[CPSC-4050/6050, SPRING 2020] ASSIGNMENT # 01: LETS GET FEW THINGS STRAIGHT!

THIS IS A ONE WEEK ASSIGNMENT WITH SUBMISSION DATE AS 31st JAN, 2020.

The objective of this assignment is to practice concepts behind line drawing algorithms and put basic knowledge of vectors and coordinate systems to use. Apart from implementation of traditional line drawing algorithms such as DDA and Bresenham, you will exercise the notions of unconventional coordinate systems, plotting one basic shape made up of lines as well as get slightly creative in plotting lines with style variations. The idea of this assignment is also to push your understanding of line drawing when considering infinity of lines defined in a slope-intercept form as well as when the endpoints are not already rounded off to integer coordinates.

Objectives:

- Implement one of the basic line drawing algorithm DDA/Bresenham.
- Develop an approach to rasterize lines defined in slope-intercept form.
- Plot non-orthogonal axes and a simple native equivalent of a geometric construct (square/cube) in that coordinate system.
- Experiment with line stylizing.

Tasks and weightage:

4050	6050
A. (30%) Implement DDA for drawing line segments	A. Implement Bresenham's Algorithm for drawing line segments
B. (20%) Implement a function to draw 'complete' y=mx+c lines using implementation from part A	B. Implement a function to draw 'complete' y=mx+c lines using implementation from part A
C. (30%) Draw 2D lines with given arbitrarily orientated and centered coordinate system (i) draw axes (ii) draw a square	C. Draw 3D lines with given arbitrarily orientated and centered coordinate system (i) draw axes (ii) draw a cube
D. (20%) Draw one of the following stylized lines (i) dotted (ii) dashed (iii) dot and dash interleaved	D. Draw two of the following stylized lines (i) dotted (ii) dashed (iii) dot and dash interleaved
BONUS: (20%) for either of the below (i) Part A for 6050 (ii) Implement any two styles from part D with arbitrary line thickness in pixel units	BONUS: (i) Implement part A with arbitrary line thickness in pixel units (10%) (ii) Implement part D with arbitrary line thickness in pixel units (10%)

[CPSC-4050/6050, SPRING 2020] ASSIGNMENT # 01: LETS GET FEW THINGS STRAIGHT!

OS required: Linux, Compiler: C/C++, Build procedure: using make file. You are expected to work on general linux machines available for students @ SoC.

WorkFlow:

- Download the compressed package from https://clemson.instructure.com/courses/95982/files?preview=5989982 and save it in your local folder
- Open a terminal/command prompt and go to the local folder where your package is saved.
- Untar it with the command **tar –xzvf cg01.tar.** This will create a folder names **cg01**. Change your directory to this folder in your command prompt.
- This folder contains two source files main.cpp and stub.cpp and a Makefile file
- To build given package simply use the **make** command and it will produce an executable named **cg01.** Make sure you are in the same folder as the source code and the Makefile file. Now simply type the following at the prompt

  >> make
- There are no elaborate make options in this package. So, if you want to make a clean build, you will have to manually delete the .o files and the executable file **cg01**
- To run the default program type **./cg01** at the prompt and it will open an OpenGL window with 800x600 pixels. The default coordinate system for this 2D window has its origin at the lowest-leftmost corner for the window and positive X-axis points from left to right while the positive Y-axis points from bottom to top.
- You are only required to modify **stub.cpp** and provide implementation for respective functions in that file. You will only need to call **drawPixel()** from the rest of the package for this assignment and there is no need to write or use any OpenGL specific code explicitly. **You should not use any other OpenGL routine in you implementation and not change main.cpp**.
- Following is the mapping between given tasks and the functions in stub.cpp. Write your own code within designated functions and recompile all the files using 'make' command again. For a clean build, delete .o and cg01 files manually. **Make sure to backup your stub.cpp file periodically and not delete it accidentally during the cleanup.**

For CPSC4050 students

Task	Function
A. Implement DDA for drawing line segments	drawLineDDA()

[CPSC-4050/6050, SPRING 2020] ASSIGNMENT # 01: LETS GET FEW THINGS STRAIGHT!

B. Implement a function to draw 'complete' y=mx+c lines using implementation from part A	drawLineSlopeIntercept()
C. (30%) Draw 2D lines with given arbitrarily orientated and centered coordinate system (i) draw axes (ii) draw a square	draw2dAxesAndSquare()
D. (20%) Draw one of the following stylized lines - dotted - dashed - dot and dash interleaved	drawStyleDottedLine() drawStyleDashedLine() drawStyleDotAndDashLine() respectively.. Ignore last input parameter lineWidth
BONUS: (20%) for either of the below Part A for 6050 Implement any two styles from part D with arbitrary line thickness in pixel units	drawLineBresenham()
BONUS: (20%) Implement any two styles from part D with arbitrary line thickness in pixel units	drawStyleDottedLine() drawStyleDashedLine() drawStyleDotAndDashLine() respectively.. use last input parameter lineWidth

For CPSC6050 students

Task	Function
A. Implement Bresenham's Algorithm for drawing line segments	drawLineBresenham()
B. Implement a function to draw 'complete' y=mx+c lines using implementation from part A	drawLineSlopeIntercept()
C. Draw 3D lines with given arbitrarily orientated and centered coordinate system (i) draw axes (ii) draw a cube	draw3dAxesAndCube()
D. (20%) Draw two of the following stylized lines - dotted - dashed - dot and dash interleaved	drawStyleDottedLine() drawStyleDashedLine() drawStyleDotAndDashLine() respectively.. Ignore last input parameter lineWidth

[CPSC-4050/6050, SPRING 2020] ASSIGNMENT # 01: LETS GET FEW THINGS STRAIGHT!

BONUS: (20%) for either of the below Part A for 6050 Implement any two styles from part D with arbitrary line thickness in pixel units	drawLineBresenham()
BONUS: (i) Implement part A with arbitrary line thickness in pixel units (10%) (ii) Implement part D with arbitrary line thickness in pixel units (10%)	drawThickLineBresenham() drawStyleDottedLine() drawStyleDashedLine() drawStyleDotAndDashLine() respectively.. use last input parameter lineWidth

- All input variables for all functions above are named in a self-explanatory manner.
- The prompt level menu options are also self-explanatory. Kindly examine them well before starting your work.
- For some of the above tasks all the input values are required to be provided through prompt.
- For most of the line drawing tasks, end points have to be specified through mouse clicks and lines can be drawn repeatedly where a new mouse click is clubbed with the last mouse click to define two end points of the new line segment. To reset/deselect these endpoints, just press 'escape' key and start afresh for selecting two endpoints.
- A note on draw square/cube task: As discussed in class, a square is defined with its base at (sqrBaseX, sqrBaseX) and its sides stretching by sqrWidth units along specified +X and +Y directions for the non-orthogonal coordinate system. A cube is similarly defined in 3D.
- Prepare a PDF report with screen-grabs for the output of your implementation for each of the task.
- Prepare and upload a single zip file as your submission package that should include:
  - The above report
  - All your source code files and the makefile with a readme.txt if the need be.

Reference material:

- Chapters 2 upto 2.4, Chapter 5, Chapter 8.1.1 from the course book Fundamentals of Computer Graphics by Marschner and Shirley, 4th edition. You can access this book from Oreally's online resources. Consult your TA for details.
- Chapter 5 in the book "Computer Graphics with Open GL" by Hearn, Baker and Carithers, 4th Edition. For quick reference, relevant pages are uploaded at: https://clemson.instructure.com/courses/95982/files?preview=5990012

SUBMISSION DEADLINE: 11:59 p.m., 31st January (Friday) 2020.

[CPSC-4050/6050, SPRING 2020] ASSIGNMENT # 01: LETS GET FEW THINGS STRAIGHT!

- A delay of upto 24 hours will result in a proportional reduction of scores by 10%.
- A delay of upto 48 hours will result in a proportional reduction of scores by 20%.
- Any further delay will result in a proportional reduction of scores by 40%.

If there is a delay of for any medical exigency or any other unavoidable exigency, kindly contact me to avoid score reductions.

Upload your package over CANVAS