```
1  namespace ZebCarRental
2  {
3      public partial class Form1 : Form
4      {
5
6          public Form1()
7          {
8              InitializeComponent();
9          }
10
11         private string carType;
12         const string SEDAN = "Sedan";
13         const string SUV = "SUV";
14         const string COMP = "Compact";
15         const double TAX_MAX = 10.0;
16         private string logFile = "Rental Log File.txt";
17         internal string cfgFile = "Configuration.txt";
18         private double sedanRate;
19         private double suvRate;
20         private double compRate;
21         private double taxRate;
22         private double carTypeRate;
23         private double MIN_Rate = -1;
24         const int LISTBOX = 1;
25         const int LOGFILE = 2;
26         const int BOTH = 3;
27         // ica 9, declare form to object
28         private Form2 sf;
29
30         internal double TaxRate
31         {
32             get { return taxRate; }
33             set
34             {
35                 if (value >= MIN_Rate && value <= TAX_MAX)
36                 {
37                     taxRate = value;
38                 }
39             }
40         }
41         internal double SedanRate
42         {
43             get { return sedanRate; }
44             set
45             {
46                 if (value > MIN_Rate)
47                 {
48                     sedanRate = value;
49                 }
```

```
50              }
51
52          }
53          internal double SuvRate
54          {
55              get { return suvRate; }
56              set
57              {
58                  if (value > MIN_Rate)
59                  {
60                      suvRate = value;
61                  }
62              }
63          }
64          internal double CompRate
65          {
66              get { return compRate; }
67              set
68              {
69                  if (value > MIN_Rate)
70                  {
71                      compRate = value;
72                  }
73              }
74          }
75          private void btnQuit_Click(object sender, EventArgs e)
76          {
77
78              DialogResult ButtonSelected;
79              ButtonSelected = MessageBox.Show(
80                  "Do you really want to Quit?", "Exiting...",
81                  MessageBoxButtons.YesNo,
82                  MessageBoxIcon.Question);
83              if (ButtonSelected == DialogResult.Yes)
84              {
85                  this.Close();
86              }
87          }
88
89          private void btnClear_Click(object sender, EventArgs e)
90          {
91              txtName.Clear();
92              txtDays.Clear();
93              lstOut.Items.Clear();
94              txtName.Focus();
95              rdoSedan.Checked = true;
96          }
97
98          private void btnCalc_Click(object sender, EventArgs e)
```

```
 99          {
100              //  declare/read variables from txtbx
101              string custName;
102              int totalDays;
103              double subTotal,totalCost, taxCost;
104              bool totalDaysValid, carTypeRateValid;
105              custName = txtName.Text.Trim();
106
107              // pretend widget name is a name of a person
108              // this code is not required for your project
109              // but you may want to use it if you have a customer name
110
111              string fName, lName;
112              int posSpace;
113
114              posSpace = custName.IndexOf(" ");
115              if (posSpace != -1)
116              {
117                  fName = custName.Substring(0, posSpace);
118                  lName = custName.Substring(posSpace).Trim();
119                  lstOut.Items.Add("First Name is " + fName);
120                  lstOut.Items.Add("Last Name is " + lName);
121              }
122
123
124              //  input
125              custName = txtName.Text;
126              totalDaysValid = int.TryParse(txtDays.Text, out totalDays);
127              //processing
128              if (totalDaysValid)
129              {
130                  switch (carType)
131                  {
132                      case SEDAN:
133                          carTypeRate = sedanRate;
134                          break;
135                      case SUV:
136                          carTypeRate = suvRate;
137                          break;
138                      case COMP:
139                          carTypeRate = compRate;
140                          break;
141                      default:
142                          lstOut.Items.Add("This shouldn't happen.");
143                          break;
144
145
146                  }
147                  subTotal = totalDays * carTypeRate;
```

```
148                 taxCost = subTotal * taxRate;
149                 totalCost = subTotal + taxCost;
150
151                 //  output
152                 outputTrans("*************** Beginning of Transaction " +
                      DateTime.Now.ToString("G") + "  ****************",
                      LOGFILE);
153                 outputTrans("Customer Name: " + custName,BOTH);
154                 outputTrans("Days entered is: " + totalDays, BOTH);
155                 outputTrans("Vehicle type selected:" + carType, BOTH);
156                 outputTrans(carType + "Rate: " + carTypeRate.ToString
                      ("C"), BOTH);
157                 outputTrans("Sub-total: " + subTotal.ToString("C"), BOTH);
158                 outputTrans("Sales Tax Rate: " + taxRate, BOTH);
159                 outputTrans("Tax Charge: " + taxCost, BOTH);
160                 outputTrans("Total Charge: " + totalCost, BOTH);
161
162             }
163
164             else
165             {
166                 if (!totalDaysValid)
167                 {
168                     lstOut.Items.Add("Total Days should be whole
                          number.");
169                 }
170
171             }
172
173
174
175
176
177         }
178
179         private void outputTrans (string msg, int outputType)
180
181         {
182             StreamWriter swLog;
183             if (outputType == LISTBOX || outputType == BOTH)
184             {
185                 lstOut.Items.Add(msg);
186             }
187             if (outputType == LOGFILE || outputType == BOTH)
188             {
189                 swLog = File.AppendText(logFile);
190                 swLog.WriteLine(msg);
191                 swLog.Close();
192             }
```

```
193         }
194         private void txtDays_Enter(object sender, EventArgs e)
195         {
196             txtDays.BackColor = Color.Ivory;
197         }
198
199         private void txtDays_Leave(object sender, EventArgs e)
200         {
201             txtDays.BackColor = SystemColors.Window;
202         }
203
204
205         private void txtName_Enter(object sender, EventArgs e)
206         {
207             txtName.BackColor = Color.Ivory;
208         }
209
210         private void txtName_Leave(object sender, EventArgs e)
211         {
212             txtName.BackColor = SystemColors.Window;
213         }
214
215         private void Form1_Load(object sender, EventArgs e)
216         {
217             StreamReader srCFG;
218             //create sf, which is a form2 object
219             sf = new Form2(this);
220
221             rdoSedan.Checked = true;
222             bool fileWasNotFound = true;
223             do
224             {
225                 try
226                 {
227                     srCFG = File.OpenText(cfgFile);
228                     fileWasNotFound = false;
229                     try
230                     {
231                         SedanRate = double.Parse(srCFG.ReadLine());
232                         SuvRate = double.Parse(srCFG.ReadLine());
233                         CompRate = double.Parse(srCFG.ReadLine());
234                         TaxRate = double.Parse(srCFG.ReadLine());
235                         srCFG.Close();
236                     }
237                     catch (FormatException ex)
238                     {
239                         lstOut.ForeColor = Color.Red;
240                         lstOut.Items.Add("File data  corrupted. Values
                              were set to defaults");
```

```
241                         lstOut.Items.Add(ex.Message);
242                         sedanRate = 40;
243                         suvRate = 60;
244                         compRate = 50;
245                         taxRate = .08;
246                         srCFG.Close();
247                     }
248                 }
249                 catch (FileNotFoundException ex)
250                 {
251                     MessageBox.Show(ex.Message + " Please enter a new
                        file name", "File Not Found");
252                     // OFD.InitialDirectory =
253                     OFD.Filter = "Text Files|*.txt| All Files |*.*";
254                     OFD.Title = "Open Configuration File";
255                     OFD.ShowDialog();
256                     cfgFile = OFD.FileName;
257                 }
258             } while (fileWasNotFound);
259 
260         }
261 
262         private void rdoSedan_CheckedChanged(object sender, EventArgs e)
263         {
264             if (rdoSedan.Checked)
265             {
266                 carType = SEDAN;
267             }
268         }
269 
270         private void rdoSUV_CheckedChanged(object sender, EventArgs e)
271         {
272             if (rdoSUV.Checked)
273             {
274                 carType = SUV;
275             }
276         }
277 
278         private void rdoCompact_CheckedChanged(object sender, EventArgs e)
279         {
280             if (rdoCompact.Checked)
281             {
282                 carType = COMP;
283             }
284         }
285 
286         private void settingsToolStripMenuItem_Click(object sender,
            EventArgs e)
287         {
```

```
288             sf.txtSedanRate.Text = SedanRate.ToString();
289             sf.txtSUVRate.Text = SuvRate.ToString();
290             sf.txtCompRate.Text = CompRate.ToString();
291             sf.ShowDialog();
292         }
293
294         private void showLogToolStripMenuItem_Click(object sender,
              EventArgs e)
295         {
296             const int MAX_LINES = 2000;
297             string[] logLines = new string[MAX_LINES];
298             StreamReader sr = File.OpenText(logFile);
299             int numLines = 0;
300             while (!sr.EndOfStream)
301             {
302                 logLines[numLines] = sr.ReadLine();
303                 numLines++;
304             }
305             int begin = -2;
306             int end = 4;
307             for (int i = 0; i < numLines; i++)
308             {
309                 if (logLines[i] == "The vehicle type is " + carType)
310                 {
311                     for (int j = i + begin; j <= i + end; j++)
312                     {
313                         lstOut.Items.Add(logLines[j]);
314                     }
315                 }
316
317             }
318
319             double[] grades = new double[MAX_LINES];
320
321
322
323
324             sr.Close();
325         }
326
327         private void numberArrayTestToolStripMenuItem_Click(object
              sender, EventArgs e)
328         {
329             int[] numbers = new int[50];
330             for (int i = 0; i < 25; i++)
331             {
332                 numbers[i] = i;
333             }
334             lstOut.Items.Add(numbers.Average());
```

```
335             }
336         }
337     }
338
```