```
1  using System;
2  using System.Collections.Generic;
3  using System.Drawing;
4  using System.Windows.Forms;
5
6
7  namespace Compx102H
8  {
9    /// <summary>
10   /// <para>Wind farm designer form for COMPX102-21H Assignment 1.</para>
11   /// <para>This starting point implementation allows the user to click
12   /// in a picture box to produce grey wind turbines. Your task is to
13   /// extend it to produce different kinds of wind turbines and to edit
14   /// them in different ways.</para>
15   /// <para>Please see the assignment handout for details. Also please
16   /// look at the WindTurbine class, which contains several methods and
17   /// properties that will help you to solve this assignment.</para>
18   /// <para>Written by Robi Malik, 2020-2021.</para>
19   /// </summary>
20   public partial class WindFarmForm : Form
21   {
22     //########################################################################
23     //# Instance Variables
24     /// <summary>
25     /// List of all the turbines currently in the wind farm.
26     /// </summary>
27     private List<WindTurbine> _turbines;
28     //########################################################################
29     //# Constructor
30     public WindFarmForm()
31     {
32       _turbines = new List<WindTurbine>();
33       InitializeComponent();
34     }
35
36
37     //########################################################################
38     //# Auxiliary Methods
39     /// <summary>
40     /// Displays all the wind turbines in the given graphics context.
41     /// </summary>
42     private void Draw(Graphics paper)
43     {
44       foreach (WindTurbine turbine in _turbines) {
45         turbine.Draw(paper);
46       }
47     }
48
49
50     //########################################################################
51     //# Event Handlers
52     /// <summary>
53     /// Event handler called when the form needs redrawing.
54     /// Causes all the wind turbines to be re-displayed.
55     /// You do not need to change this method.
56     /// </summary>
```

```
57      private void PictureBoxWindFarmPaint(object sender, PaintEventArgs e)
58      {
59        Graphics paper = e.Graphics;
60        Draw(paper);
61      }
62
63      /// <summary>
64      /// Mouse-click handler of the picture box.
65      /// You need to change this method.
66      /// </summary>
67      private void PictureBoxWindFarmMouseClick(object sender, MouseEventArgs e)
68      {
69        // Read mouse-click position
70        int x = e.X;
71        int y = e.Y;
72        int poleHeight = _heightTrackBar.Value;
73        decimal capacity = _capacityUpDown.Value;
74        int bladeCount = (int)_numberOfBladesUpDown.Value;
75        float rotorRadius = ((float)_radiusTrackBar.Value / (float)100);
76        bool clockwise = _clockwiseCheckBox.Checked;
77        Color poleColour = _poleColorButton.BackColor;
78        Color rotorColour = _rotorColorButton.BackColor;
79
80
81        if (e.Button == MouseButtons.Left) {
82          foreach (WindTurbine turbine1 in _turbines) {
83            if (turbine1.IsPoleClicked(x, y)) {
84              _heightTrackBar.Value = turbine1.PoleHeight;
85              _capacityUpDown.Value = turbine1.Capacity;
86              _numberOfBladesUpDown.Value = turbine1.NumberOfBlades;
87              _radiusTrackBar.Value = (int)(turbine1.RotorRadius * (float) 100);
88              _clockwiseCheckBox.Checked = turbine1.Clockwise;
89              _poleColorButton.BackColor = turbine1.PoleColor;
90              _rotorColorButton.BackColor = turbine1.RotorColor;
91              return;
92            }
93          }
94
95          // Create wind turbine at this position, using "grey" default for
             attributes
96          WindTurbine turbine = new WindTurbine(poleHeight, rotorRadius,
             bladeCount, clockwise, poleColour, rotorColour, capacity, x, y);
97          // Add wind turbine to farm list
98          _turbines.Add(turbine);
99          // Force redraw of the picture box to show changes
100         _pictureBoxWindFarm.Refresh();
101       }
102
103
104       else if(e.Button == MouseButtons.Right) {
105
106         foreach (WindTurbine turbine1 in _turbines) {
107           if (turbine1.IsPoleClicked(x, y)) {
108             WindTurbine newTurbine = new WindTurbine(poleHeight, rotorRadius,
                bladeCount, clockwise, poleColour, rotorColour, capacity, (int)
                turbine1.CentreX, (int) turbine1.CentreY);
```

```
109             _turbines.Remove(turbine1);
110             _turbines.Add(newTurbine);
111             return;
112           }
113         }
114       }
115
116
117
118
119
120     }
121
122     /// <summary>
123     /// Tick event handler of animation timer.
124     /// This method is called 50 times per second.
125     /// It rotates the rotors of all wind turbines slightly and redraws the
        picture box,
126     /// producing the impression of rotating wind turbines.
127     /// You do not need to change this method.
128     /// </summary>
129     private void AnimationTimerTick(object sender, EventArgs e)
130     {
131       // Rotate each rotor by 5 degrees
132       foreach (WindTurbine turbine in _turbines) {
133         turbine.Rotate(5.0f);
134       }
135       // Force redraw of the picture box to show changes
136       _pictureBoxWindFarm.Refresh();
137
138       _statusUpdate(_turbines);
139     }
140
141     private void _clearButton_Click(object sender, EventArgs e)
142     {
143       _turbines.Clear();
144       _pictureBoxWindFarm.Refresh();
145     }
146
147     private void _pictureBoxWindFarm_MouseDoubleClick(object sender,
        MouseEventArgs e)
148     {
149       int x = e.X;
150       int y = e.Y;
151       foreach (WindTurbine turbine in _turbines) {
152         if (turbine.IsPoleClicked(x, y)) {
153           _turbines.Remove(turbine);
154           _pictureBoxWindFarm.Refresh();
155           return;
156         }
157       }
158     }
159
160     private void _statusUpdate(List<WindTurbine> _turbines)
161     {
162       int count = _turbines.Count;
```

```
163         decimal totalCapacity = 0;
164         foreach(WindTurbine turbine in _turbines) {
165           totalCapacity += turbine.Capacity;
166         }
167         string statusString = "These " + _turbines.Count.ToString() + " wind
             turbines can generate a total of " + totalCapacity.ToString("f1") + "MW
              of power";
168         _statusLabel.Text = statusString;
169       }
170     }
171   }
172
```