

```

1 // Illustrate the use of static member variables and static member functions
2 // Georgia Tech, ECE2036, Fall 2012
3
4 // We will define a class Point3 that manages a point in 3-D space, maintaining
5 // an x, y, and z coordinate. It also will maintain the "pixel index"
6 // that will index a 1-dimensional array of pixels that represents the
7 // set of points for an image.
8 //
9 // To do this, the conversion from x/y/z to pixel index requires
10 // knowledge of the width and height of the image. Since the width and
11 // height are the same for all instances of Point3 objects, it does not
12 // make sense to use normal member variables for these. What we want are
13 // variables that are part of the Point3 class, but are common for all
14 // instances. This is called a "static member variable", and is defined just
15 // like any other member variable, but is preceded by the "static"
16 // keyword. One other difference is that the static variables
17 // must later be "created", as illustrated in the Point3 class below.
18 //
19 // Since width and height are not in fact member variables for any
20 // specific Point3 instance, it would make sense that we have a way
21 // to set new values for these without requiring an existing object
22 // of class Point3. Recall that for normal member variables
23 // we must have a Point3 object to access a member variable or
24 // call a member function, such as:
25 //
26 // Point3 p(1, 2, 3);
27 // int x = p.GetX();
28 // int y = p.GetY();
29 //
30 // The above snippet calls member functions GetX and GetY for object
31 // "p" returning the x and y values for that specific Point3 object.
32 //
33 // To create functions that can be called WITHOUT an object, we create
34 // "static member functions". As in the static member variables, we
35 // simply precede the member function declaration with the "static"
36 // keyword. We can then call "SetW" (if SetW is static) without
37 // any existing objects of class Point3, as follows:
38 //
39 // Point3::SetW(256);
40 // Point3::SetH(256);
41 //
42 // or, if the W and H variables are public, we can just access them
43 // directly:
44 //
45 // Point3::W = 256;
46 // Point3::H = 256;
47
48 #include <iostream>
49 using namespace std;
50
51 // The following Point3 declaration would normally be put in Point3.h
52 // but is here for simplicity:
53
54 class Point3
55 {
56 public:

```

Program static-members.cc

```

57     Point3(int x0, int y0, int z0);
58     Point3(int x0, int y0);           // Assumes z = 0;
59     Point3(int i0);                 // Initialize with index
60     // Accessor functions
61 public:
62     int GetX()    const;
63     int GetY()    const;
64     int GetZ()    const;
65     int GetInd() const;
66 private:
67     // x, y, z, and ind are member variables, but private.
68     int x;
69     int y;
70     int z;
71     int ind; // Pixel index value
72
73     // Now define the "static" width and height
74 private:
75     static int W; // Width of image
76     static int H; // Height of image
77     // Create the static "setter" functions for W and H
78 public:
79     static void SetW(int w0);
80     static void SetH(int h0);
81 };
82
83 // The following would be in Point3.cc
84 //
85 // Since we have static member variables W and H, we just actually
86 // define those and (optionally) initialize.
87 //
88 int Point3::W = 0;
89 int Point3::H = 0;
90
91 // Point3 constructors
92 Point3::Point3(int x0, int y0, int z0)
93     : x(x0), y(y0), z(z0)
94 {
95     // Compute the pixel index
96     ind = z * W * H + y * W + x;
97 }
98
99 Point3::Point3(int x0, int y0)
100     : x(x0), y(y0), z(0)
101 {
102     // Compute the pixel index
103     ind = z * W * H + y * W + x;
104 }
105
106 Point3::Point3(int i)
107     : ind(i)
108 { // i is the pixel index compute x, y, and z
109     z = ind / (W * H);
110     y = (ind - z * W * H) / W;
111     x = ind % W;
112 }

```

Program static-members.cc (continued)

```

113
114 // Define the accessors
115 int Point3::GetX() const
116 {
117     return x;
118 }
119
120 int Point3::GetY() const
121 {
122     return y;
123 }
124
125 int Point3::GetZ() const
126 {
127     return z;
128 }
129
130 int Point3::GetInd() const
131 {
132     return ind;
133 }
134
135 // Define the static "setters" for W and H
136 void Point3::SetW(int w0)
137 {
138     W = w0;
139 }
140
141 void Point3::SetH(int h0)
142 {
143     H = h0;
144 }
145
146 // Main program for testing
147 int main()
148 {
149     int imageW = 128;
150     int imageH = 128; // Arbitrary values for testing
151
152     Point3::SetW(imageW);
153     Point3::SetH(imageH);
154     int maxZ = 36; // Arbitrary value for testing
155     int errorCount = 0;
156     for (int i = 0; i < imageW * imageH * maxZ; ++i)
157     { // Construct a point3 with all possible "ind" values
158         Point3 pInd(i);
159         // Construct another with the corresponding x, y, and z values
160         Point3 pXYZ(pInd.GetX(), pInd.GetY(), pInd.GetZ());
161         if (pXYZ.GetInd() != pInd.GetInd())
162             { // OOps, mismatch
163                 cout << "Error on index " << i
164                     << " XYZ.GetInd " << pXYZ.GetInd()
165                     << " Ind.Getind " << pInd.GetInd()
166                     << ", x " << pXYZ.GetX()
167                     << ", y " << pXYZ.GetY()
168                     << ", z " << pXYZ.GetZ() << endl;

```

Program static-members.cc (continued)

```

169         errorCount++;
170     }
171 }
172 if (errorCount != 0)
173 {
174     cout << "Found " << errorCount << " errors in index test" << endl;
175 }
176 errorCount = 0;
177 // Test the other way, starting with XYZ and converting to ind
178 for (int z0 = 0; z0 < maxZ; z0++)
179 {
180     for (int y0 = 0; y0 < imageH; ++y0)
181     {
182         for (int x0 = 0; x0 < imageW; ++x0)
183         {
184             Point3 pXYZ(x0, y0, z0);
185             Point3 pInd(pXYZ.GetInd());
186             // Now see if the x, y, and z match
187             if (pXYZ.GetX() != pInd.GetX() ||
188                 pXYZ.GetY() != pInd.GetY() ||
189                 pXYZ.GetZ() != pInd.GetZ())
190             {
191                 cout << "Error on index " << pInd.GetInd()
192                     << ", x1 " << pXYZ.GetX()
193                     << ", y1 " << pXYZ.GetY()
194                     << ", z1 " << pXYZ.GetZ()
195                     << ", x2 " << pInd.GetX()
196                     << ", y3 " << pInd.GetY()
197                     << ", z4 " << pInd.GetZ()
198                     << endl;
199                 errorCount++;
200             }
201         }
202     }
203 }
204 if (errorCount != 0)
205 {
206     cout << "Found " << errorCount << " errors in xyz test" << endl;
207 }
208 }
209
210

```

Program static-members.cc (continued)