Homework 3 Guide

Background

This homework will utilize For Loops, While Loops, and Case Structures to implement a fun activity.

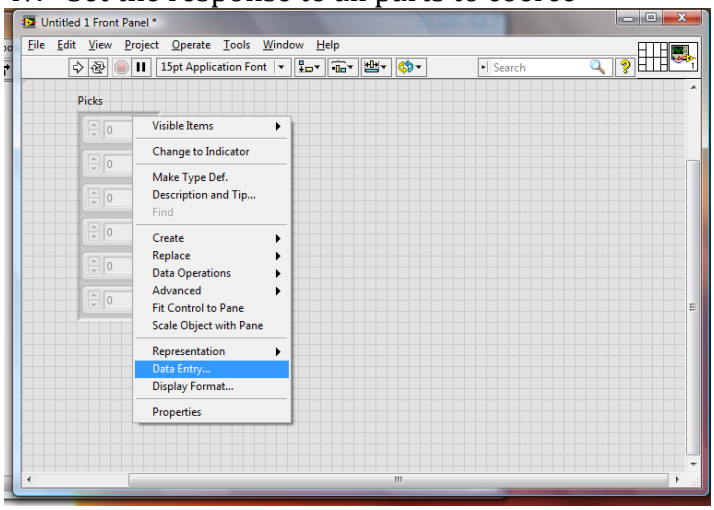
Task

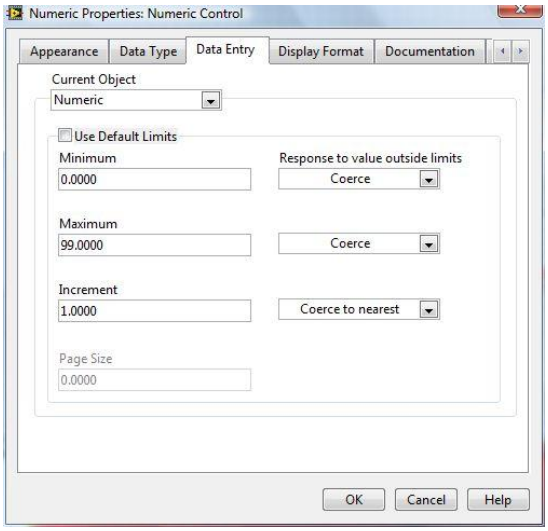
In this homework you will create a Pick 6 lottery. The user will pick 6 numbers, and then the computer will randomly draw 6 numbers between 0 and 99. The more numbers that match, the bigger the payout. You start with a certain amount in the bank and can vary how much you bet.

Steps

Front Panel – this is what the user sees and interacts with

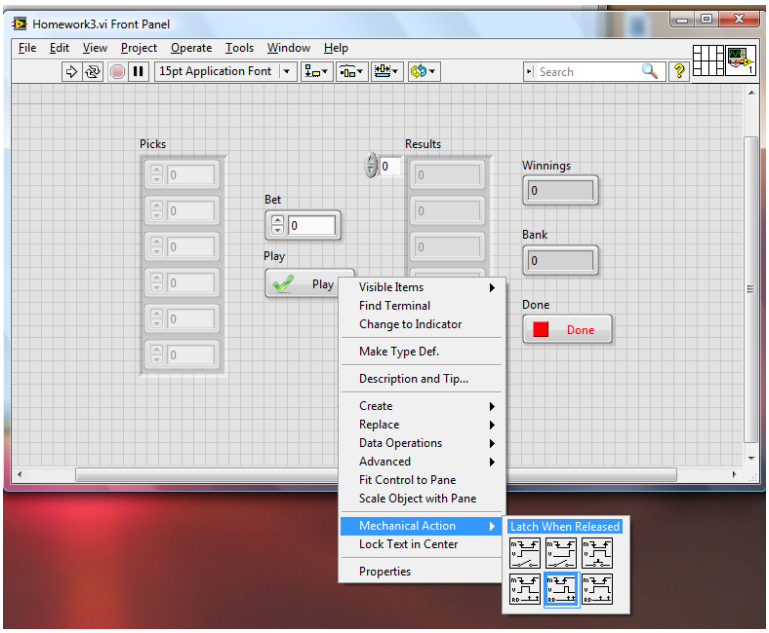
1) Create an array of numeric controls. This is where the user will enter their lotto number picks.
    a. Change the name of the array to “Picks”
    b. Drag an array shell onto the front panel
    c. Drag a numeric control into the shell
    d. Right click an element in the array and select “Data Entry”
        i. Set the minimum to 0
        ii. Set the maximum to 99
        iii. Set the increment to 1
        iv. Set the response to all parts to coerce



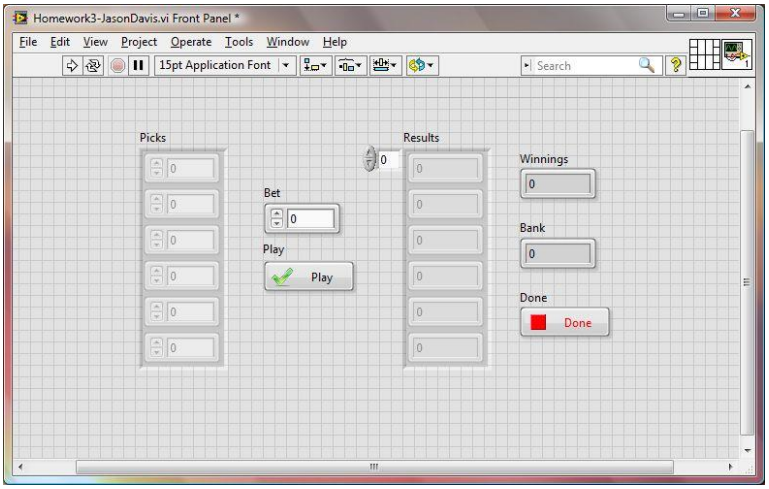


   e. Click the corner of the array to display 6 elements (may be uninitialized)

2) Create an array of numeric indicators. This will be where the lotto results will show up.
   a. Select the array "Picks" and copy it
      i. Note: copy items by selecting control and then dragging the item you wish to copy (use option key instead if you are on a Mac)
   b. Change the name of the array to "Results"
   c. Change the array to an indicator by right clicking and selecting "Change to Indicator"
3) Create a Numeric control and change the label to "Bet". This is where the user will enter how much they want to bet for their current number selections.
4) Create a Numeric indicator and change the label to "Bank". This will show how much money is in the bank.
5) Create a Numeric indicator and change the label to "Winnings". This will show how much money the user has won.
6) Create a Boolean and change its label to "Play". This will be the button that the user clicks when they want to start playing the lotto game.
   a. The mechanical action of this should be Latch when Released



7) Create a Boolean and change its label to "Done". This is the button that the user will press when they are done playing the game and want to stop.
    a. The mechanical action of this should be Latch when Released

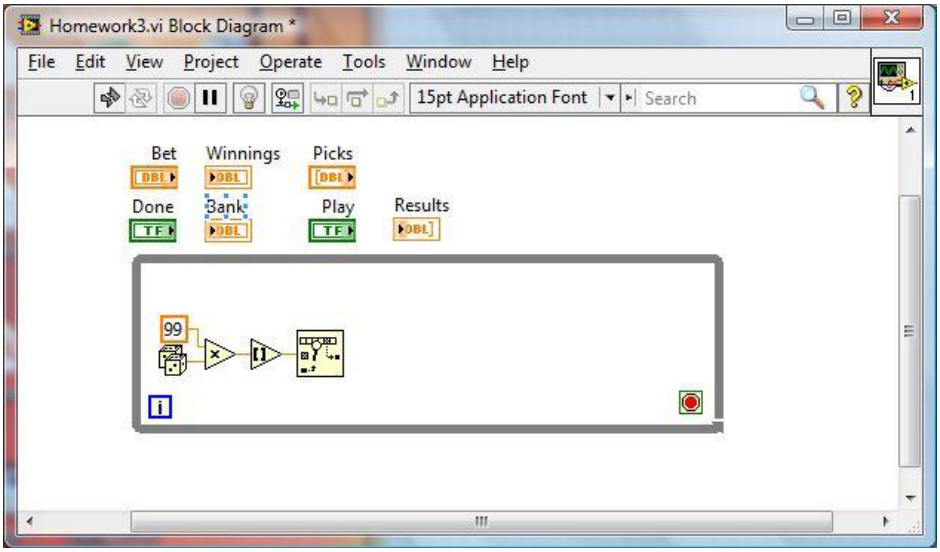


Block Diagram – this is the actual code that executes

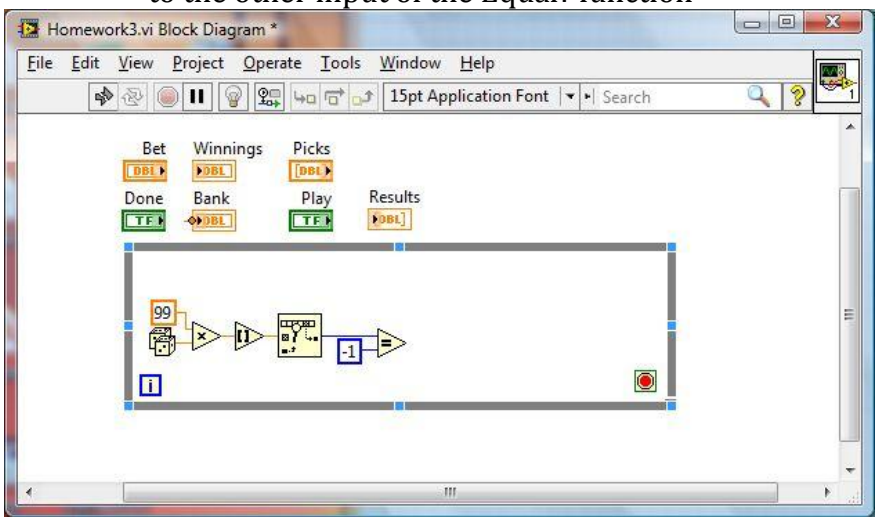
1) Create a random number that is whole and in the range of 0-99. This will be how the random lotto results are selected.
    a. Create a random number 0-98 by multiplying 99 by the function "Random Number (0-1)" found in the numeric palette
        i. The number generated is greater than or equal to 0, but less than 1, so if we want a range of 0-98, we need to multiply the random number generator by 99, since that will return to us a number greater than or equal to 0 but less than 99.

b. Use the “Round to Nearest” function to make the random number a whole number (Functions> Programming> Numeric> Round to Nearest)

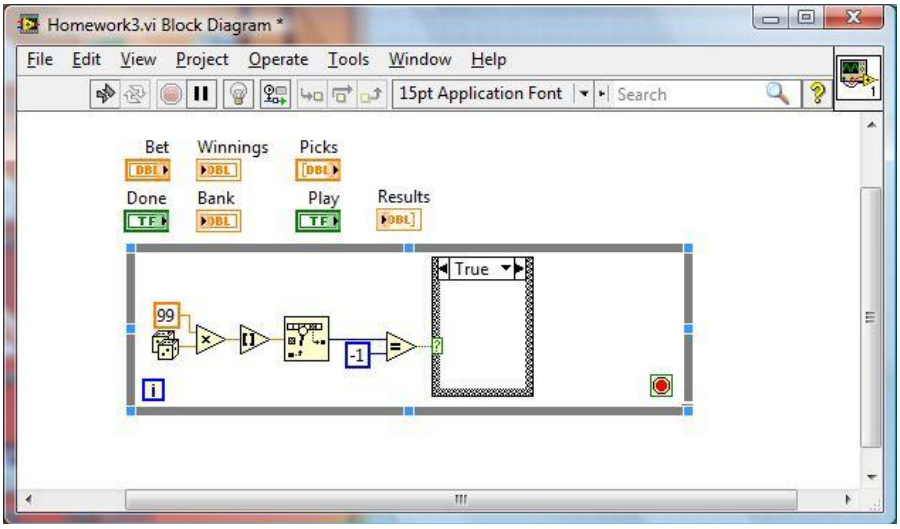
2) Generate an array of 6 non-repeating numbers. This will be the lotto results.
   a. Place the random number inside a While Loop
   b. Place a “Search 1D Array” Function down (Functions> Programming> Array> Search 1D Array) and wire the Round to Nearest Function to the input labeled “element”



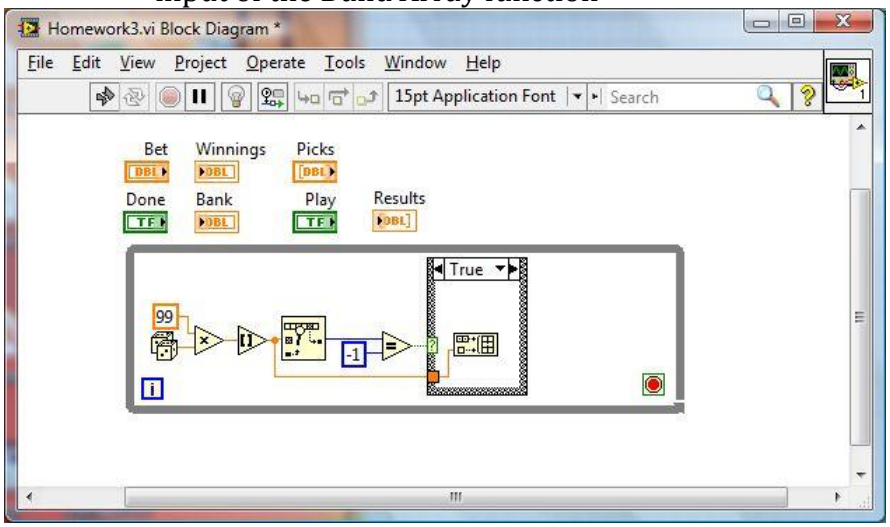
c. Place an Equal? comparison block connected to the output of the Search 1D Array function This will check whether or not the lotto result was previously selected, ensuring that the lotto results do not repeat.
    i. Connect a numeric constant with a value of "-1" and of type I32 to the other input of the Equal? function



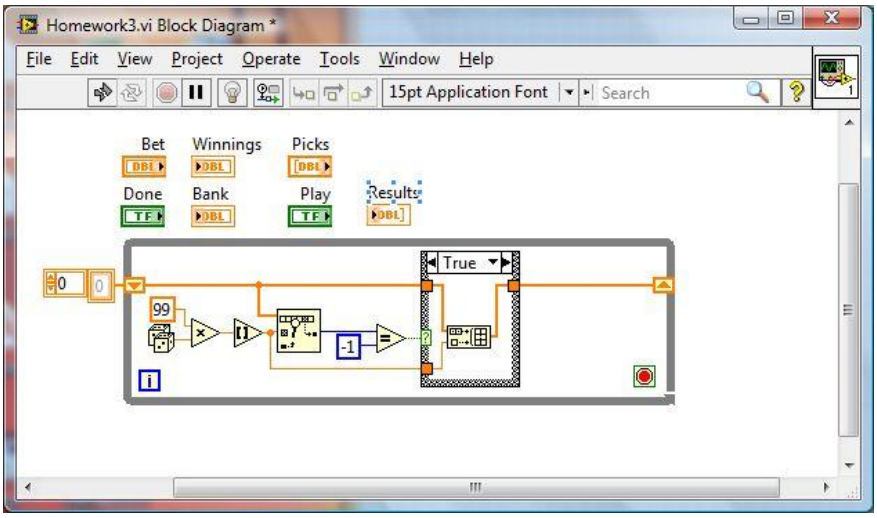
d. Place a Case Structure inside the While Loop. This is where we will decide whether or not to add the lotto result to the list of lotto results.
    i. The Case Structure is located under Programming> Structures> Case Structure
    ii. Connect the Equal? function output to the case selector terminal of the Case Structure



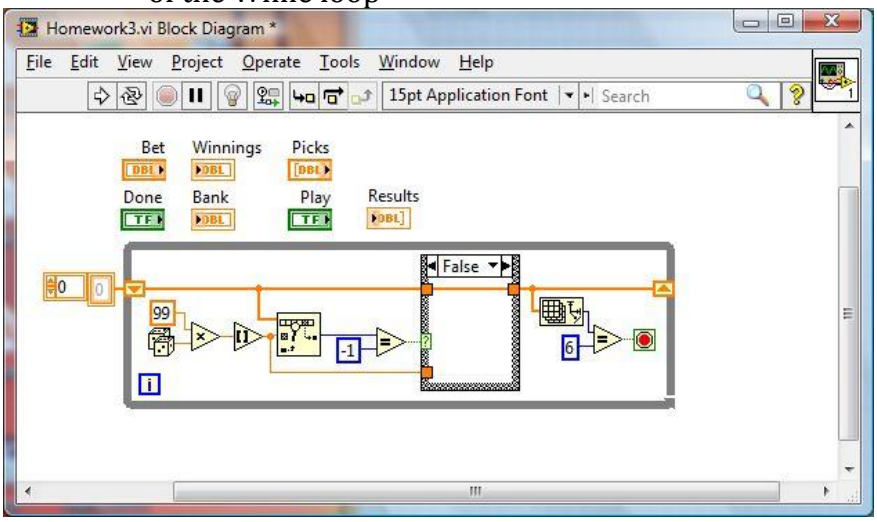
e. Place a Build Array (Functions> Programming> Array> Build Array) function inside of the “True” case. This case will add the lotto result to the lotto result array list.
    i. Expand the build array so that it has two inputs
    ii. Wire the output of the Round to Nearest function to the bottom input of the Build Array function



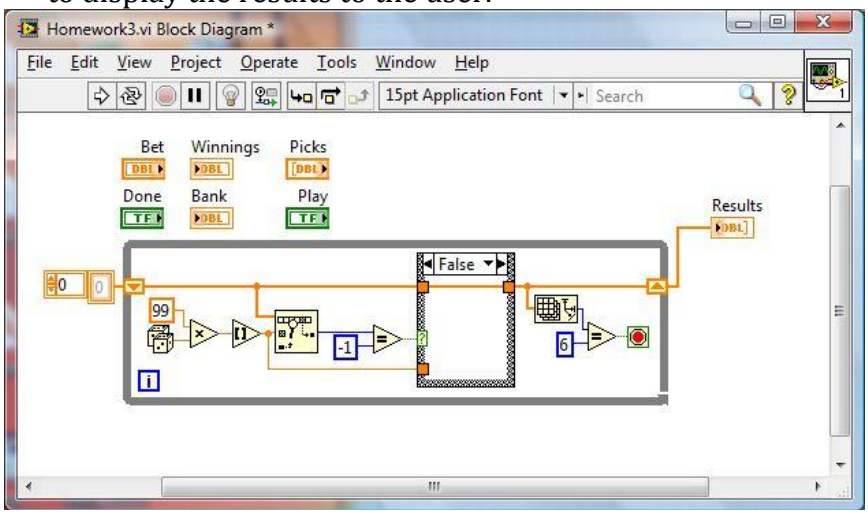
f. Create an initialized shift register that contains the array. This is so that the while loop remembers the lotto result array that was created so far.
    i. Wire the output of the Build array function to the outside of the While Loop
    ii. Right click the tunnel and select “Replace with Shift Register”
    iii. On the input side of the shift register (very left side of the While Loop) create an uninitialized array by right clicking the terminal and selecting “Create Constant”
    iv. On the inside left side of the While loop, connect the shift register to the top input of both the Search 1D array and the Build Array functions
    v. In the False case of the Case Structure wire the array through (nothing added to it)



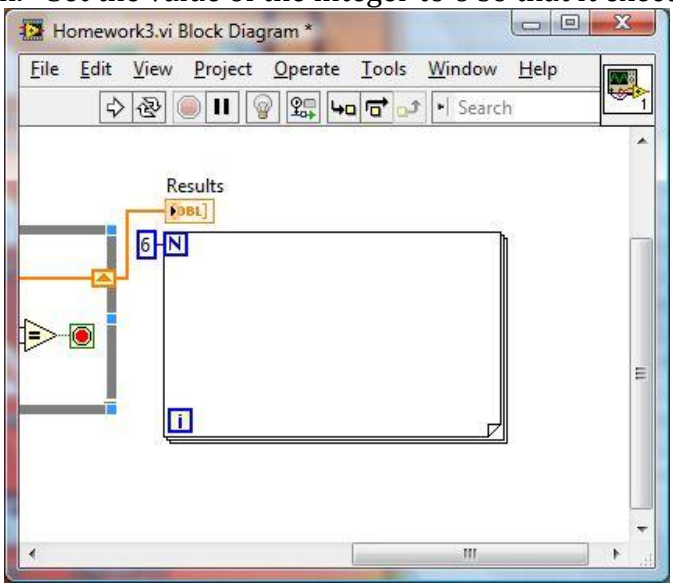
g. Ensure the array has six elements. This will stop adding results to the lotto results array once 6 numbers have been selected.
    i. Place an Array size (Functions> Programming> Array> Array Size) function after the case structure
    ii. Connect the input of the Array Size function to the array passed out of the Case Structure
    iii. Place an Equal? function with one input connected to the array size
    iv. Create a constant with a value of 6 and connect it to the other side of the Equal? function
    v. Connect the output of the Equal? function to the stop terminal of the While loop



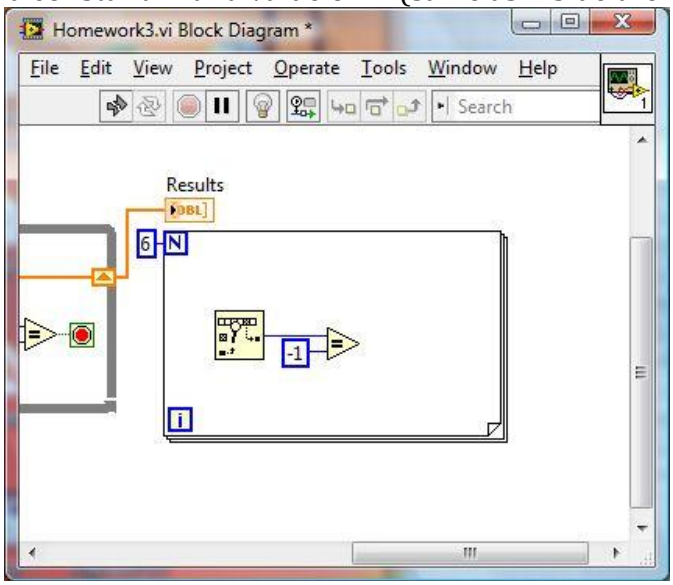
h. Connect the output of the shift register to the “Results” Array in order to display the results to the user.



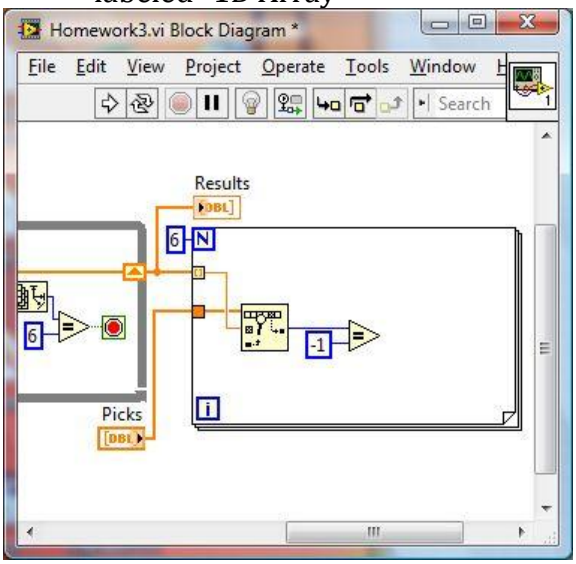
3) Create a For Loop to determine the number of matches between the lotto picks and the random numbers. We will iterate through the array of lotto results and see if each number if found in the array of the user’s lotto picks.
   a. Create a For Loop (Functions> Programming> Structures> For Loop)
      i. Right click the input to the count terminal (N) and select Create Constant
      ii. Set the value of the integer to 6 so that it executes 6 times



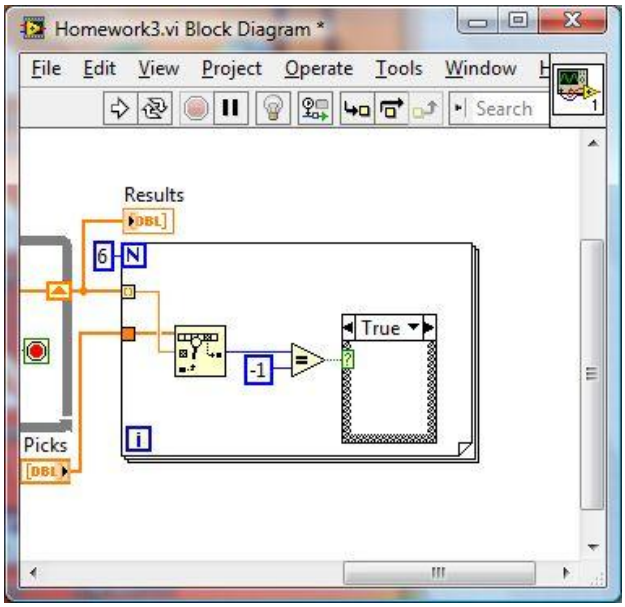
b. Place an Equal? Function with inputs of a Search 1D Array function and a constant with a value of -1 (same as inside the While Loop)



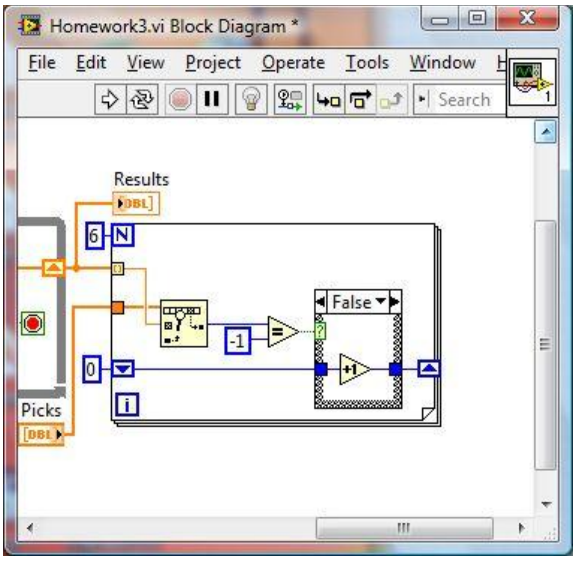
c. Wire the inputs of the Search 1D Array
   i. Connect the output of the shift register on the While Loop to the input of the Search 1D Array labeled "element". This will enable auto-indexing on this array, so that the for loop iterates through each of its elements.
   ii. Connect the output of the Array labeled "Picks" to the left side of the For Loop
      - Disable Auto-Indexing (Right click the tunnel and select Disable Indexing) so that the whole array is passed in
      - Connect the tunnel to the input of the Search 1D Array labeled "1D Array"



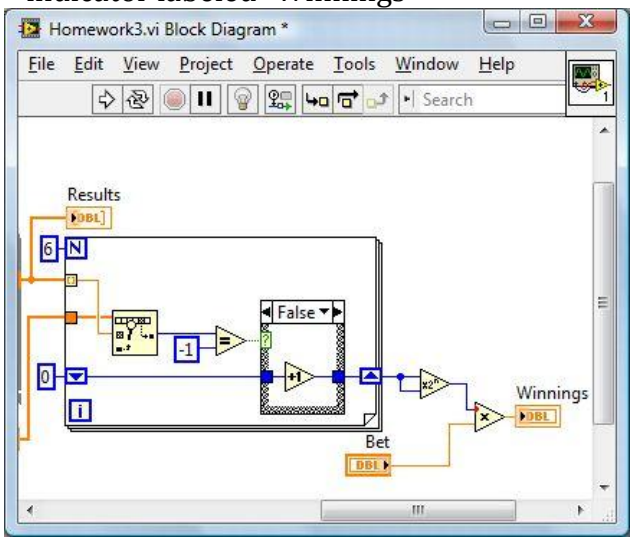
d. Place a Case Structure inside the For Loop
    i. Connect the output of the Equal? Function to the Case selector terminal



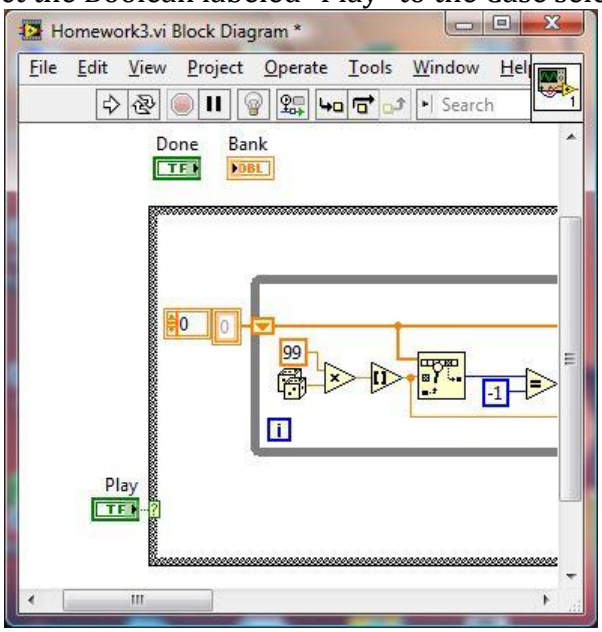
e. Count the number of user's picks that match the lotto results. If the Search 1D Array returns a -1 (True in case structure), then that means that the lotto result was not found in the user's picks, therefore they did not match that number. If the Search 1D Array returns a value other than -1 (False in case structure), it means that the lotto result was found in the user's picks.
    i. Place a Increment function (Functions> Programming> Numeric> Increment) inside the False case
    ii. Connect the output of the increment function to the right side of the For Loop
    iii. Right click and replace the Auto-Indexing tunnel with a Shift Register
    iv. Connect the output of the shift register on the left side with the input of the increment function
    v. In the true case of the Case Structure, wire the tunnels together
    vi. On the outside of the For Loop, right click the input of the shift register and select create constant, leave the value with the default, 0



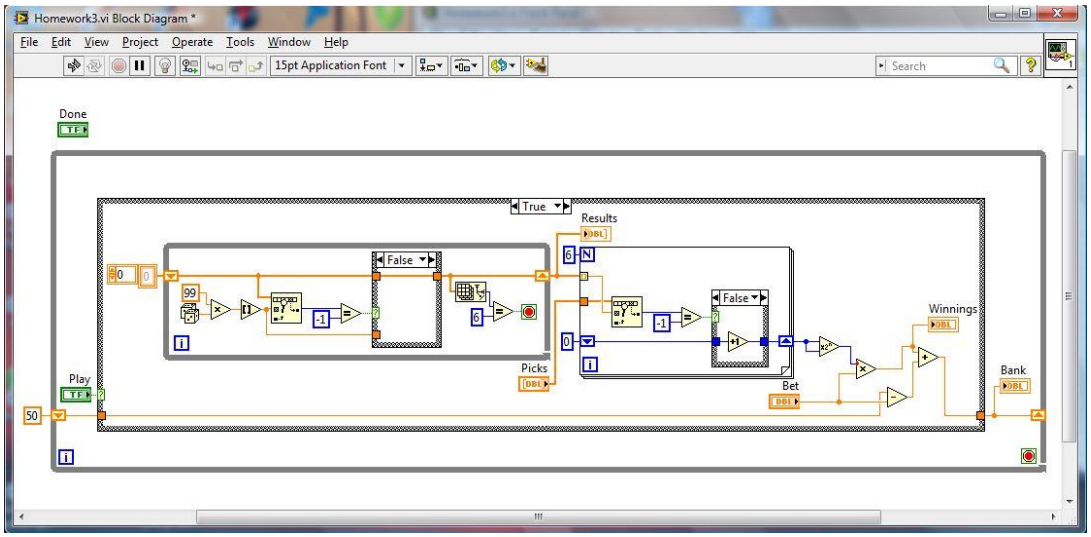
f. Determine the winnings
   i. Wire the output of the shift register on the right side of the For Loop to both inputs of a scale by two function (Programming> Numeric> Scale by Power of Two)
   ii. Wire the output of the Scale by Power of Two to a Multiply function
      - The other input of the multiply is the Numeric control labeled “Bet”
      - This logic assures that the winnings are exponential to the number of picks the user gets correct.
   iii. Wire the output of the multiply function into the Numeric indicator labeled “Winnings”



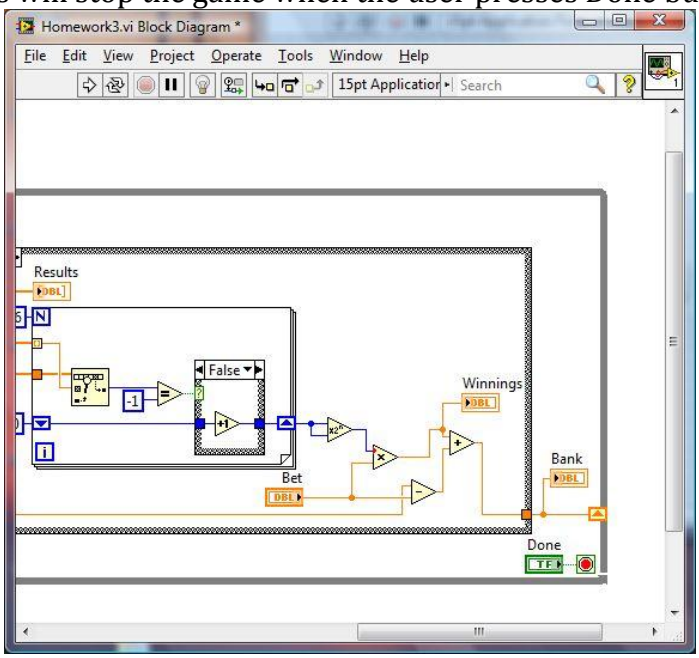
4) Determine amount in “Bank”
    a. Place Case Structure over both the For and While Loops
        i. The loops should be in the “True” case. If they are not, right click the boundary of the Case Structure and select “Make this Case True”
    b. Connect the Boolean labeled “Play” to the Case selector terminal



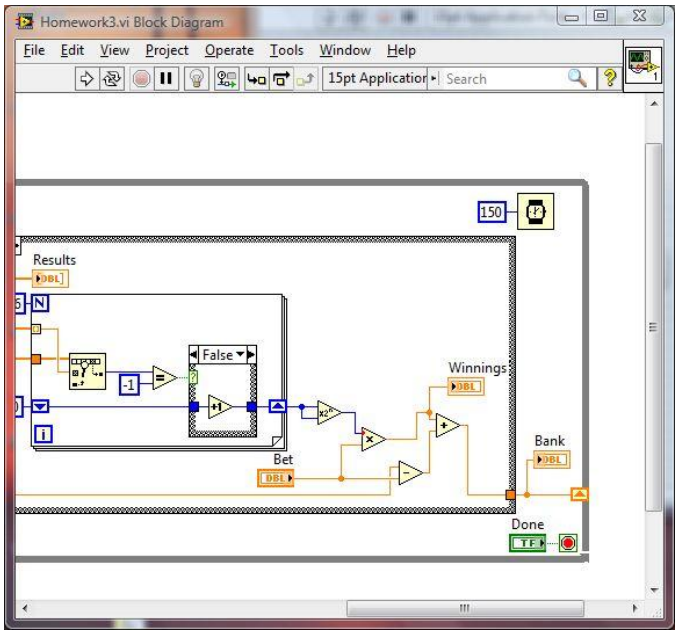
    c. Place a While Loop around the Case structure (and the Play Boolean)
    d. Create a Shift Register which contains the value of the “Bank”
        i. Create a Numeric constant on the outside of the While Loop with an initial value of 50
        ii. Wire the constant to the left side of the Case Structure
        iii. In the true case, starting from the constant, subtract “Bet” from it, and add “Winnings” to that result using the Subtract and Add functions.
        iv. Wire the output of the Add function to the outside of the While Loop
        v. Replace the tunnel with a Shift Register (connected to the tunnel wired to the Numeric Constant with a value of 50)
        vi. Wire the Numeric Indicator labeled “Bank” to the output of the Case structure
        vii. On the False case, connect the bank tunnels together
            - This is done so that the value of the bank is not lost during an iteration when the case is false



5) Wire the Boolean Control “Done” to the stop terminal of the outside While Loop. This will stop the game when the user presses Done button.



6) Place a Wait function in the outside While Loop. This is to free up time in the CPU.
    a. Create a numeric constant with a value of 150 wired to the input



Optional add-ons:

- Customize the controls so that it looks more like a lottery
- Interact with the user if they get above or below a predetermined amount in their bank
- Document your code with tip strips and free labels
- Set up your own payout structure
- Don't allow the user to guess the same number multiple times