CSE102 – Computer Programming with C (Spring 2020)

Term Project PART-2

Vector Graphics with EPS

Handed out: 9:00 pm May 19, 2020.

Due: 23:55 pm, June 19, 2020.

Hand-in Policy: Hand in via Moodle. No late submissions will be accepted.
Collaboration Policy: No collaboration is permitted.
Grading: This project may contribute up to 5 points towards your final grade out of 100. Second part will introduce another five point.

Summary: Understanding repeating patterns: https://www.youtube.com/watch?v=pg1NpMmPv48

While the video in the link is a very good descrption of the traditional geometric art, we will employ a much simpler approach. *Which is called Hankin's method.*

In this method, like the video above, we select the geometry of the underlying grid which we will refer as **archetype.** Archetypes, may include Squares(4-Side), Pentagons(5-side),Hexagons(6-Side) etc… However You are expected to use only Squares and Hexagons while implementing this assignment.

For each side, we take the center of the edge as **contact point.** And we draw two rays pointing inside of the **tile**. The angle between rays and the origin side is called **contact angle** which is shown in the figure 1(a). This construction will give rays 2 times the number of sides. Then we will use another property that we will refer as **offset** which describes distance between the **contact point** and start of the ray figure 1(b). Then we will connect each of the by calculating their intersection points and produce the indivisible substructure of our pattern called motif. Applying this motif to our canvas repeatedly will create our final pattern.

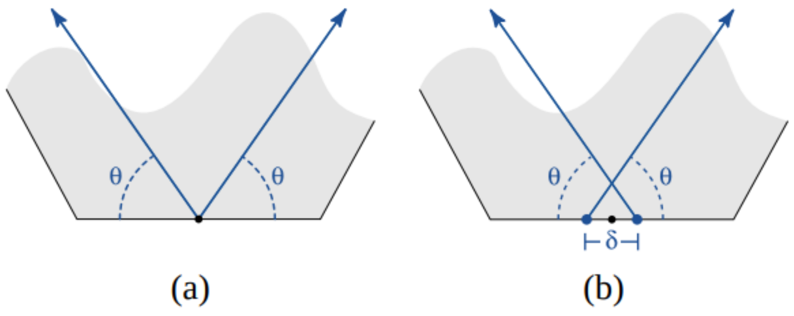

Figure 1 a) Shows the contact angle between two construction ray and b) shows the offset between center of the side and the start of the ray

Zip file includes an academic study on the subject, please read it carefully even though, we will give a brief explanation on the subject. We have also included a javascript demo file. You can execute the file by opening **index.html** file and play with the numbers and the fields to see the results.

In this part we want you to develop a software program that will generate the patterns described in a text file called **commands.txt**.

A command will define structure and visual properties of a pattern.

For ex:

background_color:(128,128,128),foreground_color:(255,0,0), tile_size:100,angle_offset:(50,10), canvas_size:(1000,1000), show_grid:True, archetype:Square, file_name:sq1.eps

Command above is telling that our software will generate a repeating pattern in square tiles with line angle of 50 and offset of 10px. Each tile cell has an edge length of 100px. background_color property sets the background color of the canvas and consists of three integers between 0 and 255. Similar to background_color, foreground_color describes the color of the pattern itself. show_grid value tells the program that generated image whether to show the underlying grid structure or not. This command file can include different commands that you have implemented in the first part of this term project. If the property is not defined then you will fall back to the default values, described in a file called **defaults.txt.** However, there are two mandatory fields, **archetype** and **file_name.** If these properties are not described, you will produce an error giving the errors line(command) number with an appropriate error message, and **continue executing rest of the commands file.**

Commands file will include one or more commands for each pattern description you will generate a new pattern and save it to the described file. Number of commands in the file is arbitrary. You should handle the commands as a dynamic data structure.

You can find the example definitions and commands in the attachments.

Rules

1. You should use dynamic memory allocation and data structures for defining geometric objects.
2. Separation of concerns: You should split your code files in to necessary **header** and **implementation** files.
3. You should build your own makefiles to compile and build necessary source files.

What to hand in: You are expected to hand in all your source code (library and test programs) along with your makefile in a ZIP or similarly archived filed named "**cse102project_lastname_firstname_studentno.zip**". When the makefile is run, it should compile everything and produce a test program. The test program should illustrate all the above functionality.