```
/***********************************************************/
/*                                                         */
/*  Course: CIS 350 -- Data Structures                     */
/*                                                         */
/*  Project: Ch16Smp1 - Infinity Factorial.csprj           */
/*                                                         */
/*  Source File: Ch16Smp1 - Infinity Factorial.cs          */
/*                                                         */
/*  Programmer: Andrew Robinson                            */
/*                                                         */
/*  Purpose: Iterative calculation of n!.                  */
/*           With only a limitation of memory size.        */
/*                                                         */
/*  Classes: 1. Ch16Smp1Form : Form                        */
/*           2. Ch16Smp1App                                */
/*                                                         */
/***********************************************************/

using System;
using System.Collections.Generic;
using System.Windows.Forms;
using LibETextBox;

/***********************************************************/
/*  Begin namespace Ch16Smp1                               */
/***********************************************************/
namespace Ch16Smp1
{

  /***********************************************************/
  /*  1. Begin main form class Ch16Smp1Form : Form           */
  /***********************************************************/
  public class Ch16Smp1Form : Form
  {
    private Button quitButton;
    private ETextBox valueETextBox;
    private Label valueLabel;
    private Label factorialLabel;
    private Button calculateButton;
    private Label memorySizeLabel;
    private Label memorySizeDisplay;
    private TextBox displayTextBox;
    private List<uint> result;

    public Ch16Smp1Form()
    {
      InitializeComponent();
    }



    /***********************************************************/
    /*  Message Handlers                                       */
    /***********************************************************/
    private void calculateButton_Click(object sender, System.EventArgs e)
    {
      uint n;

      if (valueETextBox.ReadUInt(out n))
      {
        result = new List<uint>();
        result.Add(1);

        Factorial(n);
        displayTextBox.Text = PrintList();

        memorySizeDisplay.Text = string.Format("{0}", result.Count);

        valueETextBox.SelectAll();
        valueETextBox.Select();
      }
    }

    private void valueETextBox_TextChanged(object sender, EventArgs e)
    {
      displayTextBox.Text = "";
    }

    private void quitButton_Click(object sender, System.EventArgs e)
    {
      Application.Exit();
    }

    /**************************************************************/
    /*  Auxiallary Methods                                        */
    /**************************************************************/
    private void Factorial(uint n)
    {
      while (n >= 2)
      {
        Multiply(n);
        n--;
      }
    }

    private void Multiply(uint times)
    {
      for (int i = 0; i < result.Count; i++)
        result[i] *= times;

      moveValues();
    }

    private void moveValues()
    {
      for (int i = 0; i < result.Count; i++)
      {
        uint value = result[i];

        if (value > 99999 && value / 100000 > 0)
          moveHundredThousandsPlace(i);
        if (value > 9999 && value / 10000 > 0)
          moveTenThousandsPlace(i);
        if (value > 999 && value / 1000 > 0)
          moveThousandsPlace(i);
        if (value > 99 && value / 100 > 0)
          moveHundredsPlace(i);
        if (value > 9 && value / 10 > 0)
          moveTensPlace(i);

        result[i] %= 10;
      }
    }

    private void moveTensPlace(int i)
    {
      expandIfNeededForI(i + 1);
      result[i + 1] += (result[i] % 100 - result[i] % 10) / 10;
    }

    private void moveHundredsPlace(int i)
    {
      expandIfNeededForI(i + 2);
      result[i + 2] += (result[i] % 1000 - result[i] % 100) / 100;
    }

    private void moveThousandsPlace(int i)
    {
      expandIfNeededForI(i + 3);
      result[i + 3] += (result[i] % 10000 - result[i] % 1000) / 1000;
    }

    private void moveTenThousandsPlace(int i)
    {
      expandIfNeededForI(i + 4);
      result[i + 4] += (result[i] % 100000 - result[i] % 10000) / 10000;
    }

    private void moveHundredThousandsPlace(int i)
    {
      expandIfNeededForI(i + 5);
      result[i + 5] += (result[i] % 1000000 - result[i] % 100000) / 100000;
    }

    private void expandIfNeededForI(int i)
    {
      if (result.Count <= i)
        for (int valueToExpandBy = i - result.Count; i > 0; i--)
          result.Add(0);
    }

    private string PrintList()
    {
      string value = "";

      uint max = 0;

      for (int i = result.Count - 1; i >= 0; i--)
        if (result[i] != 0 || max != 0)
        {
          value += result[i];
          if (result[i] > max)
            max = result[i];
        }

      string valueWithCommas = "";
      for (int i = value.Length - 1; i >= 0; i--)
        if (i % 3 == 0 && i != 0)
        {
          valueWithCommas += value[value.Length - 1 - i];
          valueWithCommas += ",";
        }
        else
          valueWithCommas += value[value.Length - 1 - i];

      return valueWithCommas;
    }

  }  // End main form class Ch16Smp1Form

  /*********************************************************/
  /*  2. Begin application class Ch16Smp1App               */
  /*********************************************************/
  public class Ch16Smp1App
  {
    static void Main()
    {
      Application.Run(new Ch16Smp1Form());
    }
  }  // End application class Ch16Smp1App

}  // End namespace Ch16Smp1

```