Jonathan Luu
998315852
jrluu@ucdavis.edu

Compiling the program:
1) Extract the files, and then type in "**make**" in the command line.
2)The program will be named "**poly.out**"
3) Type in "**poly.out**" in the command line to run it

Running the program:
1) Type in "**poly.out**" in the command line to run it
2) The program should prompt you for a response.
3) With an exception of the line drawing algorithms (DDA and Bresenham), the program processes one integer input at a time, so you MUST give only one input.
4) In the case of the line drawing algorithms, you must give two integers for each prompt

Closing the program:
1) Press cntl c to quit, or close the program using the x icon

Inputs(Entered in program):
This program is capable of loading polygons.
The program will prompt you for a file name by stating "Load File: "
If you don't want to input a file, simply type in a nonexistent file like "0.txt"
This will automatically rasterize your polygons.

File Format:
-of polygons
-number of points of 1st polygon
x y - coordinate 1
x y - coordinate 2
x y .... And so on.

For example, "rectangle.txt" contains

1
4
50 50
50 150
250 150
250 50

Which will create 1 polygon of 4 vertices.

For a working file example, please see "**test.txt**" or "**rectangle.txt**"

Outputs:
To save all of the polygons, select 9 on the menu.
This will save all of the polygons into "**saveFile.txt**" where it can be read back as an input.

Extra Information:
-The program CAN handle both concave and convex polygons!
There are cases where the program draws one extra line however

-The program only handles the first quadrant, and thus, it can only handle points in the positive direction.

-The program cannot handle coordinates outside the windows specified.
For example, if you attempt to draw a line from (0,0) to (200,200), while the view port is limited to a max window size of 100x100, the pixel buffer will overflow and potentially cause a seg fault.

-Scaling and Rotating occur with respect to the polygon's centriod
The polygons are ordered from 0 – (n-1), where n is the number of polygons.
If you had three polygons, the first polygon would be called polygon 0, the second would be 1, and the third or last one would be called polygon 2.

More information about Functions:

-Options 2 and 7 will not be saved.
-Only a polygon can be translated, rotated or scaled.

<u>DDA Algorithm</u>
Lines: 394 – 450 in Source.cpp, 303 – 381 in polygon.cpp
Creates a polygon class that is permanent. You can do all 2D Transformations on it, and save it.

<u>Bresenham Algorithm</u>
Lines: Source.cpp(682 – 737)
-IS NOT PERMANENT. It is there to demonstrate that the formula has been used.
-It will not create a polygon class.

<u>Rasterize/Scan Line</u>
Line: Polygon.cpp(166- 199)

<u>Translate:</u>
Line: Polygon.cpp(225- 245)
-Asks for two vectors, and move according to the vector.
-Changes will be saved.

Rotate
Lines: Polygon.cpp(248 - 270)
-Asks for a angle and rotates according to it
-Rotates about its centroids
- Changes will be saved.

Scale
Lines: Polygon.cpp(272- 288)
-Asks for two floats to expand by.
-Expands by its centroid
-Changes will be saved.

Line Clipping (Cohen-Sutherland):
Lines: Polygon.cpp(1052 – 1160) Source.cpp (500-600)
-Clipping for lines in a polygon class will be saved
-Clipping for option 7 is NOT saved.
-It is there to demonstrate mastery

Polygon Clipping(Sutherland-Hodgeman):
Lines: Polygon.cpp(485- 1048)
-Takes an input x_min, x_max, y_min, y_max, and clips ALL polygons in the view port.
-Clipping will be saved for all polygons affected.