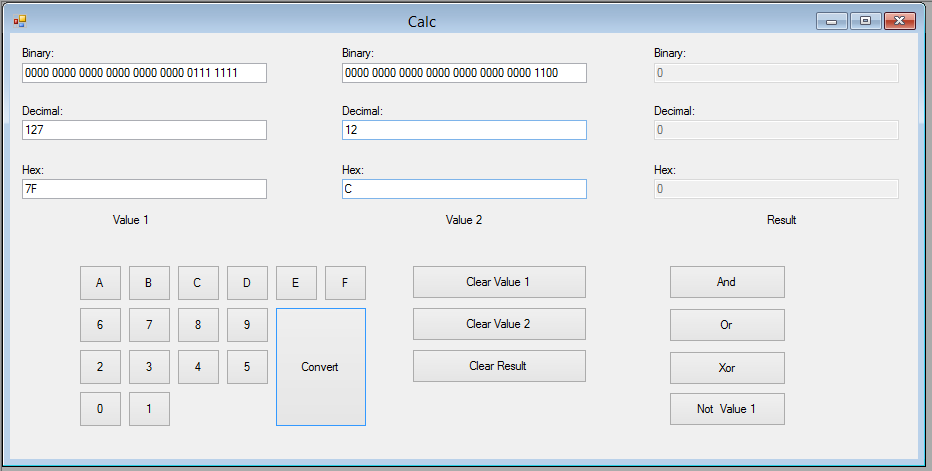
***CIS 311 – Assignment 6***

Your next assignment will require you to build an MDI calculator application. Your main form should have a main menu with the following options: File, Window and Help. The Window option should serve as a Windowlist and have the Tile and Cascade options available under it. Help should have an About option that activates an about box dialog (you can autogenerate the about box if you want). File should have submenu option choices of New and Exit. New will instantiate a new child calculator form and Exit, well, we’ll talk about that one in a minute.

The new child calculator form should look like this:



You will notice that the purpose of this calculator is to convert numbers and display them in binary, hex and decimal. If you take a look at the leftmost textboxes, these are representing “value 1” and the value decimal 127 is represented in the three aforementioned number bases. As any given textbox is clicked upon, only the appropriate digits for that base should be made clickable, e.g. for hex (0-9A-F), for decimal (0-9) and for binary (0-1). Once a value is entered into one of the textboxes for “value 1”, click Convert and the other base representations should be computed and shown.

Likewise in the middle of the screen there is a “value 2” that works the same way (in our case it is a decimal 12). You will convert the various bases for “value 2” using the same Convert button as “value 1,” so somehow you need to know which value is being worked with. In the middle of the screen, you will notice that there are buttons to clear all 3 sets of values (value 1, value 2 and result -- the rightmost textboxes which are never enabled to be input into).

The result textboxes only come into play when performing logic operations on the values in “value 1” and “value 2.” If the And button is clicked, the value in “value 1” will be “anded” with the value in “value 2” and the answer of the And function will show up in all three base representations in the “result” section of the form. Likewise there is an Or function and a Xor function too. The Not function only works on a single value, so whatever value is in “value 1” will be “notted.”

When the user tries to close a child window, any child window that does not have “0” as the value in the “value 1” and “value 2” sections should prompt the user to ensure he/she really wants the child to close. If the user still wants to close after being informed, he/she will be allowed to close, otherwise the child window should stay open. If the child contains the value “0” for both “value 1” and “value 2,” there will be no prompt about closing.

When the user selects either the Exit menu option or the upper right X on the main application form, the application should strobe through all of the child windows ensuring that they all have 0 values in “value 1” and “value 2” before exiting. Any child that does not, should prompt for closing as outlined above. If all child forms are closed then the main application MDI form should shut down.

Finally, since some of you have not yet taken CIS 333, I have provided a few hints about how to perform the number conversions here. In general, try to get things to decimal first and then convert to hex or binary.

* To convert a decimal value to binary, here’s some pseduocode:

//32 possible bits – biggest size value we’re going to mess with

For Loop = 31 DownTo 0

//Compare the decimal value we’re trying to convert with

//a power of 2 value – if we And them and get a True value

//back, that means that that power’s bit is on, e.g. a 1...

If (DecimalValue And (2 ^ Loop)) = True Then

strBinaryValue = strBinaryValue & "1"

Else

//...otherwise we need to set that power at 0

strBinaryValue = strBinaryValue & "0"

End If

Next

* Converting from binary to decimal will essentially run in the opposite direction of that above, e.g. you will add increasing powers of 2 together from right to left in the binary number wherever you encounter a 1 in the binary number.
* You should only convert decimal values to hex since the hex function handles this for you, e.g. if a user inputs a binary value, get it to decimal and then convert the decimal value to hex.
* To convert from hex to decimal, we want to use the following logic: Convert.ToInt64(*string containing a hex value*, 16)

You gotta love the power built into the Convert class! Unfortunately it only outputs a decimal number, so we had to build up the other routines above!

Complete your assignment and place your entire solution in a zip file, which you will upload to Canvas. Turn in a cover sheet, your program source code and screenshots of your program’s execution stapled together in that order in class.