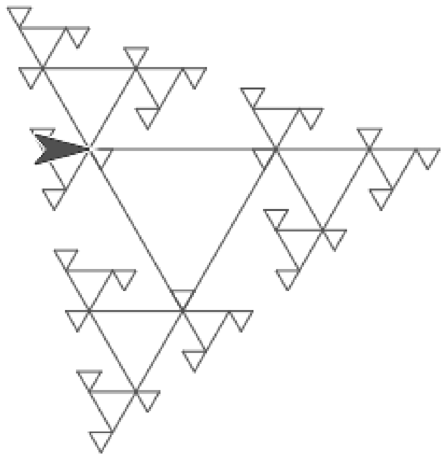
Base triangle

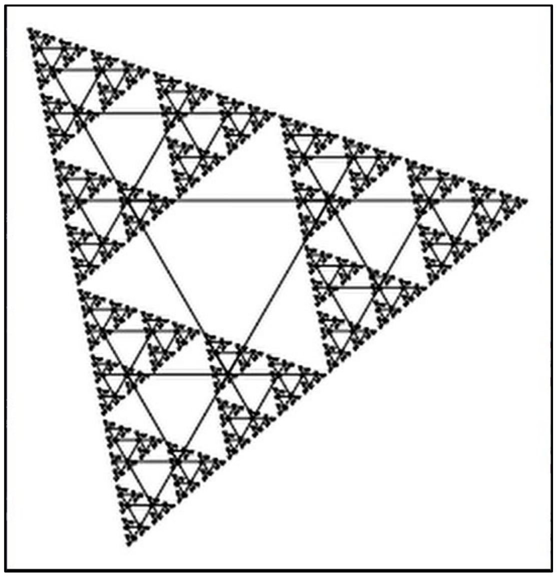


Iteration 1



Iteration 2


Iteration 3


Final triangle

You should implement class `Triangle`.

Notes:

- You should create a `Triangle` class that derives from `sf::Drawable`. Then, you can have it just `draw` itself to your main window.
- Your executable must read two parameters (integers): size of base Triangle and recursion-depth. You should create a SFML window that's your final Triangle should fit in.

Submit your work

It's important that you turn in everything needed to build your projects.

Create a directory with all your work**.**

Your `Makefile` should contain two targets: `all` and `clean`. The former should build both executables, and the latter should remove the executables `.o` files, and all other temporary files created during the build.

The directory should be named `ps5` and contain:

1. Your `TFractal.cpp`
2. Your `Triangle.cpp` and associated `Triangle.h`
3. Your `Makefile`
4. Anything else needed to build and run your code
5. Screenshot of program output
6. A `ps5-readme.txt` file

Remember, we will have to build and run your code, so make sure to submit all that's needed!

Use `tar` command from the parent directory of your `ps1`:

```
  tar czvf ''<archive-file-name>''.tar.gz ps5
```

to compress your directory structure.

Submit the archive on Blackboard.

Grading rubric

Feature	**Value**	**Comment**
implementation	9	9 pts for full & correct implementation – draws tree properly (recursive implementation) 2 pts – started implementation, draw base triangle - 2 points for non-recursive implementation
`Makefile`	1	
`readme`	1	Readme should say something meaningful about what you accomplished
	1	Your code should pass `cpplint`
Total	**12**	
extra credit	1 1	Add color to your tree Other (reasonable) added futures, i.e. animation. MUST explain your added futures in readme