COP2010 Project #3
50 pts – Due by start of class on 11/10

You are going to create an application that will generate random numbers and store data in an array. There will be text boxes to enter numbers for lower and upper bound and a text box to enter how many numbers to generate. When the application runs it will generate the specified number of random numbers and store the count in the array. You do not save the numbers - just the counts of how many times each number is generated.

Objectives:
- Use various iteration structures
- Use random number generation
- Use a single dimension array
- Practice calling procedures and event handlers

Requirements

Read the projects and assignments requirements pdf in the course documents section of canvas. You need to follow all of the guidelines for this project.

1. Text boxes for lower bound, upper bound, and number to generate.
2. Validate data entry and show a messagebox if requirements below are not met
   a. Must use a method(s) outside the event handler to validate
   b. Lower bound must be less than or equal to the upper bound (can be either negative or positive numbers.)
   c. Number to generate must be great than or equal to 1.
3. Exception handling if non-numbers are entered. Display messagebox with error message.
4. The program will generate random numbers within the entered range. Example: if 5 and 10 are entered as lower and upper bounds the random numbers generated would include 5, 10, and all numbers between.
5. Each time a number is generated it will add to a count kept in an array. Example: with a range of 5-10, when a 5 is generated it will add 1 to index 0 of the array. If a 6 is generated it will add 1 to index 1 of the array.
6. If a negative number is entered for lower or upper bound it should still run as long as lower is less than or equal to the upper.
7. Output to a listbox the range of numbers and counts for each number.
8. The form will have a menu that includes:
   a. File with option to Exit
   b. Calculate with options to generate and another option to reset
   c. Access keys for menu items e.g. Exit
9. A reset button that clears only generate text box and the listbox – lower and upper bound are not cleared. It should also place focus on generate.
10. Have an exit button to exit the application.

Notes: Use as many variables as you like but all must be named using camel case and with the required naming convention and methods must be named using pascal case.

Example form: not required to match layout, but must have all text boxes, buttons, labels, etc.

Random Number Generator

File   Caclulate

Lower Bound   [   ]   [Generate Numbers]

Upper Bound   [   ]   [Reset   ]

Number to Generate [   ]   [Exit   ]

[   ]

Random Number Generator

File   Caclulate

Lower Bound   [5   ]   [Generate Numbers]

Upper Bound   [10   ]   [Reset   ]
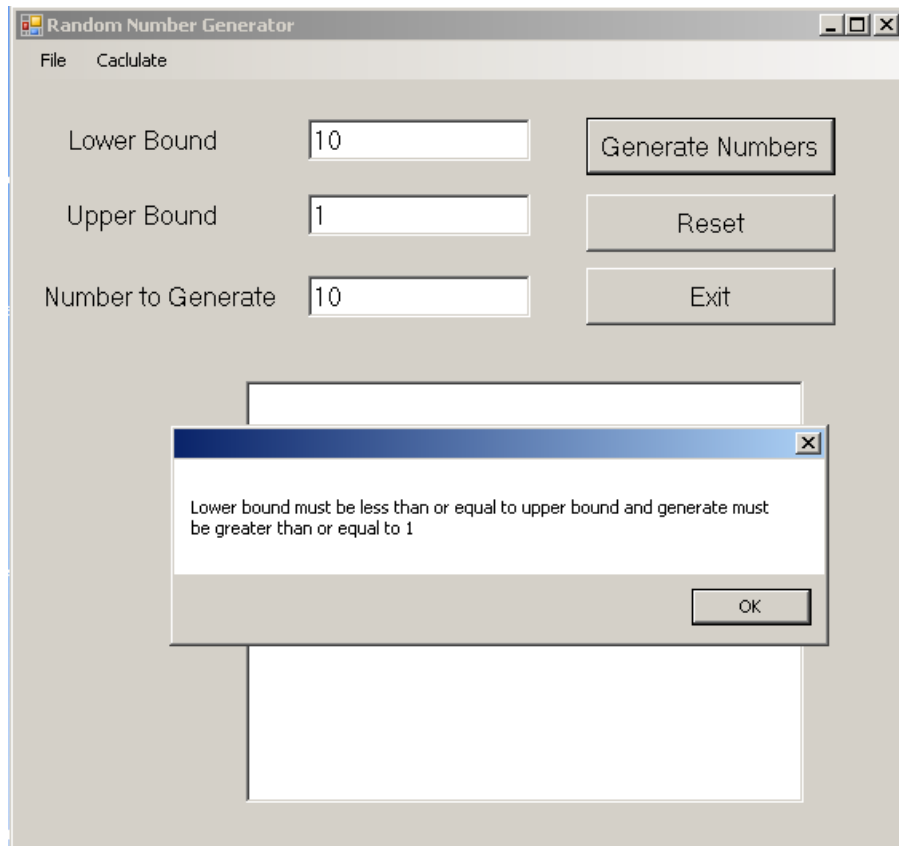
Number to Generate [10   ]   [Exit   ]

```
The number 5 was picked 3 times
The number 6 was picked 1 times
The number 7 was picked 1 times
The number 8 was picked 2 times
The number 9 was picked 1 times
The number 10 was picked 2 times
```

Grading Rubric (possible 50 pints):

A submitted project must compile and run or it will be graded with zero points

Description	Itemized Value	Total Possible	Total Earned
Correct output for array to listbox for number range and counts.	7 pts each for range & count	14	
Data validation method(s) for generate, upper, and lower bound. Must check for Lower <= Upper and generate >=0.	4 pts each box	12	
Menu items with access keys for exit, generate, and reset.	2 point each	6	
Exception handling if non-number is entered	All or nothing	4	
Correct naming of all event handlers and variables	All or nothing	5	
Proper comments and documentation	All or nothing	5	
Reset button clears and sets focus to the generate text box	All or nothing	2	
Exit button closes application	All or nothing	2	