**ECE 270**



Justin Newman

Quiz #17

SVG Drawing (C++)

November 25, 2014

# Statement of the Problem

The purpose of this program is to store the various parameters of shapes to be drawn in structures to be set by user input and passed to functions that will draw to an output file of the .SVG format. This program must make use of C++ Classes.

# Description of solution

Several classes are defined that contain the parameters for a given shape to be drawn, the functions that will be used to set the parameters, and the function that will write the output to the .svg file. This program uses the same logic as my Quiz #15 submission, but has been re-written in C++.

Two functions are set up for each type of shape, one to get user input for each shape and returns a structure to the main draw function which then will pass it to the second function, which will draw that type of shape.

The fill and stroke objects are declared within the draw function and are used and reset for each shape drawn.

UML:

|  |  |
| --- | --- |
| Fill | Circle |
| +red:int  +green:int  +blue:int  +opacity:float | +cx:float  +cy:float  +r:float |
| +Fill(r\_in:int, g\_in:int, b\_in:int, a\_in:float)  +Fill()  +setFill():void  +setFill(r\_in:int, g\_in:int, b\_in:int, a\_in:float)  +display():void | +Circle(cx\_in:float, cy\_in:float, r\_in:float)  +Circle()  +display():void  +setCircle():void  +setCenter(cx\_in:float, cy\_in:float):void  +setRadius(r\_in:float):  +drawCircle(fill:Fill,stroke:Stroke):void |
| Stroke | Rectangle |
| +red:int  +green:int  +blue:int  +opacity:float  +width:int | +xCorner:float  +yCorner:float  +width:float  +height:float |
| +Stroke(r\_in:int, g\_in:int, b\_in:int, a\_in:float, width\_in:int)  +Stroke()  +setStroke():void  +setStroke(r\_in:int, g\_in:int, b\_in:int, a\_in:float, width\_in:int) | +Rectangle(x1\_in:float, y1\_in:float, width\_in:float, height\_in:float)  +Rectangle()  +display():void  +setRect():void  +setCorner(x\_in:float, y\_in:float):void  +setDimensions(width\_in:float, height\_in:float):  +drawRect(fill:Fill,stroke:Stroke):void |
| Line | Path |
| +x1:float  +y1:float  +x2:float  +y2:float | +n:int  +\*x:\*float  +\*y:\*float |
| +Line(x1\_in: float, y1\_in: float, x2\_in: float, y2\_in: float)  +Line()  +display():void  +setLine():void  +setStart(x1\_in: float, y1\_in: float):void  +setEnd(x2\_in: float, y2\_in: float):void  +drawLine(stroke:Stroke):void | +Path(n\_in:int, \*x\_in:\*float, \*y\_in:\*float)  +Path()  +display():void  +setPath():void  +setPoint(n\_in:int, x\_in:float, y\_in:float):void  +drawPath(fill:Fill, stroke:Stroke):void |

# Output and Testing

This program was simple to test, but not so simple to debug. Testing amounted to verifying that the .SVG file was correctly produced and that the console output was properly recorded.

**Console output:**

Welcome to Justin Newman's ECE270 Quiz #17 C++ SVG drawing program

Please make your selection from the following menu:

L: Draw a line

C: Draw a circle

R: Draw a rectangle

P: Define your own shape

Q: To stop drawing

K: To draw a cool pre-defined graphic

Please enter your selection now:K

Program terminating...

enjoy your drawing

Your drawing can be found in image.svg in the same folder as this program

**SVG Output:**



# Code

**main.cpp:**

1 #include <iostream>

2 #include <cstdlib>

3 #include "./quiz17.h"

4 #include "./quiz17-implementation.cpp"

5

6 **char** fileName[] = "image.svg";

7 **int** imWidth = 1000; //image width (total)

8 **int** imHeight = 1000; //image height (total)

9

10 **void** writeSVGHeader(**char** [], **int** , **int** );

11 **void** writeSVGFooter();

12 **void** menu();

13 **void** draw();

14

15 **using namespace std**;

16

17 **int** main()

18 {

19 output=fopen("quiz17.txt","w");

20

21 writeSVGHeader(fileName, imWidth, imHeight);

22

23 draw();

24

25 writeSVGFooter();

26

27 **return** 0;

28 }

29

30 **void** writeSVGHeader(**char** fileName[], **int** width, **int** height)

31 {

32 //Open file for writing

33 svg = fopen(fileName,"w");

34

35 //Setup SVG header

36 fprintf(svg,"<?xml version='1.0' standalone='no'?>");

37 fprintf(svg,"\n<svg xmlns='http://www.w3.org/2000/svg' xmlns:xlink='http://www.w3.org/1999/xlink' version='1.1' width = '%d' height = '%d'>", width, height);

38 }

39

40 **void** writeSVGFooter()

41 {

42 //Closing SVG tag and close file

43 fprintf(svg,"\n</svg>");

44 fclose(svg);

45 }

46

47 **void** draw()

48 {

49 Line line(0,0,0,0);

50 Circle circle(0,0,0);

51 Rectangle rect(0,0,0,0);

52 Stroke stroke(0,0,0,0,0);

53 Fill **fill**(0,0,0,0);

54 Path path(0,(**float**\*)calloc(1,**sizeof**(**float**)),(**float**\*)calloc(1,**sizeof**(**float**)));

55 **char** selection=0;

56

57 menu();

58

59 **do**

60 {

61 **cin** >> selection;

62

63 **switch**(selection)

64 {

65 **case** 'L':**case** 'l':

66 system("cls");

67 stroke.setStroke();

68 line.setLine();

69 line.drawLine(stroke);

70 menu();

71 **break**;

72 **case** 'C':**case** 'c':

73 system("cls");

74 **fill**.setFill();

75 stroke.setStroke();

76 circle.setCircle();

77 circle.drawCircle(**fill**,stroke);

78 menu();

79 **break**;

80 **case** 'R':**case** 'r':

81 system("cls");

82 **fill**.setFill();

83 stroke.setStroke();

84 rect.setRect();

85 rect.drawRect(**fill**,stroke);

86 menu();

87 **break**;

88 **case** 'P':**case** 'p':

89 system("cls");

90 **fill**.setFill();

91 stroke.setStroke();

92 path.setPath();

93 path.drawPath(**fill**,stroke);

94 menu();

95 **break**;

96 **case** 'K':**case** 'k':

97 system("cls");

98 **fill**.setFill(30,30,30,1.0);

99 stroke.setStroke(0,0,0,0,0);

100 rect.setCorner(0,0);

101 rect.setDimensions(1000,200);

102 rect.drawRect(**fill**,stroke);

103 **fill**.setFill(213,43,30,1.0);

104 stroke.setStroke(0,0,0,0,0);

105 rect.setCorner(0,200);

106 rect.setDimensions(1000,200);

107 rect.drawRect(**fill**,stroke);

108 **fill**.setFill(254,209,0,1.0);

109 stroke.setStroke(0,0,0,0,0);

110 rect.setCorner(0,400);

111 rect.setDimensions(1000,200);

112 rect.drawRect(**fill**,stroke);

113 selection='Q';

114 system("cls");

115 **cout** << "\nProgram terminating...\nenjoy your drawing";

116 fprintf(output,"\nProgram terminating...\nenjoy your drawing");

117

118 **cout** << "\nYour drawing can be found in image.svg in the same folder as this program";

119 fprintf(output,"\nYour drawing can be found in image.svg in the same folder as this program");y

120 **break**;

121 **case** 'Q':**case** 'q':

122 system("cls");

123 **cout** << "\nProgram terminating...\nenjoy your drawing";

124 fprintf(output,"\nProgram terminating...\nenjoy your drawing");

125

126 **cout** << "\nYour drawing can be found in image.svg in the same folder as this program";

127 fprintf(output,"\nYour drawing can be found in image.svg in the same folder as this program");

128 **break**;

129 **default** :

130 system("cls");

131 **cout** << "\nPlease make a valid selection from the following menu:\n";

132 fprintf(output,"\nPlease make a valid selection from the following menu:\n");

133 menu();

134 }

135 }**while**(selection!='Q'&&selection!='q');

136 }

137

138 **void** menu()

139 {

140 **cout** << "\nWelcome to Justin Newman's ECE270 Quiz #17 C++ SVG drawing program";

141 fprintf(output,"\nWelcome to Justin Newman's ECE270 Quiz #17 C++ SVG drawing program");

142

143 **cout** << "\n\nPlease make your selection from the following menu:";

144 fprintf(output,"\n\nPlease make your selection from the following menu:");

145

146 **cout** << "\n\nL:\tDraw a line";

147 fprintf(output,"\n\nL:\tDraw a line");

148

149 **cout** << "\nC:\tDraw a circle";

150 fprintf(output,"\nC:\tDraw a circle");

151

152 **cout** << "\nR:\tDraw a rectangle";

153 fprintf(output,"\nR:\tDraw a rectangle");

154

155 **cout** << "\nP:\tDefine your own shape";

156 fprintf(output,"\nP:\tDefine your own shape");

157

158 **cout** << "\nQ:\tTo stop drawing";

159 fprintf(output,"\nQ:\tTo stop drawing");

160

161 **cout** << "\nK:\tTo draw a cool pre-defined graphic";

162 fprintf(output,"\nK:\tTo draw a cool pre-defined graphic");

163

164 **cout** << "\n\nPlease enter your selection now:";

165 fprintf(output,"\n\nPlease enter your selection now:");

166 }

**Quiz17.h:**

1 /\*Justin Newman

2 ECE270 Quiz 17

3 11/24/14\*/

4 #ifndef QUIZ17\_H

5 #define QUIZ17\_H

6

7 **class** Fill{

8 **public**:

9 **int** red; //set red tone 0-255

10 **int** green; //set green tone 0-255

11 **int** blue; //set blue tone 0-255

12 **float** opacity; //set opacity 0.0-1.0

13

14 Fill(**int** r\_in, **int** g\_in, **int** b\_in, **float** a\_in);

15 Fill();

16 **void** setFill();

17 **void** setFill(**int** r\_in, **int** g\_in, **int** b\_in, **float** a\_in);

18 **void** display();

19 };

20

21 **class** Stroke{

22 **public**:

23 **int** red; //set red tone 0-255

24 **int** green; //set green tone 0-255

25 **int** blue; //set blue tone 0-255

26 **float** opacity; //set opacity 0.0-1.0

27 **int** width; //set width

28

29 Stroke(**int** r\_in,**int** g\_in, **int** b\_in, **float** a\_in, **int** width\_in);

30 Stroke();

31 **void** setStroke();

32 **void** setStroke(**int** r\_in,**int** g\_in, **int** b\_in, **float** a\_in, **int** width\_in);

33 };

34

35 **class** Line{

36 **public**:

37 **float** x1; //x-coord of line start

38 **float** y1; //y-coord of line start

39 **float** x2; //x-coord of line end

40 **float** y2; //y-coord of line end

41

42 Line(**float** x1\_in, **float** y1\_in, **float** x2\_in, **float** y2\_in);

43 Line();

44 **void** display();

45 **void** setLine();

46 **void** setStart(**float** x1\_in, **float** y1\_in);

47 **void** setEnd(**float** x2\_in, **float** y2\_in);

48 **void** drawLine(Stroke stroke);

49 };

50

51 **class** Circle{

52 **public**:

53 **float** cx; //center x-coordinate

54 **float** cy; //center y-coordinate

55 **float** r; //radius

56

57 Circle(**float** cx\_in, **float** cy\_in, **float** r\_in);

58 Circle();

59 **void** display();

60 **void** setCircle();

61 **void** setCenter(**float** cx\_in, **float** cy\_in);

62 **void** setRadius(**float** r\_in);

63 **void** drawCircle(Fill **fill**,Stroke stroke);

64 };

65

66

67 **class** Rectangle{

68 **public**:

69 **float** xCorner; //x-coord of upper-left corner of rectangle

70 **float** yCorner; //y-coord of upper-left corner of rectangle

71 **float** width; //horizontal size of rectangle

72 **float** height; //vertical size of rectangle

73

74 Rectangle(**float** x1\_in, **float** y1\_in, **float** width\_in, **float** height\_in);

75 Rectangle();

76 **void** display();

77 **void** setRect();

78 **void** setCorner(**float** x\_in, **float** y\_in);

79 **void** setDimensions(**float** width\_in, **float** height\_in);

80 **void** drawRect(Fill **fill**, Stroke stroke);

81 };

82

83

84 **class** Path{

85 **public**:

86 **int** n; //number of points in path

87 **float** \*x; //array of x-coordinates

88 **float** \*y; //array of y-coordinates

89

90 Path(**int** n, **float** \*x\_in, **float** \*y\_in);

91 Path();

92 **void** display();

93 **void** setPath();

94 **void** setPoint(**int** n\_in, **float** x\_in, **float** y\_in);

95 **void** drawPath(Fill **fill**,Stroke stroke);

96 };

97 #endif

**Quiz17-implementation.cpp:**

1 /\*Justin Newman

2 ECE270 Quiz 17

3 11/24/14\*/

4

5 #include <iostream>

6 #include "./quiz17.h"

7 #include <stdio.h>

8 #include <cstdlib>

9 **using namespace std**;

10

11 FILE \*svg;

12 FILE \*output;

13

14

15 Fill::Fill(**int** r\_in, **int** g\_in, **int** b\_in, **float** a\_in)

16 {

17 red=r\_in;

18 green=g\_in;

19 blue=b\_in;

20 opacity=a\_in;

21 }

22

23 Fill::Fill()

24 {

25 **cout** << "\nPlease enter the desired amount of red for your shape's fill color (0-255)";

26 **cin** >> red;

27 fprintf(output,"\nPlease enter the desired amount of red for your shape's fill color (0-255): %d",red);

28

29 **cout** << "\nPlease enter the desired amount of green for your shape's fill color (0-255)";

30 **cin** >> green;

31 fprintf(output,"\nPlease enter the desired amount of green for your shape's fill color (0-255): %d",green);

32

33 **cout** << "\nPlease enter the desired amount of blue for your shape's fill color (0-255)";

34 **cin** >> blue;

35 fprintf(output,"\nPlease enter the desired amount of red for your shape's fill color (0-255): %d",blue);

36

37 **cout** << "\nPlease enter the desired opacity of your shape's fill color (0.0-1.0)";

38 **cin** >> opacity;

39 fprintf(output,"\nPlease enter the desired opacity of shape's fill color (0.0-1.0): %f",opacity);

40 }

41 **void** Fill::setFill()

42 {

43 **cout** << "\nPlease enter the desired amount of red for your shape's fill color (0-255)";

44 **cin** >> red;

45 fprintf(output,"\nPlease enter the desired amount of red for your shape's fill color (0-255): %d",red);

46

47 **cout** << "\nPlease enter the desired amount of green for your shape's fill color (0-255)";

48 **cin** >> green;

49 fprintf(output,"\nPlease enter the desired amount of green for your shape's fill color (0-255): %d",green);

50

51 **cout** << "\nPlease enter the desired amount of blue for your shape's fill color (0-255)";

52 **cin** >> blue;

53 fprintf(output,"\nPlease enter the desired amount of red for your shape's fill color (0-255): %d",blue);

54

55 **cout** << "\nPlease enter the desired opacity of your shape's fill color (0.0-1.0)";

56 **cin** >> opacity;

57 fprintf(output,"\nPlease enter the desired opacity of shape's fill color (0.0-1.0): %f",opacity);

58 }

59

60 **void** Fill::setFill(**int** r\_in, **int** g\_in, **int** b\_in, **float** a\_in)

61 {

62 red=r\_in;

63 green=g\_in;

64 blue=b\_in;

65 opacity=a\_in;

66 }

67

68 Stroke::Stroke(**int** r\_in, **int** g\_in, **int** b\_in, **float** a\_in, **int** width\_in)

69 {

70 red=r\_in;

71 green=g\_in;

72 blue=b\_in;

73 opacity=a\_in;

74 width=width\_in;

75 }

76

77 Stroke::Stroke()

78 {

79 **cout** << "\nPlease enter the desired amount of red for your shape's Stroke color (0-255)";

80 **cin** >> red;

81 fprintf(output,"\nPlease enter the desired amount of red for your shape's Stroke color (0-255): %d",red);

82

83 **cout** << "\nPlease enter the desired amount of green for your shape's Stroke color (0-255)";

84 **cin** >> green;

85 fprintf(output,"\nPlease enter the desired amount of green for your shape's Stroke color (0-255): %d",green);

86

87 **cout** << "\nPlease enter the desired amount of blue for your shape's Stroke color (0-255)";

88 **cin** >> blue;

89 fprintf(output,"\nPlease enter the desired amount of red for your shape's Stroke color (0-255): %d",blue);

90

91 **cout** << "\nPlease enter the desired opacity of your shape's Stroke color (0.0-1.0)";

92 **cin** >> opacity;

93 fprintf(output,"\nPlease enter the desired opacity of shape's Stroke color (0.0-1.0): %.2f",opacity);

94

95 **cout** << "\nPlease enter the desired width of your shape's Stroke";

96 **cin** >> width;

97 fprintf(output,"\nPlease enter the desired width of shape's Stroke: %d",width);

98 }

99

100 **void** Stroke::setStroke()

101 {

102 **cout** << "\nPlease enter the desired amount of red for your shape's Stroke color (0-255)";

103 **cin** >> red;

104 fprintf(output,"\nPlease enter the desired amount of red for your shape's Stroke color (0-255): %d",red);

105

106 **cout** << "\nPlease enter the desired amount of green for your shape's Stroke color (0-255)";

107 **cin** >> green;

108 fprintf(output,"\nPlease enter the desired amount of green for your shape's Stroke color (0-255): %d",green);

109

110 **cout** << "\nPlease enter the desired amount of blue for your shape's Stroke color (0-255)";

111 **cin** >> blue;

112 fprintf(output,"\nPlease enter the desired amount of red for your shape's Stroke color (0-255): %d",blue);

113

114 **cout** << "\nPlease enter the desired opacity of your shape's Stroke color (0.0-1.0)";

115 **cin** >> opacity;

116 fprintf(output,"\nPlease enter the desired opacity of shape's Stroke color (0.0-1.0): %.2f",opacity);

117

118 **cout** << "\nPlease enter the desired width of your shape's Stroke";

119 **cin** >> width;

120 fprintf(output,"\nPlease enter the desired width of shape's Stroke: %d",width);

121 }

122

123 **void** Stroke::setStroke(**int** r\_in, **int** g\_in, **int** b\_in, **float** a\_in, **int** width\_in)

124 {

125 red=r\_in;

126 green=g\_in;

127 blue=b\_in;

128 opacity=a\_in;

129 width=width\_in;

130 }

131

132 Line::Line(**float** x1\_in, **float** y1\_in, **float** x2\_in, **float** y2\_in)

133 {

134 x1=x1\_in;

135 y1=y1\_in;

136 x2=x2\_in;

137 y2=y2\_in;

138 }

139

140 Line::Line()

141 {

142 **cout** << "\nPlease enter the starting x-coordinate of your line";

143 **cin** >> x1;

144 fprintf(output,"\nPlease enter the starting x-coordinate of your line: %.2f",x1);

145

146 **cout** << "\nPlease enter the starting y-coordinate of your line";

147 **cin** >> y1;

148 fprintf(output,"\nPlease enter the starting y-coordinate of your line: %.2f",y1);

149

150 **cout** << "\nPlease enter the ending x-coordinate of your line";

151 **cin** >> x2;

152 fprintf(output,"\nPlease enter the ending x-coordinate of your line: %.2f",x2);

153

154 **cout** << "\nPlease enter the ending y-coordinate of your line";

155 **cin** >> y2;

156 fprintf(output,"\nPlease enter the ending y-coordinate of your line: %.2f",y2);

157

158 system("cls");

159 }

160

161 **void** Line::display()

162 {

163 **cout** << "\nx1:" << x1;

164 **cout** << "\ny1:" << y1;

165 **cout** << "\nx2:" << x2;

166 **cout** << "\ny2:" << y2;

167 }

168

169 **void** Line::setLine()

170 {

171 **cout** <<"\nPlease enter the starting x-coordinate of your line";

172 **cin** >> x1;

173 fprintf(output,"\nPlease enter the starting x-coordinate of your line: %.2f",x1);

174

175 **cout** << "\nPlease enter the starting y-coordinate of your line";

176 **cin** >> y1;

177 fprintf(output,"\nPlease enter the starting y-coordinate of your line: %.2f",y1);

178

179 **cout** <<"\nPlease enter the ending x-coordinate of your line";

180 **cin** >> x2;

181 fprintf(output,"\nPlease enter the ending x-coordinate of your line: %.2f",x2);

182

183 **cout** << "\nPlease enter the ending y-coordinate of your line";

184 **cin** >> y2;

185 fprintf(output,"\nPlease enter the ending y-coordinate of your line: %.2f",y2);

186

187 system("cls");

188 }

189

190 **void** Line::setStart(**float** x1\_in, **float** y1\_in)

191 {

192 x1=x1\_in;

193 y1=y1\_in;

194

195 }

196

197 **void** Line::setEnd(**float** x2\_in, **float** y2\_in)

198 {

199 x2=x2\_in;

200 y2=y2\_in;

201 }

202

203 **void** Line::drawLine(Stroke stroke)

204 {

205 fprintf(svg,"\n <line x1 = '%f' y1 = '%f' x2 = '%f' y2 = '%f'", x1, y1, x2, y2);

206 fprintf(svg," stroke = 'rgb(%d, %d, %d)' stroke-opacity = '%f'", stroke.red, stroke.green, stroke.blue, stroke.opacity);

207 fprintf(svg," stroke-width = '%d' />", stroke.width);

208 }

209

210 Circle::Circle(**float** cx\_in, **float** cy\_in, **float** r\_in)

211 {

212 cx=cx\_in;

213 cy=cy\_in;

214 r=r\_in;

215 }

216

217 Circle::Circle()

218 {

219 **cout** <<"\nPlease enter the center x-coordinate of your circle";

220 **cin** >> cx;

221 fprintf(output,"\nPlease enter the center x-coordinate of your circle: %.2f",cx);

222

223 **cout** << "\nPlease enter the center y-coordinate of your circle";

224 **cin** >> cy;

225 fprintf(output,"\nPlease enter the center y-coordinate of your circle: %.2f",cy);

226

227 **cout** <<"\nPlease enter the radius of your circle";

228 **cin** >> r;

229 fprintf(output,"\nPlease enter the radius of your circle: %.2f",r);

230

231 system("cls");

232 }

233

234 **void** Circle::display()

235 {

236 **cout** << "\ncx:" << cx;

237 **cout** << "\ncy:" << cy;

238 **cout** << "\nr:" << r;

239 }

240

241 **void** Circle::setCircle()

242 {

243 **cout** <<"\nPlease enter the center x-coordinate of your circle";

244 **cin** >> cx;

245 fprintf(output,"\nPlease enter the center x-coordinate of your circle: %.2f",cx);

246

247 **cout** << "\nPlease enter the center y-coordinate of your circle";

248 **cin** >> cy;

249 fprintf(output,"\nPlease enter the center y-coordinate of your circle: %.2f",cy);

250

251 **cout** <<"\nPlease enter the radius of your circle";

252 **cin** >> r;

253 fprintf(output,"\nPlease enter the radius of your circle: %.2f",r);

254

255 system("cls");

256 }

257

258 **void** Circle::setCenter(**float** cx\_in, **float** cy\_in)

259 {

260 cx=cx\_in;

261 cy=cy\_in;

262 }

263

264 **void** Circle::setRadius(**float** r\_in)

265 {

266 r=r\_in;

267 }

268

269 **void** Circle::drawCircle(Fill **fill**,Stroke stroke)

270 {

271 fprintf(svg,"\n <circle cx = '%f' cy = '%f' r = '%f'", cx, cy, r);

272 fprintf(svg," fill = 'rgb(%d, %d, %d)' fill-opacity = '%f'", **fill**.red, **fill**.green, **fill**.blue, **fill**.opacity);

273 fprintf(svg," stroke = 'rgb(%d, %d, %d)' stroke-opacity = '%f'", stroke.red, stroke.green, stroke.blue, stroke.opacity);

274 fprintf(svg," stroke-width = '%d' />", stroke.width);

275 }

276

277 Rectangle::Rectangle(**float** x\_in, **float** y\_in, **float** width\_in, **float** height\_in)

278 {

279 xCorner=x\_in;

280 yCorner=y\_in;

281 width=width\_in;

282 height=height\_in;

283 }

284

285 Rectangle::Rectangle()

286 {

287 **cout** <<"\nPlease enter the x-coordinate for the upper left corner of your Rectangle";

288 **cin** >> xCorner;

289 fprintf(output,"\nPlease enter the x-coordinate for the upper left corner of your Rectangle: %.2f",xCorner);

290

291 **cout** << "\nPlease enter the y-coordinate for the upper left corner of your Rectangle";

292 **cin** >> yCorner;

293 fprintf(output,"\nPlease enter the y-coordinate for the upper left corner of your Rectangle: %.2f",yCorner);

294

295 **cout** <<"\nPlease enter the width of your Rectangle";

296 **cin** >> width;

297 fprintf(output,"\nPlease enter the width of your Rectangle: %.2f",width);

298

299 **cout** << "\nPlease enter the height of your Rectangle";

300 **cin** >> height;

301 fprintf(output,"\nPlease enter the height of your Rectangle: %.2f",height);

302

303 system("cls");

304 }

305

306 **void** Rectangle::display()

307 {

308 **cout** << "\nx corner:" << xCorner;

309 **cout** << "\ny corner:" << yCorner;

310 **cout** << "\nwidth:" << width;

311 **cout** << "\nheight:" << height;

312 }

313

314 **void** Rectangle::setRect()

315 {

316 **cout** <<"\nPlease enter the x-coordinate for the upper left corner of your Rectangle";

317 **cin** >> xCorner;

318 fprintf(output,"\nPlease enter the x-coordinate for the upper left corner of your Rectangle: %.2f",xCorner);

319

320 **cout** << "\nPlease enter the y-coordinate for the upper left corner of your Rectangle";

321 **cin** >> yCorner;

322 fprintf(output,"\nPlease enter the y-coordinate for the upper left corner of your Rectangle: %.2f",yCorner);

323

324 **cout** <<"\nPlease enter the width of your Rectangle";

325 **cin** >> width;

326 fprintf(output,"\nPlease enter the width of your Rectangle: %.2f",width);

327

328 **cout** << "\nPlease enter the height of your Rectangle";

329 **cin** >> height;

330 fprintf(output,"\nPlease enter the height of your Rectangle: %.2f",height);

331

332 system("cls");

333 }

334

335 **void** Rectangle::setCorner(**float** x\_in, **float** y\_in)

336 {

337 xCorner=x\_in;

338 yCorner=y\_in;

339 }

340

341 **void** Rectangle::setDimensions(**float** width\_in, **float** height\_in)

342 {

343 width=width\_in;

344 height=height\_in;

345 }

346

347 **void** Rectangle::drawRect(Fill **fill**,Stroke stroke)

348 {

349 fprintf(svg,"\n <rect x = '%f' y = '%f' width = '%f' height = '%f'", xCorner, yCorner, width, height);

350 fprintf(svg," fill = 'rgb(%d, %d, %d)' fill-opacity = '%f'", **fill**.red, **fill**.green, **fill**.blue, **fill**.opacity);

351 fprintf(svg," stroke = 'rgb(%d, %d, %d)' stroke-opacity = '%f'", stroke.red, stroke.green, stroke.blue, stroke.opacity);

352 fprintf(svg," stroke-width = '%d' />", stroke.width);

353 }

354

355 Path::Path(**int** n\_in, **float** \*x\_in, **float** \*y\_in)

356 {

357 n=n\_in;

358 x=x\_in;

359 y=y\_in;

360 x=(**float**\*)calloc(n\_in,**sizeof**(**float**));

361 y=(**float**\*)calloc(n\_in,**sizeof**(**float**));

362 }

363

364 Path::Path()

365 {

366 **int** i;

367

368 **cout** << "\nPlease enter the number of points in your path";

369 **cin** >> n;

370 fprintf(output,"\nPlease enter the number of points in your Path: %d",n);

371

372 x=(**float**\*)calloc(n,**sizeof**(**float**));

373 y=(**float**\*)calloc(n,**sizeof**(**float**));

374

375 **for**(i=0;i<n;i++)

376 {

377 **cout** <<"\nPlease enter the x-coordinate for point #" << i << " of your path\n";

378 **cin** >> x[i];

379 fprintf(output,"\nPlease enter the x-coordinate for point #%d of your Path: %.2f",i,x[i]);

380

381 **cout** <<"\nPlease enter the y-coordinate for point #" << i << " of your path\n";

382 **cin** >> y[i];

383 fprintf(output,"\nPlease enter the y-coordinate for point #%d of your Path: %.2f",i,y[i]);

384 }

385

386 system("cls");

387 }

388

389 **void** Path::display()

390 {

391 **int** i;

392 **for**(i=0;i<n;i++)

393 {

394 **cout** <<"\nThe x-coordinate for point #" << i << " of your path is:" << x[i];

395 fprintf(output,"\nThe x-coordinate for point #%d of your Path is: %.2f",i,x[i]);

396

397 **cout** <<"\nThe y-coordinate for point #" << i << " of your path is" << y[i];

398 fprintf(output,"\nThe y-coordinate for point #%d of your Path is: %.2f",i,y[i]);

399 }

400 }

401

402 **void** Path::setPath()

403 {

404 **int** i;

405

406 **cout** << "\nPlease enter the number of points in your path";

407 **cin** >> n;

408 fprintf(output,"\nPlease enter the number of points in your Path: %d",n);

409

410 x=(**float**\*)calloc(n,**sizeof**(**float**));

411 y=(**float**\*)calloc(n,**sizeof**(**float**));

412

413 **for**(i=0;i<n;i++)

414 {

415 **cout** <<"\nPlease enter the x-coordinate for point #" << i << " of your path\n";

416 **cin** >> x[i];

417 fprintf(output,"\nPlease enter the x-coordinate for point #%d of your Path: %.2f",i,x[i]);

418

419 **cout** <<"\nPlease enter the y-coordinate for point #" << i << " of your path\n";

420 **cin** >> y[i];

421 fprintf(output,"\nPlease enter the y-coordinate for point #%d of your Path: %.2f",i,y[i]);

422 }

423

424 system("cls");

425 }

426

427 **void** Path::setPoint(**int** n\_in, **float** x\_in, **float** y\_in)

428 {

429 x[n]=x\_in;

430 y[n]=y\_in;

431 }

432

433 **void** Path::drawPath(Fill **fill**,Stroke stroke)

434 {

435 **int** i;

436

437 fprintf(svg,"\n <path d ='M%f %f ", x[0], y[0]);

438

439 **for**(i=1;i<n;i++)

440 {

441 fprintf(svg,"L%f %f ", x[i], y[i]);

442 }

443

444 fprintf(svg,"z' fill = 'rgb(%d, %d, %d)' fill-opacity = '%f'", **fill**.red, **fill**.green, **fill**.blue, **fill**.opacity);

445 fprintf(svg," stroke = 'rgb(%d, %d, %d)' stroke-opacity = '%f'", stroke.red, stroke.green, stroke.blue, stroke.opacity);

446 fprintf(svg," stroke-width = '%d' />", stroke.width);

447 }