```
1
    /* Program: A3P2 - Pointland!
  2
        Author: Tom Stutler
  3
        Last Date Modified: 3/19/15
  4
 5
        The intent of this program is to repeatedly provide the user with a menu of
choices to calculate/simulate
        a variety of tasks on an xy-coordinate plane given the user's entered points
 6
(x,y).
 7
 8
 9 #include <iostream>
10 #include <cmath>
11 using namespace std;
12
13 ///Define new type point which stores the x and y coordinates of a point on 2D
plane.
14 struct point
15 {
16
         float x;
 17
         float y;
 18 };
 19
 20 float dist (point aParam, point bParam);
 21 ///The purpose of this function is to take two points and return the distance
between them on the xy-coordinate plane.
 22
 23 float slope (point aParam, point bParam, bool& defParam);
 24 ///The purpose of this function is to take two points and return the slope of the
line connecting them.
 25 ///If there is no slope defined the return NULL and set slopeDefined as false.
 2.6
 27 point midpoint (point aParam, point bParam);
 28 ///The purpose of this function is to take two points and return the point that is
in the middle
 29 ///of the line segment connecting them.
 30
 31 void equation (point aParam, point bParam);
32 ///The purpose of this function is to take two point calculate and display the
equation for the line
 33 //that connects the two points together.
 34
 35
    bool collinear (point aParam, point bParam, point cParam);
 36
    ///The purpose of this function is to take three points and return true if they are
collinear or false if not.
 37
 38 point readpt ();
    ///The purpose of this function is to prompt the user to enter a point in standard
 39
format (x,y)
 40 ///and return a point variable.
 41
 42 void showpt (point aParam);
 43 ///The purpose of this function is to take a point and display it to the user in
standard format (x,y).
 44
 45
    int main()
 46 {
 47
         //Initialize loop variable.
 48
        bool repeat=true;
 49
 50
         //Loop to repeat until user selects to exit.
 51
        do {
 52
            //Declare variables.
 53
            point a, b, c, mid;
            bool slopeDefined, isCollinear;
 54
 55
            float m;
 56
            int selection;
```

```
57
              //Display menu to user.
 58
              cout << "POINTLAND\n"</pre>
 59
                   << "What do you want to do?\n"
 60
 61
                   << "1 - Find the distance between two points\n"
                   << "2 - Find slope\n"
 62
                   << "3 - Find a midpoint\n"
 63
 64
                   << "4 - Find an equation of a line\n"
 65
                   << "5 - Determine if three points are collinear\n"
 66
                   << "6 - Exit\n";
 67
 68
 69
              //Prompt user for menu selection.
 70
             cout << "Selection => ";
 71
             cin >> selection;
 72
 73
 74
             switch (selection)
 75
 76
              default:
 77
                  cout << "Please enter a valid input\n";</pre>
 78
                  break;
 79
 80
 81
                  //Prompt user for two points, pass them through dist(), then display the
 82
                  cout << "Enter point 1: ";</pre>
 83
                  a = readpt();
 84
                  cout << "Enter point 2: ";</pre>
 85
                  b = readpt();
 86
                  cout << "Distance = " << dist(a, b) << endl;</pre>
 87
                  break;
 88
 89
             case 2:
 90
                  //Prompt user for two points, pass them through slope(), then display
 91
                  cout << "Enter point 1: ";</pre>
 92
                  a = readpt();
 93
                  cout << "Enter point 2: ";</pre>
 94
                  b = readpt();
 95
 96
                  m = slope(a, b, slopeDefined);
 97
 98
                  if (slopeDefined == true) {
 99
                      cout << "Slope = " << m << endl;</pre>
100
                  } else {
101
                      cout << "Slope is not defined for a vertical line.\n";</pre>
102
103
                  break;
104
105
              case 3:
                  //Prompt user for two points, pass them through midpoint(), then display
106
the midpoint.
107
                  cout << "Enter point 1: ";</pre>
108
                  a = readpt();
109
                  cout << "Enter point 2: ";</pre>
110
                  b = readpt();
111
112
                  mid = midpoint(a, b);
113
114
                  cout << "Midpoint = ";</pre>
115
                  showpt(mid);
116
                  break;
117
118
              case 4:
119
                  //Prompt user for two points, pass them through equation().
```

```
120
                  cout << "Enter point 1: ";</pre>
121
                  a = readpt();
122
                  cout << "Enter point 2: ";</pre>
123
                  b = readpt();
124
125
                  equation(a, b);
126
                  break;
127
128
             case 5:
129
                  //Prompt user for three points, pass them through collinear(), then
display if the points are collinear.
                  cout << "Enter point 1: ";</pre>
130
131
                  a = readpt();
132
                  cout << "Enter point 2: ";</pre>
133
                  b = readpt();
134
                  cout << "Enter point 3: ";</pre>
135
                  c = readpt();
136
137
                  isCollinear = collinear(a, b, c);
138
139
                  if (isCollinear == true) {
                      cout << "These points are collinear.\n";</pre>
140
141
                  } else {
                      cout << "These points are not collinear.\n";</pre>
142
143
144
                  break;
145
146
             case 6:
147
                  cout << "Thanks for using POINTLAND\n";</pre>
148
                  repeat = false;
149
                  break;
150
              }
151
152
              //Display blank line between runs.
153
              cout << "\n";</pre>
154
         } while (repeat == true);
155
156
157
    float dist (point aParam, point bParam)
158
159
160
         return sqrt(pow((aParam.x - bParam.x), 2) + pow((aParam.y - bParam.y), 2));
161
162
163
     float slope (point aParam, point bParam, bool& defParam)
164
165
         if (aParam.x == bParam.x) {
166
              defParam = false;
167
              return 999;
168
         } else {
169
             defParam = true;
170
              return ((bParam.y - aParam.y)/(bParam.x - aParam.x));
171
         }
172
173
174
    point midpoint (point aParam, point bParam)
175
176
         point middle;
177
178
         middle.x = (aParam.x + bParam.x)/2;
179
         middle.y = (aParam.y + bParam.y)/2;
180
181
         return middle;
    }
182
183
184
    void equation (point aParam, point bParam)
```

```
185
186
         float m, b;
187
         bool slopeDefined;
188
189
         m = slope(aParam, bParam, slopeDefined);
190
         b = (aParam.y - m * aParam.x);
191
192
         if ((slopeDefined == true) && (m != 0)) {
             cout << "Equation: y = " << m << "x+" << b << endl;</pre>
193
         } else if ((slopeDefined == true) && (m == 0)) {
194
195
             cout << "Equation: y = " << b << endl;</pre>
196
         } else {
197
             cout << "Equation: x = " << aParam.x << endl;</pre>
198
199
200
201 bool collinear (point aParam, point bParam, point cParam)
202 {
203
         float m1, m2, m3;
204
         bool slopeDefined;
205
206
         m1 = slope(aParam, bParam, slopeDefined);
207
         m2 = slope(aParam, cParam, slopeDefined);
208
         m3 = slope(bParam, cParam, slopeDefined);
209
         if ((m1 == m2) | (m1 == m3) | (m2 == m3)) {
210
211
             return true;
212
         } else {
213
             return false;
214
215
216
217
    point readpt ()
218
219
         point userPoint;
220
         char parenL, parenR, comma;
221
222
         cin >> parenL >> userPoint.x >> comma >> userPoint.y >> parenR;
223
224
         return userPoint;
225
226
227
    void showpt (point aParam)
228
229
         cout << "(" << aParam.x << "," << aParam.y << ")\n";</pre>
230
```