Visual Basic Programming Assignment #1 (50 Points)

Due: Noon on Thursday, January 21, 2010

Your first VBPA is to implement a basic calculator, as illustrated below.

The calculator form contains 21 buttons for user input and one textbox for the output of results. An executable version is available on the course Web site (http://www.cs.siue.edu/wwhite/CS275/Syllabus.htm).

The form is displayed below, along with the required names and properties of the various controls.

- **Form FrmCalculator**
  - BackColor: Silver
  - Font: Arial, 12pt, Bold
  - FormBorderStyle: FixedToolWindow
  - MaximumSize, MinimumSize, and Size: 300,325
  - Text: Calculator

- **Buttons Btn1, Btn2, Btn3, Btn4, Btn5, Btn6, Btn7, Btn8, Btn9, Btn0**
  - FlatStyle: System
  - Font: Arial, 22pt, Bold
  - Location: 1: (31,73); 2: (73,73); 3: (115,73); 4: (31,115); 5: (73,115); 6: (115,115); 7: (31,157); 8: (73,157); 9: (115,157); 0: (73,199)
  - Margin: 1,1,1,1
  - Size: 40,40
  - Tag: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
  - Text: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0

- **Button BtnPosNeg**
  - FlatStyle: System
  - Font: Arial, 14pt, Bold
  - Location: 31,199
  - Margin: 1,1,1,1
  - Size: 40,40
  - Text: +/-

- **Button BtnDecimal**
  - FlatStyle: System
  - Font: Arial, 14pt, Bold
  - Location: 115,199
  - Margin: 1,1,1,1
  - Size: 40,40
  - Text: .

- **Buttons BtnDivide, BtnMultiply, BtnSubtract, BtnAdd**
  - FlatStyle: System
  - Font: Arial, 20pt, Bold
  - Location: /: (169,73); *: (169,115); -: (169,157); +: (169,199)
  - Margin: 1,1,1,1
  - Size: 40,40
  - Text: /, *, -, +

- **Button BtnClear**
  - FlatStyle: System
  - Font: Arial, 20pt, Bold
  - Location: 211,73
  - Margin: 1,1,1,1
  - Size: 40,40
  - Text: C

- **Button BtnSquare**
  - FlatStyle: System
  - Font: Arial, 10pt, Bold
  - Location: 211,115
  - Margin: 1,1,1,1
  - Size: 40,40
  - Text: x^2

- **Button BtnReciprocal**
  - FlatStyle: System
  - Font: Arial, 11pt, Bold
  - Location: 211,157
  - Margin: 1,1,1,1
  - Size: 40,40
  - Text: 1/x

- **Button BtnEquals**
  - FlatStyle: System
  - Font: Arial, 20pt, Bold
  - Location: 211,199
  - Margin: 1,1,1,1
  - Size: 40,40
  - Text: =

- **Button BtnOff**
  - FlatStyle: System
  - Font: Arial, 16pt, Bold
  - Location: 104,245
  - Margin: 1,1,1,1
  - Size: 86,38
  - Text: OFF

- **TextBox TxtResult**
  - BackColor: Black
  - Font: Courier New, 18pt, Bold
  - ForeColor: White
  - Location: 9,21
  - MaxLength: 15
  - ReadOnly: True
  - Size: 273,35
  - TextAlign: Right
The twelve buttons on the left side of the form will be used to input numerical values, while the eight buttons on the right side of the form will be used to operate upon the numerical values that are input. The OFF button, of course, merely closes the form and terminates the application.

The application needs to keep track of at most two input values, maintained as string variables. A boolean flag should be used to indicate whether the calculator is accepting input for the first operand or the second operand. Both operands should be initialized to “0” when the application starts, as well as whenever the Clear button is pressed.

When the user presses a numerical button (1, 2, 3, 4, 5, 6, 7, 8, 9, 0), the corresponding digit should be appended to the current operand (although leading zeros should be ignored, so, for instance, “005” should be interpreted as “5”). When the user presses the Decimal key, a decimal point is appended to the current operand (unless, of course, a decimal point has already been inserted in the string, in which case the additional decimal point is ignored). Finally, when the +/- key is pressed, a negative sign (“-”) is appended to the front of the operand if it doesn't already have one; otherwise, the existing negative sign is removed. (Useful string-related functions include StartsWith, EndsWith, Substring, ToString, and CStr.)

When one of the binary operator buttons (/, *, -, +) is pressed, the second operand is activated, if it hasn’t already been activated. If the second operand has already been activated then the operator that caused its activation should be executed on the existing pair of operands, with the result placed in the first operand. In any case, a new operator causes the second operand to become activated and ready for user input. When the user presses the Equals button, the active operation is performed on the two operands. (Converting strings to double values can be performed via CDbl.)

Finally, when one of the unary operator buttons (the Square button or the Reciprocal button) is pressed, the corresponding operation is performed on the second operand if it exists, or the first operand if there is no second operand.

The output in the TextBox will always be the current operand until the Equals button or one of the unary operation buttons is pressed, at which point the results of the active operation is displayed. When a division by zero is attempted (either via the Divide button of the Reciprocal button), an ERROR message should be displayed and both operands should be cleared.

Your completed application is expected to look and behave exactly like the executable that has been provided. Two more advanced features, however, will be considered extra credit:

1. Fixed-Point Display: All results displayed in the TextBox are in fixed-point format, not scientific notation (e.g., 0.0000000000123 instead of 1.23E-11).

2. Overflow Avoidance: When operations yield values that are beyond the system’s capacity, the TextBox should merely display an OVERFLOW message and clear both operands.

Zip-compress your entire project file and copy it to your dropbox by Thursday, January 21, 2010, at Noon. Late assignments are not accepted without verifiable medical documentation. You must write your own code, and no one but the instructor may see your code.