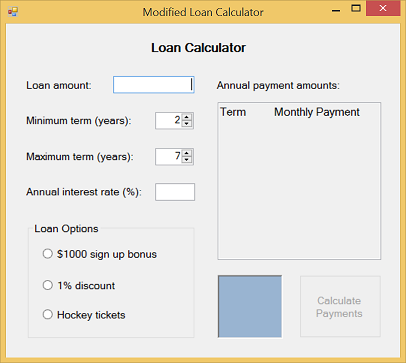
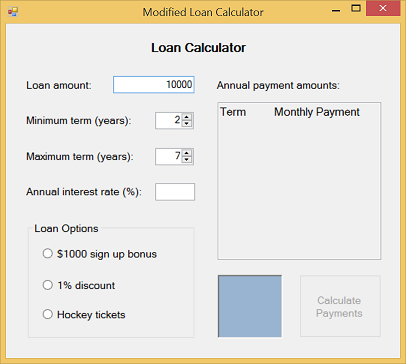
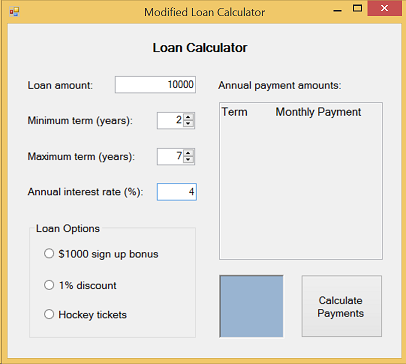
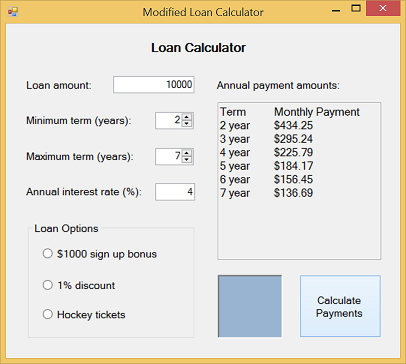
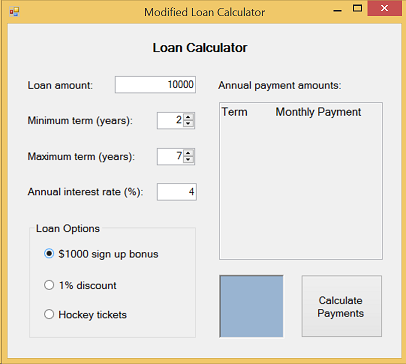
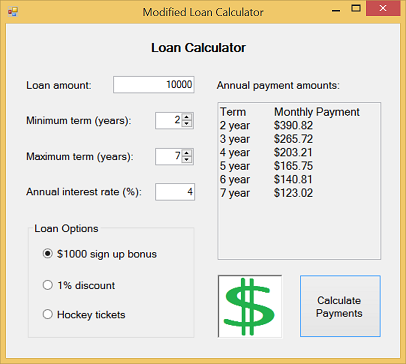
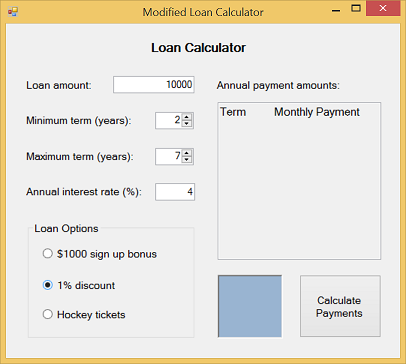
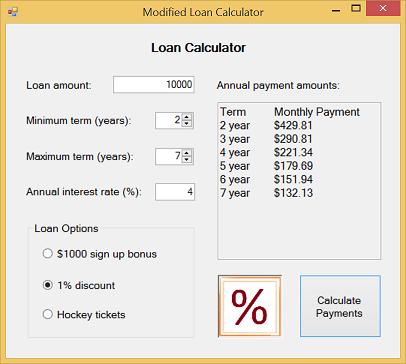
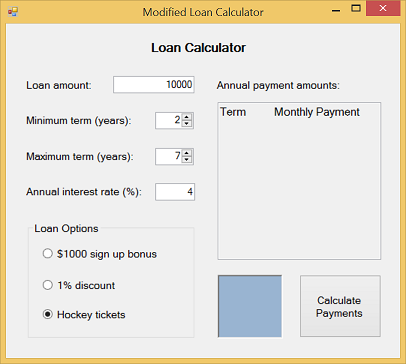
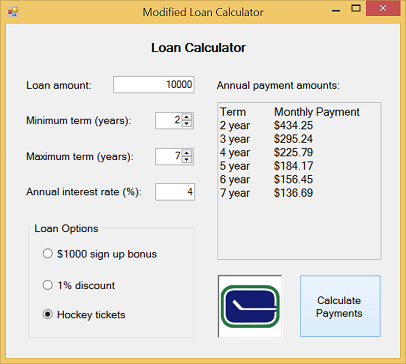
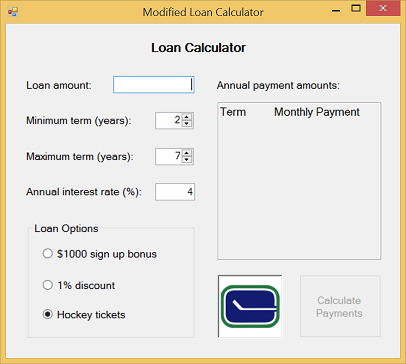
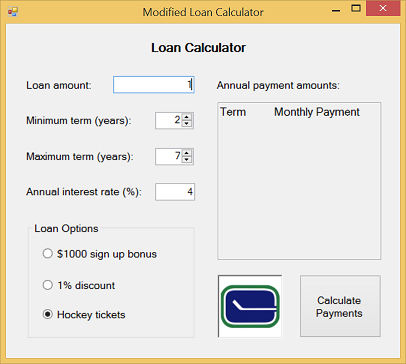
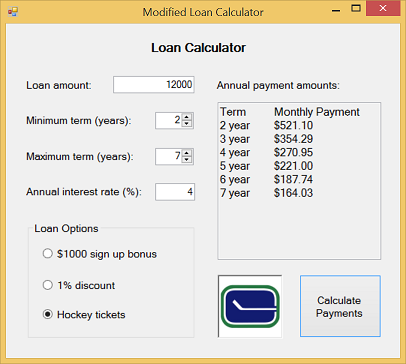
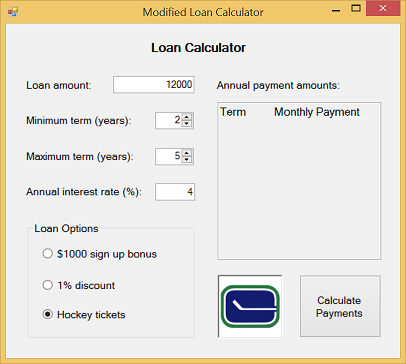
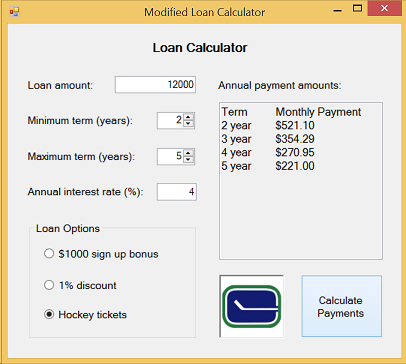
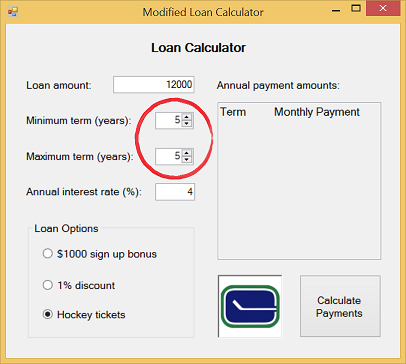
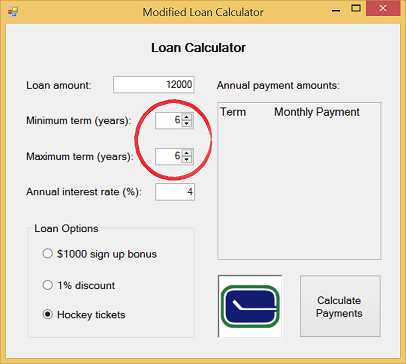
Here is a demonstration of the use of this app:  
  
When the app first loads:



The user enters a value into the **Loan amount** *TextBox*:  
  
  
  
Notice that the **Calculate Payment** *Button* is still disabled. The user enters a value into the **Annual interest rate** *TextBox*:  
  
  
  
Notice that the **Calculate Payment** *Button* is now enabled.   
  
The user then clicks the **Calculate Payment** button:  
  
  
  
Each row in the payment Multiline ReadOnly *TextBox* indicates the monthly payment amounts that would pay down the loan balance to 0 by the end of the specified term. For example, the third line of output shows that if you took a $10,000 loan with a 3-year term at 4% interest, a monthly payment of $295.24 would be needed to pay off the loan after 3 years.  
  
Notice that no image displays in the *PictureBox* because no loan option was selected.  
  
The user selects the **Bonus** *RadioButton*:  
  
  
  
When the user selects a *RadioButton*, the contents of the Multiline ReadOnly *TextBox* are cleared (except for the heading line). This is done to keep the GUI in a consistent state.  
  
The user then clicks the **Calculate Payment** button:  
  
  
  
The new monthly payments are listed in the Multiline ReadOnly *TextBox*. The payments reflect a loan amount that is discounted by $1000, making the effective loan amount $9,000. Notice that the monthly payments for the 3-year term loan is now reduced from $295.24 to $265.72.  
  
The user then clicks the **1% discount** *RadioButton*:  
  
  
  
When the user selects a different *RadioButton*, the contents of the Multiline ReadOnly *TextBox* are cleared (except for the heading line) keeping the GUI in a consistent state. Notice that the **Bonus***RadioButton* is deselected automatically as required since a user can only have one option in this version of the application.  
  
The user then clicks the **Calculate Payment** button:  
  
  
  
The new monthly payments are listed in the Multiline ReadOnly *TextBox*. The payments reflect a $10000 loan amount with an effective annual interest rate of 3% (resulting from 4% discounted by 1%). Notice that the monthly payments for the 3-year term loan is now $290.81.  
  
The user selects the **Hockey tickets** *RadioButton*:  
  
  
  
When the user selects a different *RadioButton*, the contents of the Multiline ReadOnly *TextBox* are cleared (except for the heading line) keeping the GUI in a consistent state.  
  
The user then clicks the **Calculate Payment** *Button* to display the new monthly payment calculations:  
  
  
  
Notice that the new monthly calculations appear and the Vancouver Canucks logo appears.   
  
Next, the user wants to change the loan amount to 12000 by first clearing the **Loan amount** *TextBox*:  
  
  
  
Notice that the contents of the Multiline ReadOnly *TextBox* are cleared (except for the heading line) keeping the GUI in a consistent state. Also notice that the **Calculate Payments** *Button* is disabled.   
  
As soon as the user type the '1' of 12000 the **Calculate Payments** *Button* becomes enabled:  
  
  
  
Notice the logo is still displayed in the PictureBox.  
  
The user finishes entering the new loan amount 12000 and clicks **Calculate Payments** *Button*:  
  
  
  
Now let's look at the NUD controls.  
  
The user clicks the down arrow on the **Maximum term** *NUD*:  
  
  
  
As soon as either of the *NUD*s change the contents of the Multiline ReadOnly *TextBox* are cleared (except for the heading line) keeping the GUI in a consistent state.   
  
The user then clicks the **Calculate Payment** *Button*:  
  
  
  
Notice that only the calculations are made for terms of 2 year to 5 years.  
  
Now, let's say the user wants to display the calculations for terms in the range of 6 years to 10 years. The user could click the up arrow of the **Maximum term** *NUD* until it displays 10 followed by the user clicking the up arrow of the **Minimum term** *NUD* until it display 6.  
  
However, if the user first clicks the up arrow of the **Minimum term** *NUD* until it displays 6, we could be in a situation where the minimum term is higher than the maximum term. To avoid this situation, as the user increases the **Minimum term** *NUD*, the **ValueChanged** event handler for the **Minimum term** *NUD* should check to make sure the **Maximum term** *NUD* has a **Value** property that is at least the **Value** of the **Minimum term** *NUD*. If it doesn't, the **Valu**e property of the **Maximum term** *NUD* should be increased to match the **Value** of the **Minimum term** *NUD*.  
  
For example, suppose the new range the user wants is 6 years to 10 years. The user could click the up arrow on the  **Minimum term** *NUD* until it displays 6. When she/he has clicked it to 5, the GUI would look like this:  
  
  
  
When she/he clicks the up arrow on the the **Minimum term** *NUD* one more time, the value displayed in the **Maximum term** *NUD* also increases by one to make sure the minimum term can never be greater than the maximum term:  
  
  
  
Note that a complementary effect should be handled in the case of the user decreasing the **Maximum term** *NUD* beyond the minimum term value.  
  
To finish the adjustment for the 6 to 10-year term specifications, the user would click the up arrow on the **Maximum term** *NUD* until its value is 10 and then the user would click the **Calculate Payment** *Button*:  
  
 