

```

1  /*=====
2  PROGRAMMER:          Brett Story,      Kyle Falconer
3  FOLDERS:             Brett322,        Falconer1
4  BRETT'S TASKS:      Patterned polygon, pixelmap, storing file (Contribution: 50%)
5  KYLE'S TASKS:       Circle, Bitmap, text      (Contribution: 50%)
6  COURSE:              CSC 525/625
7  MODIFIED BY:         N/A
8  LAST MODIFIED DATE: Oct. 5, 2013
9  DESCRIPTION:         Demo: drawing points.
10 NOTE:                Alpha transparencies were implemented below the desk to give
11                      a shadow look. The image in the background was loaded from a
12                      C struct, which the image editing program GIMP can export to.
13
14                      We worked on the assignment together in person and completed
15                      an equal number of tasks, allotted to about 50% each.
16
17  FILES:               h3.cpp, (hwProject.sln, ...)
18  IDE/COMPILER:        MicroSoft Visual Studio 2012
19  INSTRUCTION FOR COMPILATION AND EXECUTION:
20      1.      Double click on labProject.sln to OPEN the project
21      2.      Press Ctrl+F7 to COMPILE
22      3.      Press Ctrl+Shift+B to BUILD (COMPILE+LINK)
23      4.      Press Ctrl+F5 to EXECUTE
24  =====*/
25 #define _USE_MATH_DEFINES
26 #include <iostream>
27 #include <GL/glut.h>          // include GLUT library
28 #include <cmath>              // include math library
29 #include <string>
30 using namespace std;
31
32 // To allow file reading
33 #include <fstream>
34 using std::ifstream;
35
36 // To exit if the file doesn't exist
37 #include <cstdlib>
38
39 //*****
40
41 GLfloat PixelsRead[786432];
42
43 // Is the pattern used for the wall paper
44 GLubyte WallPaperPattern[128] = {0X00, 0X00, 0X00, 0X00,
45     0X00, 0X00, 0X00, 0X00,
46     0X00, 0X00, 0X00, 0X00,
47     0X00, 0X00, 0X00, 0X00,
48
49     0X00, 0X0F, 0XF0, 0X00,
50     0X00, 0X0F, 0XF0, 0X00,
51     0X00, 0X0F, 0XF0, 0X00,
52     0X00, 0X0F, 0XF0, 0X00,
53
54     0X00, 0XFF, 0XFF, 0X00,

```

-2-

```
109 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
110 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
111 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
112 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
113 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
114 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
115 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
116 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
117 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
118 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
119 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
120 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
121 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
122 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
123 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
124 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
125 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
126 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
127 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
128 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
129 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
130 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
131 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
132 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
133 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
134 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
135 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
136 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
137 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
138 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
139 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
140 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
141 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
142 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
143 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
144 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
145 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\315_\327\323"
146 "a\327\315]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
147 "j\327\312]\327\312]\327\315_\327\323_\327\315]\327\312]\327\312]\327\312]"
148 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
149 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\315_\273\260s?<\35\313"
150 "\277Y\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
151 "\312]\327\312]\304\270VLI(\302\267V\327\315_\327\312]\327\312]\327\312]\327"
152 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
153 "\312]\327\312]\327\312]\327\315_\247\235I\35\34\17\5\5\5:8\34\327\320_\327"
154 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\320_>;$\25"
155 "\25\25.,\36\260\246P\327\315_\327\312]\327\312]\327\312]\327\312]\327\312]"
156 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]"
157 "j\276\263T\26\25\14\1\1\3\5\5\5\7\7\7\216\207>\327\312]\327\312]\327\312]"
158 "j\327\312]\327\312]\327\312]\327\312]\207\201?\24\24\24\25\25\25\24\24\26"
159 "+)\37\304\273V\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
160 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\315],*\26\2\2\4\5"
161 "\5\5\6\6\6\7\7\7\33\33\21\327\312]\327\312]\327\312]\327\312]\327\312]\327"
162 "\312]\323\306[#"\31\25\25\25\27\27\27\31\31\31\30\30\32FC*\327\315]\327"
```

```
163 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
164 "\312]\327\312]\327\312]\250\236J\2\2\4\5\5\5\6\6\6\10\10\10\11\11\11\12\12"
165 "\12JF$\327\315_\327\312]\327\312]\327\312]\327\315_HE(\25\25\25\30\30\30"
166 "\31\31\31\33\33\33\34\34\34\35\35\35\263\251Q\327\312]\327\312]\327\312]"
167 "\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\315_"
168 "IE!\5\5\5\7\7\7\10\10\10\11\11\11\12\12\12\14\14\14\20\20\16\240\226F\327"
169 "\315_\327\315_\327\315_\227\220E\30\30\26\30\30\30\31\31\31\32\32\32\34\34"
170 "\34\35\35\35\37\37\37fb8\327\315]\327\312]\327\312]\327\312]\327\312]\327"
171 "\312]\327\312]\327\312]\327\312]\327\312]\327\315]\10\10\6\6\6\6\10\10\10"
172 "\11\11\11\13\13\13\14\14\14\16\16\16\20\20\20\271\256Q\247\237K\206\200>"
173 "\255\242M\261\251O\26\26\26\32\32\32\33\33\33\35\35\35\36\36\36\37\37\37"
174 "!!!++%\327\315]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
175 "\312]\327\312]\327\312]\317\303[\1\1\3\6\6\10\7\7\11\12\12\12\12\12\14\13"
176 "\13\15\14\14\16\203~==zu:\40\37\30\36\35\30##\33\207\201Aa]3\30\31\32\33\33"
177 "\35\34\34\36\37\37\37!!!!\"\\"31\32!\323\307]\327\312]\327\312]\327\312"
178 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\316\302Z\17\17\13"
179 "\33\33\21\34\34\22\36\35\24\37\36\25!!\27#\\"27\316\302Z\31\31\25\23\23\25"
180 "\27\27\27\30\30\30\40\40\34\302\267V,,\"..$/%10'21(33)++'\323\307]\327\312"
181 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
182 "j\326\311^\306\272V\307\273W\307\273W\307\273W\307\273W\307\273W\307\273"
183 "W\324\310^\14\15\22\27\27\27\31\31\31\32\32\32\24\24\30\324\310^\310\274"
184 "X\310\274X\310\274X\311\275Y\311\275Y\311\275Y\310\275Z\327\312]\327\312"
185 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
186 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
187 "j\327\312]to8\21\22\25\26\26\30\24\24\30\177z?\327\312]\327\312]\327\312"
188 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
189 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
190 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
191 "j\324\307\\\247\237K\204\177@\253\243M\324\310^\327\312]\327\312]\327\312"
192 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
193 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
194 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
195 "j\216\210D><(SP1CA+\223\214E\327\312]\327\312]\327\312]\327\312]\327\312"
196 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
197 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
198 "j\327\312]\327\312]\327\312]\327\312]\327\312]\324\310^\"! \36\37\37\37\""
199 "\"\40\\""(' $\323\307]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
200 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
201 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
202 "\312]\327\312]\327\312]\327\312]uq;\36\36\36\40\40\40!!!##%%%ea9\327\312"
203 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
204 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
205 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\315\304"
206 "[\36\36\36\40\40\40\\""##$$$$&&' '\301\266W\327\312]\327\312]\327\312"
207 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
208 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
209 "j\327\312]\327\312]\327\312]\327\312]Y3\40\40\40\\""##$$$$&&' ' '))A@"
210 "1\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
211 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312"
212 "j\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\300\265V\35\35\37"
213 "!!!##$$$$&&' ' '))*****\234\226L\327\312]\327\312]\327\312]\327\312]\327"
214 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
215 "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327"
216 "\312]\305\274WZW4\"!#\ "$$$$&%&' (( )) (( *MK5\254\242P\327\312]\327\312]\327"
```

-5-

```

271     "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\"
272     "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\"
273     "\312]\327\312]\327\312]\327\312]\327\312]\327\312]\327\312]\",
274 };
275
276 #define win_width 32
277 #define win_height 32
278 GLubyte win_logo[128] = {
279     0x00, 0x00, 0x00, 0x00,
280     0x00, 0x00, 0x00, 0x00,
281     0x00, 0x00, 0x00, 0x00,
282     0x00, 0x00, 0xfc, 0x00,
283     0xf0, 0x3f, 0xf3, 0x00,
284     0xff, 0xff, 0x03, 0xc0,
285     0xff, 0xfc, 0x00, 0xc0,
286     0xcf, 0xcc, 0x00, 0xc0,
287     0xc0, 0x0c, 0x00, 0xc0,
288     0xc0, 0x0c, 0x00, 0xc0,
289     0xc0, 0x0c, 0x00, 0xc0,
290     0xc0, 0x0c, 0x00, 0xc0,
291     0xc0, 0x0c, 0x00, 0xc0,
292     0xc0, 0x0c, 0x00, 0xc0,
293     0xc0, 0x0c, 0x00, 0xc0,
294     0xc0, 0x0c, 0xfc, 0xc0,
295     0xf0, 0x3c, 0xff, 0xc0,
296     0xff, 0xff, 0xff, 0xc0,
297     0xff, 0xff, 0x03, 0xc0,
298     0xff, 0xfc, 0x00, 0xc0,
299     0xcf, 0xcc, 0x00, 0xc0,
300     0xc0, 0x0c, 0x00, 0xc0,
301     0xc0, 0x0c, 0x00, 0xc0,
302     0xc0, 0x0c, 0x00, 0xc0,
303     0xc0, 0x0c, 0x00, 0xc0,
304     0xc0, 0x0c, 0x00, 0xc0,
305     0xc0, 0x0c, 0x00, 0xc0,
306     0xc0, 0x0c, 0x00, 0xc0,
307     0xc0, 0x0c, 0xfc, 0xc0,
308     0xf0, 0x3c, 0xff, 0xc0,
309     0xff, 0xff, 0xff, 0xc0,
310     0x1f, 0xff, 0x03, 0xc0
311 };
312
313
314
315
316 void drawPoints()
317 {
318     // Following section draws a polygon pattern as background wall paper
319     glEnable(GL_POLYGON_STIPPLE); //Enables polygon stipple
320     glPolygonStipple(WallPaperPattern); //Loads custom pattern
321     glBegin(GL_POLYGON);
322     glColor3f(0.5, .6, 0); //Creates polygon from vertices
323     glVertex2i(-300, 300); //and changes color for each point.
324     glColor3f(0.5, 0.5, .1);

```

```
325     glVertex2i(-300, -300);
326     glColor3f(0.5, .6, 0);
327     glVertex2i(300, -300);
328     glColor3f(0.5, 0.5, .1);
329     glVertex2i(300, 300);
330     glEnd();
331     glDisable(GL_POLYGON_STIPPLE);
332
333     // Following draws shadow beneath desk
334     glPolygonMode(GL_CCW, GL_FILL); //Changes mode to fill
335     glEnable(GL_BLEND); //Enables alpha blending
336     glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
337     glBegin(GL_POLYGON);
338     glColor4f(0, 0, 0, .75);
339     glVertex2i(-300, -175);
340     glVertex2i(300, -175);
341     glVertex2i(300, -300);
342     glVertex2i(-300, -300);
343     glEnd();
344     glDisable(GL_BLEND); //Disables Alpha blending
345
346     // Following section draws a rectangle for the desk surface
347
348     glBegin(GL_POLYGON); //Creates polygon from vertices
349     glColor3f(.5, .3, .1); //and changes color for each point.
350     glVertex2i(-300, -50);
351     glVertex2i(300, -50);
352     glVertex2i(300, -150);
353     glVertex2i(-300, -150);
354     glEnd();
355
356     // Following draws dark area of desk front
357     glBegin(GL_POLYGON);
358     glColor3f(.4, .2, .05);
359     glVertex2i(-300, -150);
360     glVertex2i(300, -150);
361     glVertex2i(300, -175);
362     glVertex2i(-300, -175);
363     glEnd();
364
365     // Following draws line on desk for psuedo-perspective
366     int CurrentLineStartX = 600;
367     int TotalSurfaceLines = 12;
368     glLineWidth(4);
369     for (int i = 0; i < TotalSurfaceLines; i++){
370         glBegin(GL_LINES);
371         glColor3f(.45, .25, 0.08);
372         glVertex2i(CurrentLineStartX, -50);
373         glVertex2i(CurrentLineStartX - 400, -150);
374         glEnd();
375         CurrentLineStartX -= 100;
376     }
377
378     glBegin(GL_LINES);
```

```
379     glVertex2i(-300, -50);
380     glVertex2i(300, -50);
381     glEnd();
382     glBegin(GL_LINES);
383     glVertex2i(-300, -150);
384     glVertex2i(300, -150);
385     glEnd();
386
387     // Draws a multicolor bitmap
388     glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
389
390     glRasterPos2i(100, 0);
391     glPixelZoom(5, 5);
392     glDrawPixels(32, 48, GL_RGB, GL_UNSIGNED_BYTE, gimp_image.pixel_data);
393     glPixelZoom(1, 1);
394
395     //Following draws computer monitor w/o screen
396     glBegin(GL_POLYGON);
397     glColor3f(0.6, 0.6, 0.5);
398     glVertex2i(150, -100);
399     glVertex2i(150, 100);
400     glVertex2i(200, 100);
401     glVertex2i(200, -75);
402     glEnd();
403
404     glBegin(GL_POLYGON);
405     glColor3f(0.8, 0.8, 0.7);
406     glVertex2i(-50, -100);
407     glVertex2i(150, -100);
408     glVertex2i(150, 100);
409     glVertex2i(-50, 100);
410     glEnd();
411
412     //Following draws buttons onto the computer monitor bezel
413     float radius = 1;
414     float x_offset = 130;
415     float y_offset = -62;
416     glBegin(GL_POINTS);
417     glColor3f(0.85, 0.55, 0.55);
418     for (int x=0; x<360; x++){
419         float radians = x*(M_PI/180);
420         glVertex2f(cos(radians)*radius+x_offset, sin(radians)*radius+y_offset);
421     }
422     glEnd();
423
424
425     radius = 4.5;
426     x_offset = -31;
427     y_offset = -62;
428     glBegin(GL_POINTS);
429     glColor3f(0.6, 0.6, 0.6);
430     for (int x=0; x<360; x++){
431         float radians = x*(M_PI/180);
432         glVertex2f(cos(radians)*radius+x_offset, sin(radians)*radius+y_offset);
```



```
433     }
434     glEnd();
435
436     radius = 4;
437     x_offset = -32;
438     y_offset = -62;
439     glBegin(GL_POINTS);
440     glColor3f(0.65, 0.65, 0.65);
441     for (int x=0; x<360; x++){
442         float radians = x*(M_PI/180);
443         glVertex2f(cos(radians)*radius+x_offset, sin(radians)*radius+y_offset);
444     }
445     glEnd();
446
447
448
449     //Following draws screen onto computer monitor (black)
450     glBegin(GL_POLYGON);
451     glColor3f(0, 0, 0);
452     glVertex2i(-25, -50);
453     glVertex2i(125, -50);
454     glVertex2i(125, 75);
455     glVertex2i(-25, 75);
456     glEnd();
457
458     //Adds detail lines to computer monitor
459     glBegin(GL_LINES);
460     glColor3f(0.6, 0.6, 0.5);
461     glVertex2i(-50, -50);
462     glVertex2i(150, -50);
463     glEnd();
464
465     glBegin(GL_LINES);
466     glVertex2i(125, 75);
467     glVertex2i(149, 99);
468     glEnd();
469
470     glBegin(GL_LINES);
471     glVertex2i(-25, 75);
472     glVertex2i(-49, 99);
473     glEnd();
474
475     // Add text to the computer monitor
476     glColor3f(0.0, 0.75, 0.0);
477     glRasterPos2i(-10, 57);
478     std::string s = "blargh";
479     void * font = GLUT_BITMAP_8_BY_13;
480     for (std::string::iterator i = s.begin(); i != s.end(); ++i)
481     {
482         char c = *i;
483         glutBitmapCharacter(font, c);
484     }
485
486
```

```
487 // Draw an artistic version of the windows logo onto the computer monitor
488 glPixelStorei(GL_UNPACK_ALIGNMENT, 4);
489 glRasterPos2i(0, 0);
490 glBitmap( win_width, win_height, -20, 20, 0, 0, win_logo);
491
492
493 //This section begins the file saving
494 glPixelStoref(GL_UNPACK_ALIGNMENT, 8);
495 glReadPixels(0, 0, 512, 512, GL_RGB, GL_FLOAT, PixelsRead);
496
497 // Removes any previous version of savedImg.txt
498 remove("C:\\TEMP\\savedImg.txt");
499
500 ofstream ResultFile("C:\\TEMP\\savedImg.txt");
501
502 for (int i = 0; i < 786432; i++)
503 {
504     ResultFile << PixelsRead[i] << " ";
505 }
506
507 // Closes the file so it is no longer streaming
508 ResultFile.close();
509
510 cout << "File has been saved in C:\\TEMP\\savedImg.txt";
511 }
512
513
514 //*****
515 void myInit()
516 {
517     glClearColor(0, .3, .4, 0); // specify a background color: blueish-green
518     gluOrtho2D(-300, 300, -300, 300); // specify a viewing area
519 }
520
521 //*****
522 void myDisplayCallback()
523 {
524     glClear(GL_COLOR_BUFFER_BIT); // draw the background
525
526     drawPoints();
527
528     glFlush(); // flush out the buffer contents
529 }
530
531 //*****
532 void main(int argc, char ** argv)
533 {
534     glutInit(& argc, argv);
535
536     glutInitWindowSize(512, 512); // specify a window size
537     glutInitWindowPosition(100, 0); // specify a window position
538     glutCreateWindow("Simple Point Drawing"); // create a titled window
539
540     myInit(); // setting up
```

```
541
542     glutDisplayFunc(myDisplayCallback);    // register a callback
543
544     glutMainLoop();                        // get into an infinite loop
545 }
546
```