```
 1  /************************************************************/
 2  /*                                                          */
 3  /*  Course: CIS 350 -- Data Structures                      */
 4  /*                                                          */
 5  /*  Project: Ch16Smp1 - Infinity Factorial.csprj            */
 6  /*                                                          */
 7  /*  Source File: Ch16Smp1 - Infinity Factorial.cs           */
 8  /*                                                          */
 9  /*  Programmer: Andrew Robinson                             */
10  /*                                                          */
11  /*  Purpose: Iterative calculation of n!.                   */
12  /*           With only a limitation of memory size.         */
13  /*                                                          */
14  /*  Classes: 1. Ch16Smp1Form : Form                         */
15  /*           2. Ch16Smp1App                                 */
16  /*                                                          */
17  /************************************************************/
18
19  using System;
20  using System.Collections.Generic;
21  using System.Windows.Forms;
22  using LibETextBox;
23
24  /************************************************************/
25  /*  Begin namespace Ch16Smp1                                */
26  /************************************************************/
27  namespace Ch16Smp1
28  {
29
30    /**********************************************************/
31    /*  1. Begin main form class Ch16Smp1Form : Form          */
32    /**********************************************************/
33    public class Ch16Smp1Form : Form
34    {
35      private Button quitButton;
36      private ETextBox valueETextBox;
37      private Label valueLabel;
38      private Label factorialLabel;
39      private Button calculateButton;
40      private Label memorySizeLabel;
41      private Label memorySizeDisplay;
42      private TextBox displayTextBox;
43      private List<uint> result;
44
45      public Ch16Smp1Form()
46      {
47        InitializeComponent();
48      }
49
50
51
52      /**********************************************************/
53      /*  Message Handlers                                      */
54      /**********************************************************/
55      private void calculateButton_Click(object sender, System.EventArgs e)
56      {
57        uint n;
58
59        if (valueETextBox.ReadUInt(out n))
60        {
```

```
61          result = new List<uint>();
62          result.Add(1);
63
64          Factorial(n);
65          displayTextBox.Text = PrintList();
66
67          memorySizeDisplay.Text = string.Format("{0}", result.Count);
68
69          valueETextBox.SelectAll();
70          valueETextBox.Select();
71        }
72      }
73
74      private void valueETextBox_TextChanged(object sender, EventArgs e)
75      {
76        displayTextBox.Text = "";
77      }
78
79      private void quitButton_Click(object sender, System.EventArgs e)
80      {
81        Application.Exit();
82      }
83
84      /*************************************************************/
85      /*  Auxiallary Methods                                       */
86      /*************************************************************/
87      private void Factorial(uint n)
88      {
89        while (n >= 2)
90        {
91          Multiply(n);
92          n--;
93        }
94      }
95
96      private void Multiply(uint times)
97      {
98        for (int i = 0; i < result.Count; i++)
99          result[i] *= times;
100
101       moveValues();
102     }
103
104     private void moveValues()
105     {
106       for (int i = 0; i < result.Count; i++)
107       {
108         uint value = result[i];
109
110         if (value > 99999 && value / 100000 > 0)
111           moveHundredThousandsPlace(i);
112         if (value > 9999 && value / 10000 > 0)
113           moveTenThousandsPlace(i);
114         if (value > 999 && value / 1000 > 0)
115           moveThousandsPlace(i);
116         if (value > 99 && value / 100 > 0)
117           moveHundredsPlace(i);
118         if (value > 9 && value / 10 > 0)
119           moveTensPlace(i);
120
```

```
121           result[i] %= 10;
122         }
123       }
124
125       private void moveTensPlace(int i)
126       {
127         expandIfNeededForI(i + 1);
128         result[i + 1] += (result[i] % 100 - result[i] % 10) / 10;
129       }
130
131       private void moveHundredsPlace(int i)
132       {
133         expandIfNeededForI(i + 2);
134         result[i + 2] += (result[i] % 1000 - result[i] % 100) / 100;
135       }
136
137       private void moveThousandsPlace(int i)
138       {
139         expandIfNeededForI(i + 3);
140         result[i + 3] += (result[i] % 10000 - result[i] % 1000) / 1000;
141       }
142
143       private void moveTenThousandsPlace(int i)
144       {
145         expandIfNeededForI(i + 4);
146         result[i + 4] += (result[i] % 100000 - result[i] % 10000) / 10000;
147       }
148
149       private void moveHundredThousandsPlace(int i)
150       {
151         expandIfNeededForI(i + 5);
152         result[i + 5] += (result[i] % 1000000 - result[i] % 100000) / 100000;
153       }
154
155       private void expandIfNeededForI(int i)
156       {
157         if (result.Count <= i)
158           for (int valueToExpandBy = i - result.Count; i > 0; i--)
159             result.Add(0);
160       }
161
162       private string PrintList()
163       {
164         string value = "";
165
166         uint max = 0;
167
168         for (int i = result.Count - 1; i >= 0; i--)
169           if (result[i] != 0 || max != 0)
170           {
171             value += result[i];
172             if (result[i] > max)
173               max = result[i];
174           }
175
176         string valueWithCommas = "";
177         for (int i = value.Length - 1; i >= 0; i--)
178           if (i % 3 == 0 && i != 0)
179           {
180             valueWithCommas += value[value.Length - 1 - i];
```

```
181             valueWithCommas += ",";
182           }
183           else
184             valueWithCommas += value[value.Length - 1 - i];
185
186         return valueWithCommas;
187       }
188
189     }  // End main form class Ch16Smp1Form
190
191     /*************************************************************/
192     /*  2. Begin application class Ch16Smp1App                   */
193     /*************************************************************/
194     public class Ch16Smp1App
195     {
196       static void Main()
197       {
198         Application.Run(new Ch16Smp1Form());
199       }
200     }  // End application class Ch16Smp1App
201
202   }  // End namespace Ch16Smp1
203
```