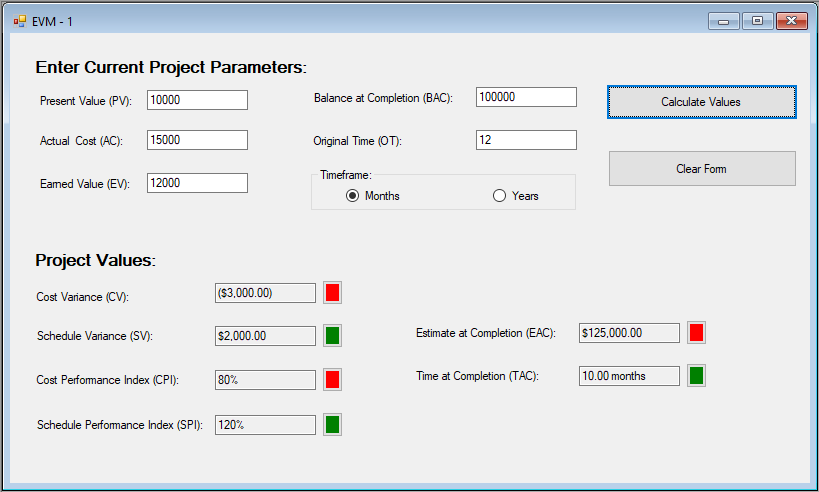
***CIS 311 – Assignment 6***

Your next assignment will require you to build an MDI calculator application. Your MDI parent 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 (Horizontal and Vertical) 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 determine the health of a project by examining its Earned Value Management (EVM) numbers.

The upper portion of the calculator (Current Project Parameters) allows a user to enter the four dollar amount values (PV, AC, EV and BAC) of where a project is at a particular point in time along with the original time (OT) that the project was supposed to take in month/year time units.

Once the user has entered this information, pressing the Calculate Values button will generate all of the bottom values through calculations (CV, SV, CPI, SPI, EAC and TAC). Notice that some are formatted as currency, some as percentages and some as 2 point place decimal numbers. Also note that the TAC value has the calculated value along with the proper timeframe units as specified with the upper portion’s radio button. Furthermore, next to each project value is a button that is color coded showing the health of that particular statistic: green (good), yellow (caution/breaking even) and red (bad). Here are the formulas for calculating the various values along with how to determine what a value means.

* CV = EV - AC; if CV is negative, this means costs are higher than expected, which is bad; a CV of 0 is yellow; a positive CV is what we are after – green
* SV = EV - PV; if SV is negative, this means time is taking longer than expected so things are costing more, which is bad; a SV of 0 is yellow; a positive SV is what we are after – green
* CPI = EV / AC; CPI is a percentage: anything over 100% is under budget (good, so green), anything under 100% is over budget (red); 0% is yellow.
* SPI = EV / PV; SPI is a percentage: anything over 100% is under time (good so, green), anything under 100% is over time (red); 0% is yellow.
* EAC = BAC / CPI; if EAC > BAC this is bad (things are costing more than what we thought they would)
* TAC = OT / SPI; if TAC > OT this is bad (things are taking longer than we thought they would)

Some things to watch out for: AC cannot be 0 in the CPI calc and PV cannot be 0 in the SPI calc – if they are, then you need to write an appropriate message in the resultant output box, for example in CPI you would write “AC cannot be $0.00”. Likewise, you would do the same thing as appropriate for the SPI box. Now if you wrote one of those strings into the CPI or SPI box, then you cannot calculate EAC and/or TAC. You should write an appropriate message there too. For example, in EAC you would write “CPI must be numeric” if you don’t have a numeric value in the CPI box. Ditto for the TAC box.

The Clear Form button is pretty straightforward as it will reset all numeric fields to 0s and reset the button colors back to their default form color (typically gray).

One final requirement involves closing the child windows. When the user tries to close a child window, any child window that does not have “0” in each of the PV, AC, EV, BAC and OT values 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 and the child disappears, otherwise the child window should remain open. If the child contains all 0 values for those items, there will be no prompt about closing.

When the user selects either the Exit menu option or the upper right X on the MDI parent form, the application should strobe through each of the child windows following the same rules outlined above. If all child forms are closed, then the main application MDI form should close as well and the application will terminate.

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