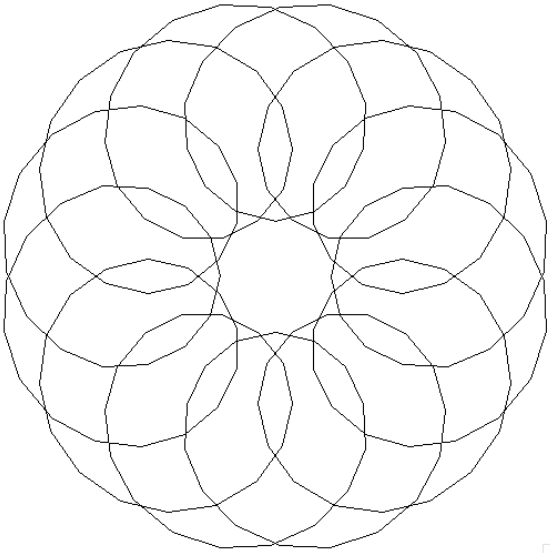
Assignment 1
Computer Graphics, Spring'22
IIT Kharagpur

Posted: 27th January 2022
Due: 9th February 2022, 11:55 PM

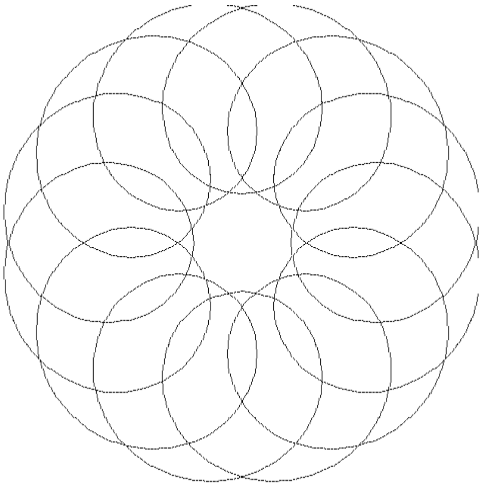
Description

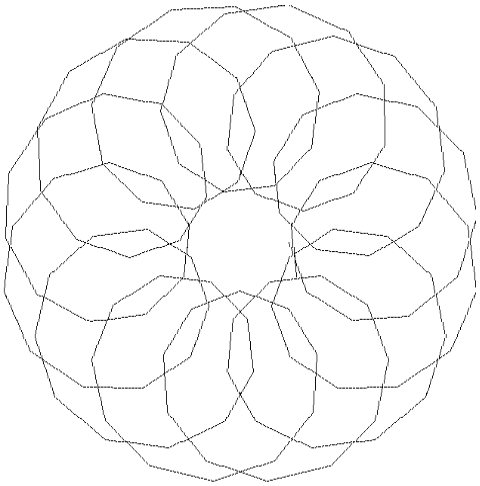
The purpose of this assignment is to apply Bresenham's line drawing algorithm to create an artistic design. Your job will be to create a spiral-like shape along the circumference of a circle using Bresenham's line drawing algorithm. DO NOT use circle drawing algorithm for this assignment. Your output should look similar to the following shape:



Notice that the circular shape is drawn using line segments. So your program should include all the slope cases of Bresenham's line drawing algorithm. You may use combination of sinusoid and/or cosine for generating the shape (or generating the path which will be followed by the lines). The following figure demonstrates the effect of choosing the discrete intervals for drawing the lines.

If the interval is small (i.e. small line segments), then the smooth circular shape is visible (the left figure), and if the interval is bigger, then the line segments are visible and the spiral does not look smooth (the right figure):





What to submit?

Submit the program file(s) you have implemented. Put all the file(s) into a zip and submit via MS Teams (no files will be accepted by email). Please do not submit any unnecessary files (such as the whole project).

Late penalty

The late policy is a penalty of 5% per day up to 3 days of lateness.

Plagiarism

Copying the code is a serious academic offence, which will be treated with zero tolerance. Remind that changing variable names and white spaces do not make your code unique, it's very easy to detect these cases using softwares.

General marking scheme

The marks will be distributed as follows:

- Working program: 70%
  - (will be divided into different parts of the assignment)

- Documentation: 10%
  - Main comment block identifying the student (name, roll number, email address): 4%
  - Defining input and output parameters for a function: 3%
  - Purpose of functions/blocks of code: 3%
- Program style: 10%
  - Meaningful variable names: 3%
  - Constants instead of “magic numbers”: 2%
  - Readability (complete sentences, indentation, white spaces, etc): 2%
  - Code flows “nicely”: 3%
- Program structure: 10%
  - Modular code: 4%
  - Uses appropriate data structure: 3%
  - Loops when needed/no loops when not needed: 3%