```cpp
#include "shapes.h"
#include <iostream>
#include <sstream>

#include <GL/glew.h>
#include <GLFW/glfw3.h>
#include <glm/glm.hpp>

Shapes::Shapes() {

};

Shapes::~Shapes() {

}

void Shapes::LoadObj() {

    std::vector< glm::vec3 > obj_vertices;
    std::vector< unsigned int > vertexIndices;
    istringstream rawDataStream(rawData);
    string dataLine;  int linesDone = 0;

    while (std::getline(rawDataStream, dataLine)) {
        if (dataLine.find("v ") != string::npos) {  // does this line have a vector?
            glm::vec3 vertex;

            int foundStart = dataLine.find(" ");  int foundEnd = dataLine.find(" ", foundStart + 1);
            vertex.x = stof(dataLine.substr(foundStart, foundEnd - foundStart));

            foundStart = foundEnd; foundEnd = dataLine.find(" ", foundStart + 1);
            vertex.y = stof(dataLine.substr(foundStart, foundEnd - foundStart));

            foundStart = foundEnd; foundEnd = dataLine.find(" ", foundStart + 1);
            vertex.z = stof(dataLine.substr(foundStart, foundEnd - foundStart));

            obj_vertices.push_back(vertex);
        }
        else if (dataLine.find("f ") != string::npos) { // does this line defines a triangle face?
            string parts[3];

            int foundStart = dataLine.find(" ");  int foundEnd = dataLine.find(" ", foundStart + 1);
            parts[0] = dataLine.substr(foundStart + 1, foundEnd - foundStart - 1);
```

```
44
45              foundStart = foundEnd; foundEnd = dataLine.find( " ", foundStart
                  + 1);
46              parts[1] = dataLine.substr(foundStart + 1, foundEnd - foundStart
                   - 1);
47
48              foundStart = foundEnd; foundEnd = dataLine.find( " ", foundStart
                  + 1);
49              parts[2] = dataLine.substr(foundStart + 1, foundEnd - foundStart
                   - 1);
50
51              for (int i = 0; i < 3; i++) {       // for each part
52
53                  vertexIndices.push_back(stoul(parts[i].substr(0, parts
                      [i].find("/"))));
54
55                  int firstSlash = parts[i].find("/"); int secondSlash = parts
                      [i].find("/", firstSlash + 1);
56
57                  if (firstSlash != (secondSlash + 1)) {  // there is texture
                      coordinates.
58                                                          // add code for my
                         texture coordintes here.
59                  }
60              }
61          }
62
63          linesDone++;
64      }
65
66      for (unsigned int i = 0; i < vertexIndices.size(); i += 3) {
67          vertexPositions.push_back(obj_vertices[vertexIndices[i + 0] - 1].x);
68          vertexPositions.push_back(obj_vertices[vertexIndices[i + 0] - 1].y);
69          vertexPositions.push_back(obj_vertices[vertexIndices[i + 0] - 1].z);
70
71          vertexPositions.push_back(obj_vertices[vertexIndices[i + 1] - 1].x);
72          vertexPositions.push_back(obj_vertices[vertexIndices[i + 1] - 1].y);
73          vertexPositions.push_back(obj_vertices[vertexIndices[i + 1] - 1].z);
74
75          vertexPositions.push_back(obj_vertices[vertexIndices[i + 2] - 1].x);
76          vertexPositions.push_back(obj_vertices[vertexIndices[i + 2] - 1].y);
77          vertexPositions.push_back(obj_vertices[vertexIndices[i + 2] - 1].z);
78      }
79  }
80
81
82  void Shapes::Load() {
83      static const char * vs_source[] = { R"(
84  #version 330 core
85
86  in vec4 position;
87  uniform mat4 mv_matrix;
88  uniform mat4 proj_matrix;
```

```
 89
 90 void main(void){
 91     gl_Position = proj_matrix * mv_matrix * position;
 92 }
 93 )" };
 94
 95     static const char * fs_source[] = { R"(
 96 #version 330 core
 97
 98 uniform vec4 inColor;
 99 out vec4 color;
100
101 void main(void){
102     color = inColor;
103 }
104 )" };
105
106     program = glCreateProgram();
107     GLuint fs = glCreateShader(GL_FRAGMENT_SHADER);
108     glShaderSource(fs, 1, fs_source, NULL);
109     glCompileShader(fs);
110     checkErrorShader(fs);
111
112     GLuint vs = glCreateShader(GL_VERTEX_SHADER);
113     glShaderSource(vs, 1, vs_source, NULL);
114     glCompileShader(vs);
115     checkErrorShader(vs);
116
117     glAttachShader(program, vs);
118     glAttachShader(program, fs);
119
120     glLinkProgram(program);
121
122     mv_location = glGetUniformLocation(program, "mv_matrix");
123     proj_location = glGetUniformLocation(program, "proj_matrix");
124     color_location = glGetUniformLocation(program, "inColor");
125
126     glGenVertexArrays(1, &vao);
127     glBindVertexArray(vao);
128
129     glGenBuffers(1, &buffer);
130     glBindBuffer(GL_ARRAY_BUFFER, buffer);
131     glBufferData(GL_ARRAY_BUFFER,
132         vertexPositions.size() * sizeof(GLfloat),
133         &vertexPositions[0],
134         GL_STATIC_DRAW);
135     glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 0, NULL);
136     glEnableVertexAttribArray(0);
137
138     glLinkProgram(0);   // unlink
139     glDisableVertexAttribArray(0); // Disable
140     glBindVertexArray(0);   // Unbind
141 }
```

```
142
143 void Shapes::Draw() {
144     glUseProgram(program);
145     glBindVertexArray(vao);
146     glEnableVertexAttribArray(0);
147
148     glUniformMatrix4fv(proj_location, 1, GL_FALSE, &proj_matrix[0][0]);
149     glUniformMatrix4fv(mv_location, 1, GL_FALSE, &mv_matrix[0][0]);
150
151     glUniform4f(color_location, fillColor.r, fillColor.g, fillColor.b,
        fillColor.a);
152     glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
153     glDrawArrays(GL_TRIANGLES, 0, vertexPositions.size() / 3);
154
155     glUniform4f(color_location, lineColor.r, lineColor.g, lineColor.b,
        lineColor.a);
156     glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);  glLineWidth(lineWidth);
157     glDrawArrays(GL_TRIANGLES, 0, vertexPositions.size() / 3);
158 }
159
160
161 void Shapes::checkErrorShader(GLuint shader) {
162     // Get log length
163     GLint maxLength;
164     glGetShaderiv(shader, GL_INFO_LOG_LENGTH, &maxLength);
165
166     // Init a string for it
167     std::vector<GLchar> errorLog(maxLength);
168
169     if (maxLength > 1) {
170         // Get the log file
171         glGetShaderInfoLog(shader, maxLength, &maxLength, &errorLog[0]);
172
173         cout << "--------------Shader compilation error-------------\n";
174         cout << errorLog.data();
175     }
176 }
177
178 Particle::Particle() {
179     // Exported from Blender a cube by default (OBJ File)
180     rawData = R"(
181 o Cone
182 v 0.000000 -1.000000 -1.000000
183 v 0.000000 1.000000 0.000000
184 v 0.195090 -1.000000 -0.980785
185 v 0.382683 -1.000000 -0.923880
186 v 0.555570 -1.000000 -0.831470
187 v 0.707107 -1.000000 -0.707107
188 v 0.831470 -1.000000 -0.555570
189 v 0.923880 -1.000000 -0.382683
190 v 0.980785 -1.000000 -0.195090
191 v 1.000000 -1.000000 -0.000000
192 v 0.980785 -1.000000 0.195090
```

```
193  v 0.923880 -1.000000 0.382683
194  v 0.831470 -1.000000 0.555570
195  v 0.707107 -1.000000 0.707107
196  v 0.555570 -1.000000 0.831470
197  v 0.382683 -1.000000 0.923880
198  v 0.195090 -1.000000 0.980785
199  v -0.000000 -1.000000 1.000000
200  v -0.195091 -1.000000 0.980785
201  v -0.382684 -1.000000 0.923879
202  v -0.555571 -1.000000 0.831469
203  v -0.707107 -1.000000 0.707106
204  v -0.831470 -1.000000 0.555570
205  v -0.923880 -1.000000 0.382683
206  v -0.980785 -1.000000 0.195089
207  v -1.000000 -1.000000 -0.000001
208  v -0.980785 -1.000000 -0.195091
209  v -0.923879 -1.000000 -0.382684
210  v -0.831469 -1.000000 -0.555571
211  v -0.707106 -1.000000 -0.707108
212  v -0.555569 -1.000000 -0.831470
213  v -0.382682 -1.000000 -0.923880
214  v -0.195089 -1.000000 -0.980786
215  vn 0.0878 0.4455 -0.8910
216  vn 0.2599 0.4455 -0.8567
217  vn 0.4220 0.4455 -0.7896
218  vn 0.5680 0.4455 -0.6921
219  vn 0.6921 0.4455 -0.5680
220  vn 0.7896 0.4455 -0.4220
221  vn 0.8567 0.4455 -0.2599
222  vn 0.8910 0.4455 -0.0878
223  vn 0.8910 0.4455 0.0878
224  vn 0.8567 0.4455 0.2599
225  vn 0.7896 0.4455 0.4220
226  vn 0.6921 0.4455 0.5680
227  vn 0.5680 0.4455 0.6921
228  vn 0.4220 0.4455 0.7896
229  vn 0.2599 0.4455 0.8567
230  vn 0.0878 0.4455 0.8910
231  vn -0.0878 0.4455 0.8910
232  vn -0.2599 0.4455 0.8567
233  vn -0.4220 0.4455 0.7896
234  vn -0.5680 0.4455 0.6921
235  vn -0.6921 0.4455 0.5680
236  vn -0.7896 0.4455 0.4220
237  vn -0.8567 0.4455 0.2599
238  vn -0.8910 0.4455 0.0878
239  vn -0.8910 0.4455 -0.0878
240  vn -0.8567 0.4455 -0.2599
241  vn -0.7896 0.4455 -0.4220
242  vn -0.6921 0.4455 -0.5680
243  vn -0.5680 0.4455 -0.6921
244  vn -0.4220 0.4455 -0.7896
245  vn -0.2599 0.4455 -0.8567
```

```
246 vn -0.0878 0.4455 -0.8910
247 vn 0.0000 -1.0000 0.0000
248 usemtl None
249 s off
250 f 1//1 2//1 3//1
251 f 3//2 2//2 4//2
252 f 4//3 2//3 5//3
253 f 5//4 2//4 6//4
254 f 6//5 2//5 7//5
255 f 7//6 2//6 8//6
256 f 8//7 2//7 9//7
257 f 9//8 2//8 10//8
258 f 10//9 2//9 11//9
259 f 11//10 2//10 12//10
260 f 12//11 2//11 13//11
261 f 13//12 2//12 14//12
262 f 14//13 2//13 15//13
263 f 15//14 2//14 16//14
264 f 16//15 2//15 17//15
265 f 17//16 2//16 18//16
266 f 18//17 2//17 19//17
267 f 19//18 2//18 20//18
268 f 20//19 2//19 21//19
269 f 21//20 2//20 22//20
270 f 22//21 2//21 23//21
271 f 23//22 2//22 24//22
272 f 24//23 2//23 25//23
273 f 25//24 2//24 26//24
274 f 26//25 2//25 27//25
275 f 27//26 2//26 28//26
276 f 28//27 2//27 29//27
277 f 29//28 2//28 30//28
278 f 30//29 2//29 31//29
279 f 31//30 2//30 32//30
280 f 32//31 2//31 33//31
281 f 33//32 2//32 1//32
282 f 17//33 25//33 9//33
283 f 33//33 1//33 3//33
284 f 3//33 4//33 5//33
285 f 5//33 6//33 7//33
286 f 7//33 8//33 5//33
287 f 9//33 10//33 11//33
288 f 11//33 12//33 9//33
289 f 13//33 14//33 17//33
290 f 15//33 16//33 17//33
291 f 17//33 18//33 19//33
292 f 19//33 20//33 21//33
293 f 21//33 22//33 23//33
294 f 23//33 24//33 25//33
295 f 25//33 26//33 27//33
296 f 27//33 28//33 29//33
297 f 29//33 30//33 33//33
298 f 31//33 32//33 33//33
```

```
299 f 33//33 3//33 9//33
300 f 5//33 8//33 9//33
301 f 9//33 12//33 13//33
302 f 14//33 15//33 17//33
303 f 17//33 19//33 25//33
304 f 21//33 23//33 25//33
305 f 25//33 27//33 33//33
306 f 30//33 31//33 33//33
307 f 3//33 5//33 9//33
308 f 9//33 13//33 17//33
309 f 19//33 21//33 25//33
310 f 27//33 29//33 33//33
311 f 33//33 9//33 25//33
312 )";
313
314     LoadObj();
315 }
316
317 Particle::~Particle() {
318
319 }
320
321 Cube::Cube() {
322     // Exported from Blender a cube by default (OBJ File)
323     rawData = R"(
324 o Cone
325 v 0.000000 -1.000000 -1.000000
326 v 0.000000 1.000000 0.000000
327 v 0.195090 -1.000000 -0.980785
328 v 0.382683 -1.000000 -0.923880
329 v 0.555570 -1.000000 -0.831470
330 v 0.707107 -1.000000 -0.707107
331 v 0.831470 -1.000000 -0.555570
332 v 0.923880 -1.000000 -0.382683
333 v 0.980785 -1.000000 -0.195090
334 v 1.000000 -1.000000 -0.000000
335 v 0.980785 -1.000000 0.195090
336 v 0.923880 -1.000000 0.382683
337 v 0.831470 -1.000000 0.555570
338 v 0.707107 -1.000000 0.707107
339 v 0.555570 -1.000000 0.831470
340 v 0.382683 -1.000000 0.923880
341 v 0.195090 -1.000000 0.980785
342 v -0.000000 -1.000000 1.000000
343 v -0.195091 -1.000000 0.980785
344 v -0.382684 -1.000000 0.923879
345 v -0.555571 -1.000000 0.831469
346 v -0.707107 -1.000000 0.707106
347 v -0.831470 -1.000000 0.555570
348 v -0.923880 -1.000000 0.382683
349 v -0.980785 -1.000000 0.195089
350 v -1.000000 -1.000000 -0.000001
351 v -0.980785 -1.000000 -0.195091
```

```
v -0.923879 -1.000000 -0.382684
v -0.831469 -1.000000 -0.555571
v -0.707106 -1.000000 -0.707108
v -0.555569 -1.000000 -0.831470
v -0.382682 -1.000000 -0.923880
v -0.195089 -1.000000 -0.980786
vn 0.0878 0.4455 -0.8910
vn 0.2599 0.4455 -0.8567
vn 0.4220 0.4455 -0.7896
vn 0.5680 0.4455 -0.6921
vn 0.6921 0.4455 -0.5680
vn 0.7896 0.4455 -0.4220
vn 0.8567 0.4455 -0.2599
vn 0.8910 0.4455 -0.0878
vn 0.8910 0.4455 0.0878
vn 0.8567 0.4455 0.2599
vn 0.7896 0.4455 0.4220
vn 0.6921 0.4455 0.5680
vn 0.5680 0.4455 0.6921
vn 0.4220 0.4455 0.7896
vn 0.2599 0.4455 0.8567
vn 0.0878 0.4455 0.8910
vn -0.0878 0.4455 0.8910
vn -0.2599 0.4455 0.8567
vn -0.4220 0.4455 0.7896
vn -0.5680 0.4455 0.6921
vn -0.6921 0.4455 0.5680
vn -0.7896 0.4455 0.4220
vn -0.8567 0.4455 0.2599
vn -0.8910 0.4455 0.0878
vn -0.8910 0.4455 -0.0878
vn -0.8567 0.4455 -0.2599
vn -0.7896 0.4455 -0.4220
vn -0.6921 0.4455 -0.5680
vn -0.5680 0.4455 -0.6921
vn -0.4220 0.4455 -0.7896
vn -0.2599 0.4455 -0.8567
vn -0.0878 0.4455 -0.8910
vn 0.0000 -1.0000 0.0000
usemtl None
s off
f 1//1 2//1 3//1
f 3//2 2//2 4//2
f 4//3 2//3 5//3
f 5//4 2//4 6//4
f 6//5 2//5 7//5
f 7//6 2//6 8//6
f 8//7 2//7 9//7
f 9//8 2//8 10//8
f 10//9 2//9 11//9
f 11//10 2//10 12//10
f 12//11 2//11 13//11
f 13//12 2//12 14//12
```

```
405  f 14//13 2//13 15//13
406  f 15//14 2//14 16//14
407  f 16//15 2//15 17//15
408  f 17//16 2//16 18//16
409  f 18//17 2//17 19//17
410  f 19//18 2//18 20//18
411  f 20//19 2//19 21//19
412  f 21//20 2//20 22//20
413  f 22//21 2//21 23//21
414  f 23//22 2//22 24//22
415  f 24//23 2//23 25//23
416  f 25//24 2//24 26//24
417  f 26//25 2//25 27//25
418  f 27//26 2//26 28//26
419  f 28//27 2//27 29//27
420  f 29//28 2//28 30//28
421  f 30//29 2//29 31//29
422  f 31//30 2//30 32//30
423  f 32//31 2//31 33//31
424  f 33//32 2//32 1//32
425  f 17//33 25//33 9//33
426  f 33//33 1//33 3//33
427  f 3//33 4//33 5//33
428  f 5//33 6//33 7//33
429  f 7//33 8//33 5//33
430  f 9//33 10//33 11//33
431  f 11//33 12//33 9//33
432  f 13//33 14//33 17//33
433  f 15//33 16//33 17//33
434  f 17//33 18//33 19//33
435  f 19//33 20//33 21//33
436  f 21//33 22//33 23//33
437  f 23//33 24//33 25//33
438  f 25//33 26//33 27//33
439  f 27//33 28//33 29//33
440  f 29//33 30//33 33//33
441  f 31//33 32//33 33//33
442  f 33//33 3//33 9//33
443  f 5//33 8//33 9//33
444  f 9//33 12//33 13//33
445  f 14//33 15//33 17//33
446  f 17//33 19//33 25//33
447  f 21//33 23//33 25//33
448  f 25//33 27//33 33//33
449  f 30//33 31//33 33//33
450  f 3//33 5//33 9//33
451  f 9//33 13//33 17//33
452  f 19//33 21//33 25//33
453  f 27//33 29//33 33//33
454  f 33//33 9//33 25//33
455  )";
456
457      LoadObj();
```

```
458 }
459
460 Cube::~Cube() {
461
462 }
463
464 Sphere::Sphere() {
465
466     rawData = R"(
467 o Sphere
468 v -0.097545 0.490393 0.000000
469 v -0.277785 0.415735 0.000000
470 v -0.415735 0.277785 0.000000
471 v -0.490393 0.097545 0.000000
472 v -0.490393 -0.097545 0.000000
473 v -0.415735 -0.277785 0.000000
474 v -0.277785 -0.415735 0.000000
475 v -0.097545 -0.490393 0.000000
476 v -0.090120 0.490393 -0.037329
477 v -0.256640 0.415735 -0.106304
478 v -0.384089 0.277785 -0.159095
479 v -0.453064 0.097545 -0.187665
480 v -0.453064 -0.097545 -0.187665
481 v -0.384089 -0.277785 -0.159095
482 v -0.256640 -0.415735 -0.106304
483 v -0.090120 -0.490393 -0.037329
484 v -0.068975 0.490393 -0.068975
485 v -0.196424 0.415735 -0.196424
486 v -0.293969 0.277785 -0.293969
487 v -0.346760 0.097545 -0.346760
488 v -0.346760 -0.097545 -0.346760
489 v -0.293969 -0.277785 -0.293969
490 v -0.196424 -0.415735 -0.196424
491 v -0.068975 -0.490393 -0.068975
492 v -0.037329 0.490393 -0.090120
493 v -0.106304 0.415735 -0.256640
494 v -0.159095 0.277785 -0.384089
495 v -0.187665 0.097545 -0.453064
496 v -0.187665 -0.097545 -0.453064
497 v -0.159095 -0.277785 -0.384089
498 v -0.106304 -0.415735 -0.256640
499 v -0.037329 -0.490393 -0.090120
500 v 0.000000 0.490393 -0.097545
501 v 0.000000 0.415735 -0.277785
502 v 0.000000 0.277785 -0.415735
503 v 0.000000 0.097545 -0.490393
504 v 0.000000 -0.097545 -0.490393
505 v 0.000000 -0.277785 -0.415735
506 v 0.000000 -0.415735 -0.277785
507 v 0.000000 -0.490393 -0.097545
508 v 0.037329 0.490393 -0.090120
509 v 0.106304 0.415735 -0.256640
510 v 0.159095 0.277785 -0.384089
```

```
511  v 0.187665 0.097545 -0.453064
512  v 0.187665 -0.097545 -0.453064
513  v 0.159095 -0.277785 -0.384089
514  v 0.106304 -0.415735 -0.256640
515  v 0.037329 -0.490393 -0.090120
516  v 0.068975 0.490393 -0.068975
517  v 0.196424 0.415735 -0.196424
518  v 0.293969 0.277785 -0.293969
519  v 0.346760 0.097545 -0.346760
520  v 0.346760 -0.097545 -0.346760
521  v 0.293969 -0.277785 -0.293969
522  v 0.196424 -0.415735 -0.196424
523  v 0.068975 -0.490393 -0.068975
524  v 0.090120 0.490393 -0.037329
525  v 0.256640 0.415735 -0.106304
526  v 0.384089 0.277785 -0.159095
527  v 0.453064 0.097545 -0.187665
528  v 0.453064 -0.097545 -0.187665
529  v 0.384089 -0.277785 -0.159095
530  v 0.256640 -0.415735 -0.106304
531  v 0.090120 -0.490393 -0.037329
532  v 0.097545 0.490393 0.000000
533  v 0.277785 0.415735 -0.000000
534  v 0.415735 0.277785 0.000000
535  v 0.490393 0.097545 0.000000
536  v 0.490393 -0.097545 0.000000
537  v 0.415735 -0.277785 0.000000
538  v 0.277785 -0.415735 0.000000
539  v 0.097545 -0.490393 -0.000000
540  v 0.090120 0.490393 0.037329
541  v 0.256640 0.415735 0.106304
542  v 0.384089 0.277785 0.159095
543  v 0.453064 0.097545 0.187665
544  v 0.453064 -0.097545 0.187665
545  v 0.384089 -0.277785 0.159095
546  v 0.256640 -0.415735 0.106304
547  v 0.090120 -0.490393 0.037329
548  v 0.068975 0.490393 0.068975
549  v 0.196424 0.415735 0.196424
550  v 0.293969 0.277785 0.293969
551  v 0.346760 0.097545 0.346760
552  v 0.346760 -0.097545 0.346760
553  v 0.293969 -0.277785 0.293969
554  v 0.196424 -0.415735 0.196424
555  v 0.068975 -0.490393 0.068975
556  v 0.000000 -0.500000 0.000000
557  v 0.037329 0.490393 0.090120
558  v 0.106304 0.415735 0.256640
559  v 0.159095 0.277785 0.384089
560  v 0.187665 0.097545 0.453064
561  v 0.187665 -0.097545 0.453064
562  v 0.159095 -0.277785 0.384089
563  v 0.106304 -0.415735 0.256640
```

```
564 v 0.037329 -0.490393 0.090120
565 v 0.000000 0.490393 0.097545
566 v 0.000000 0.415735 0.277785
567 v 0.000000 0.277785 0.415735
568 v 0.000000 0.097545 0.490392
569 v 0.000000 -0.097545 0.490392
570 v 0.000000 -0.277785 0.415735
571 v 0.000000 -0.415735 0.277785
572 v 0.000000 -0.490393 0.097545
573 v -0.037329 0.490393 0.090120
574 v -0.106304 0.415735 0.256640
575 v -0.159095 0.277785 0.384089
576 v -0.187665 0.097545 0.453063
577 v -0.187665 -0.097545 0.453063
578 v -0.159095 -0.277785 0.384089
579 v -0.106304 -0.415735 0.256640
580 v -0.037329 -0.490393 0.090120
581 v -0.068975 0.490393 0.068975
582 v -0.196424 0.415735 0.196424
583 v -0.293969 0.277785 0.293969
584 v -0.346760 0.097545 0.346760
585 v -0.346760 -0.097545 0.346760
586 v -0.293969 -0.277785 0.293969
587 v -0.196423 -0.415735 0.196424
588 v -0.068975 -0.490393 0.068975
589 v 0.000000 0.500000 0.000000
590 v -0.090120 0.490393 0.037329
591 v -0.256640 0.415735 0.106304
592 v -0.384088 0.277785 0.159095
593 v -0.453063 0.097545 0.187665
594 v -0.453063 -0.097545 0.187665
595 v -0.384088 -0.277785 0.159095
596 v -0.256640 -0.415735 0.106304
597 v -0.090120 -0.490393 0.037329
598 s off
599 f 7 14 15
600 f 3 10 11
601 f 12 3 11
602 f 8 15 16
603 f 5 12 13
604 f 2 125 124
605 f 2 9 10
606 f 6 13 14
607 f 89 8 16
608 f 122 17 9
609 f 7 128 6
610 f 20 27 28
611 f 8 129 7
612 f 22 29 30
613 f 19 26 27
614 f 29 36 37
615 f 31 22 30
616 f 89 16 24
```

```
617  f 26 33 34
618  f 24 31 32
619  f 28 35 36
620  f 122 25 17
621  f 27 34 35
622  f 37 44 45
623  f 38 29 37
624  f 89 24 32
625  f 42 33 41
626  f 32 39 40
627  f 36 43 44
628  f 31 38 39
629  f 122 33 25
630  f 43 34 42
631  f 45 52 53
632  f 46 37 45
633  f 89 32 40
634  f 43 50 51
635  f 48 39 47
636  f 52 43 51
637  f 39 46 47
638  f 50 41 49
639  f 122 41 33
640  f 53 60 61
641  f 47 54 55
642  f 46 53 54
643  f 48 55 56
644  f 60 51 59
645  f 58 49 57
646  f 122 49 41
647  f 89 40 48
648  f 61 68 69
649  f 55 62 63
650  f 54 61 62
651  f 51 58 59
652  f 58 65 66
653  f 68 59 67
654  f 122 57 49
655  f 56 63 64
656  f 89 48 56
657  f 63 70 71
658  f 62 69 70
659  f 59 66 67
660  f 69 76 77
661  f 66 73 74
662  f 122 65 57
663  f 64 71 72
664  f 76 67 75
665  f 89 56 64
666  f 79 70 78
667  f 70 77 78
668  f 67 74 75
669  f 77 84 85
```

```
f 72 79 80
f 122 73 65
f 76 83 84
f 89 64 72
f 74 81 82
f 87 78 86
f 86 77 85
f 75 82 83
f 85 93 94
f 80 87 88
f 84 92 93
f 122 81 73
f 89 72 80
f 91 81 90
f 87 95 96
f 86 94 95
f 83 91 92
f 94 101 102
f 93 100 101
f 89 80 88
f 122 90 81
f 91 98 99
f 88 96 97
f 95 102 103
f 92 99 100
f 102 109 110
f 96 103 104
f 122 98 90
f 89 88 97
f 99 106 107
f 105 96 104
f 109 100 108
f 108 99 107
f 110 117 118
f 104 111 112
f 122 106 98
f 89 97 105
f 107 114 115
f 103 110 111
f 117 108 116
f 113 104 112
f 108 115 116
f 118 126 127
f 120 111 119
f 122 114 106
f 115 123 124
f 111 118 119
f 89 105 113
f 113 120 121
f 126 116 125
f 119 127 128
f 116 124 125
f 120 128 129
```

```
f 89 113 121
f 121 129 130
f 122 123 114
f 89 121 130
f 122 1 123
f 89 130 8
f 3 126 125
f 5 126 4
f 15 22 23
f 10 17 18
f 24 15 23
f 13 20 21
f 18 25 26
f 14 21 22
f 21 28 29
f 12 19 20
f 11 18 19
f 1 124 123
f 122 9 1
f 6 127 5
f 7 6 14
f 3 2 10
f 12 4 3
f 8 7 15
f 5 4 12
f 2 3 125
f 2 1 9
f 6 5 13
f 7 129 128
f 20 19 27
f 8 130 129
f 22 21 29
f 19 18 26
f 29 28 36
f 31 23 22
f 26 25 33
f 24 23 31
f 28 27 35
f 27 26 34
f 37 36 44
f 38 30 29
f 42 34 33
f 32 31 39
f 36 35 43
f 31 30 38
f 43 35 34
f 45 44 52
f 46 38 37
f 43 42 50
f 48 40 39
f 52 44 43
f 39 38 46
f 50 42 41
```

```
776  f 53 52 60
777  f 47 46 54
778  f 46 45 53
779  f 48 47 55
780  f 60 52 51
781  f 58 50 49
782  f 61 60 68
783  f 55 54 62
784  f 54 53 61
785  f 51 50 58
786  f 58 57 65
787  f 68 60 59
788  f 56 55 63
789  f 63 62 70
790  f 62 61 69
791  f 59 58 66
792  f 69 68 76
793  f 66 65 73
794  f 64 63 71
795  f 76 68 67
796  f 79 71 70
797  f 70 69 77
798  f 67 66 74
799  f 77 76 84
800  f 72 71 79
801  f 76 75 83
802  f 74 73 81
803  f 87 79 78
804  f 86 78 77
805  f 75 74 82
806  f 85 84 93
807  f 80 79 87
808  f 84 83 92
809  f 91 82 81
810  f 87 86 95
811  f 86 85 94
812  f 83 82 91
813  f 94 93 101
814  f 93 92 100
815  f 91 90 98
816  f 88 87 96
817  f 95 94 102
818  f 92 91 99
819  f 102 101 109
820  f 96 95 103
821  f 99 98 106
822  f 105 97 96
823  f 109 101 100
824  f 108 100 99
825  f 110 109 117
826  f 104 103 111
827  f 107 106 114
828  f 103 102 110
```

```
f 117 109 108
f 113 105 104
f 108 107 115
f 118 117 126
f 120 112 111
f 115 114 123
f 111 110 118
f 113 112 120
f 126 117 116
f 119 118 127
f 116 115 124
f 120 119 128
f 121 120 129
f 3 4 126
f 5 127 126
f 15 14 22
f 10 9 17
f 24 16 15
f 13 12 20
f 18 17 25
f 14 13 21
f 21 20 28
f 12 11 19
f 11 10 18
f 1 2 124
f 6 128 127

)";

    LoadObj();
}

Sphere::~Sphere() {

}

Arrow::Arrow() {

    rawData = R"(
o Cone
v 0.000000 0.800000 -0.100000
v 0.070711 0.800000 -0.070711
v 0.100000 0.800000 -0.000000
v 0.000000 1.000000 0.000000
v 0.070711 0.800000 0.070711
v -0.000000 0.800000 0.100000
v -0.070711 0.800000 0.070711
v -0.100000 0.800000 -0.000000
v -0.070711 0.800000 -0.070711
s off
f 4 7 6
f 5 7 2
f 4 8 7
```

```
882 f 3 4 5
883 f 5 4 6
884 f 4 9 8
885 f 4 1 9
886 f 2 1 4
887 f 2 4 3
888 f 9 1 2
889 f 2 3 5
890 f 5 6 7
891 f 7 8 9
892 f 9 2 7
893 o Cylinder
894 v 0.000000 0.000000 -0.050000
895 v 0.009755 0.900000 -0.049039
896 v 0.019134 0.000000 -0.046194
897 v 0.027779 0.900000 -0.041573
898 v 0.035355 0.000000 -0.035355
899 v 0.041573 0.900000 -0.027779
900 v 0.046194 0.000000 -0.019134
901 v 0.049039 0.900000 -0.009755
902 v 0.050000 0.000000 -0.000000
903 v 0.049039 0.900000 0.009755
904 v 0.046194 0.000000 0.019134
905 v 0.041573 0.900000 0.027779
906 v 0.035355 0.000000 0.035355
907 v 0.027779 0.900000 0.041573
908 v 0.019134 0.000000 0.046194
909 v 0.009755 0.900000 0.049039
910 v -0.000000 0.000000 0.050000
911 v -0.009755 0.900000 0.049039
912 v -0.019134 0.000000 0.046194
913 v -0.027779 0.900000 0.041573
914 v -0.035355 0.000000 0.035355
915 v -0.041574 0.900000 0.027778
916 v -0.046194 0.000000 0.019134
917 v -0.049039 0.900000 0.009754
918 v -0.050000 0.000000 -0.000000
919 v -0.049039 0.900000 -0.009755
920 v -0.046194 0.000000 -0.019134
921 v -0.041573 0.900000 -0.027779
922 v -0.035355 0.000000 -0.035355
923 v -0.027778 0.900000 -0.041574
924 v -0.019134 0.000000 -0.046194
925 v -0.009754 0.900000 -0.049039
926 s off
927 f 13 15 14
928 f 16 14 15
929 f 17 19 18
930 f 18 16 17
931 f 19 21 20
932 f 20 18 19
933 f 21 23 22
934 f 22 20 21
```

```
935  f 23 25 24
936  f 24 22 23
937  f 25 27 26
938  f 26 24 25
939  f 27 29 28
940  f 28 26 27
941  f 29 31 30
942  f 30 28 29
943  f 31 33 32
944  f 32 30 31
945  f 33 35 34
946  f 34 32 33
947  f 35 37 36
948  f 36 34 35
949  f 37 39 38
950  f 38 36 37
951  f 41 40 39
952  f 40 38 39
953  f 41 10 40
954  f 29 21 37
955  f 11 12 10
956  f 24 32 16
957  f 15 17 16
958  f 11 13 12
959  f 14 12 13
960  f 10 41 11
961  f 13 11 41
962  f 41 39 37
963  f 37 35 33
964  f 33 31 29
965  f 29 27 25
966  f 25 23 29
967  f 21 19 17
968  f 17 15 13
969  f 13 41 37
970  f 37 33 29
971  f 29 23 21
972  f 21 17 13
973  f 13 37 21
974  f 40 10 12
975  f 12 14 16
976  f 16 18 20
977  f 20 22 24
978  f 24 26 28
979  f 28 30 32
980  f 32 34 36
981  f 36 38 40
982  f 40 12 16
983  f 16 20 24
984  f 24 28 32
985  f 32 36 40
986  f 40 16 32
987  )";
```

```
988
989     LoadObj();
990 }
991
992 Arrow::~Arrow() {
993
994 }
```