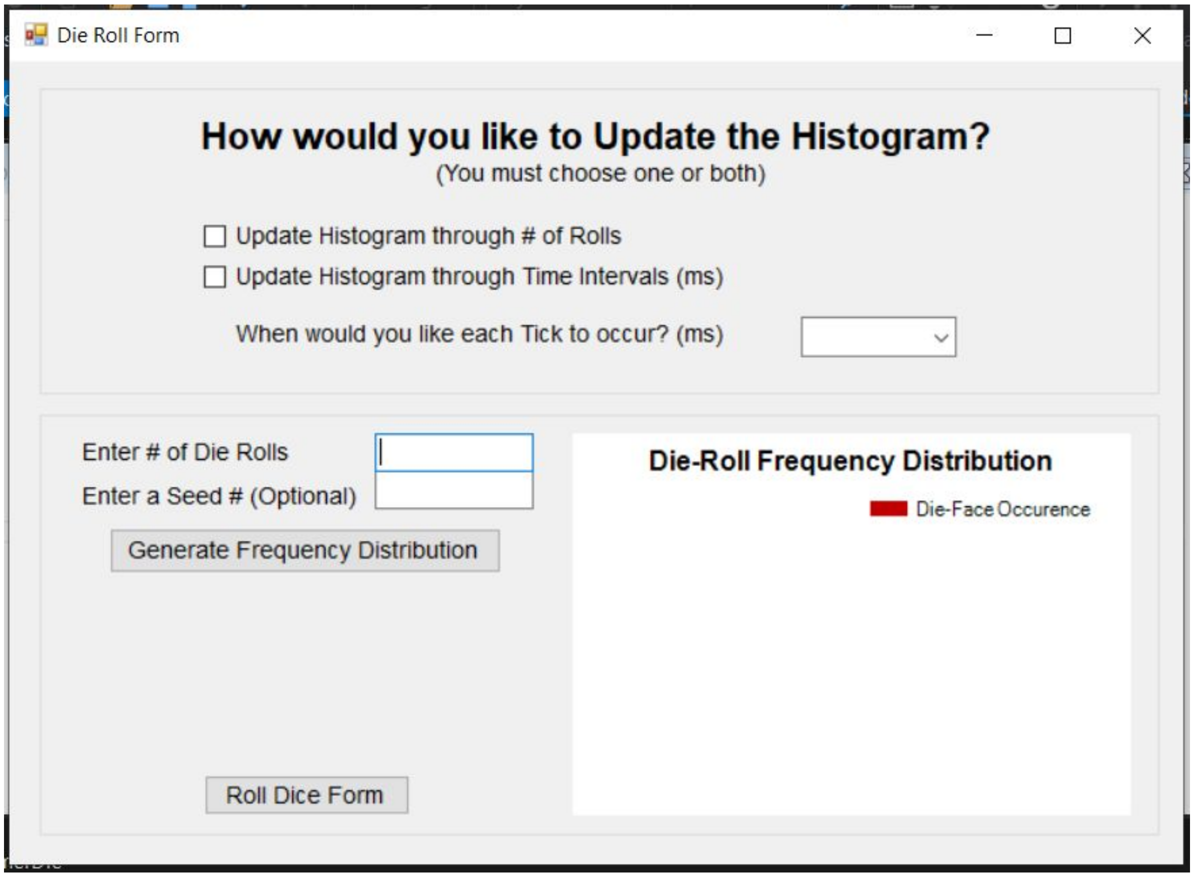
COP 4365
Name: Thi Vo

Project 2 Report

For this project, I was tasked with updating two forms in C# simulating die rolls. It was a good experience for me to enhance my VS 2019 and C# skills through more controls and Object-Oriented Programming.

The first form involves rolling one die a given number of times from the user, while the second form involves rolling a pair of dice a given number of times by the user. How the first form looks after the program starts is shown below.


Die Roll Form

How would you like to Update the Histogram?
(You must choose one or both)

☐ Update Histogram through # of Rolls
☐ Update Histogram through Time Intervals (ms)

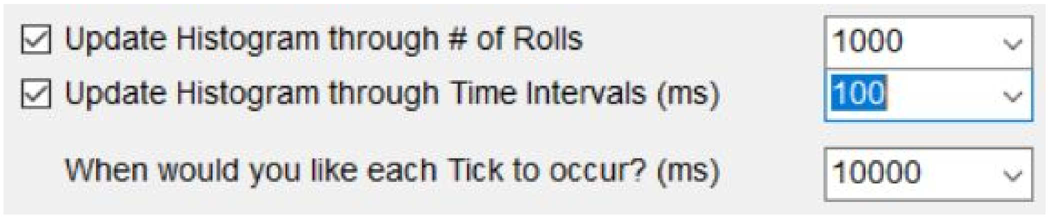
When would you like each Tick to occur? (ms)

Enter # of Die Rolls
Enter a Seed # (Optional)

Generate Frequency Distribution
Roll Dice Form
Die-Roll Frequency Distribution
🟥 Die-Face Occurence

The specifications of project #1 are highlighted in the lower group box (the bottom half of the form) in the DieRollForm and DiceRollForm.

The top group box highlights the essential options of this project. The user must select a way to update the histogram as well as the interval of the timer ticks. Either (or both) checkboxes for updating the histogram must be selected. Once a checkbox is selected, their respective combo box appears as shown below. Each form's timer keeps it's tick counts based on the users tick interval selection.



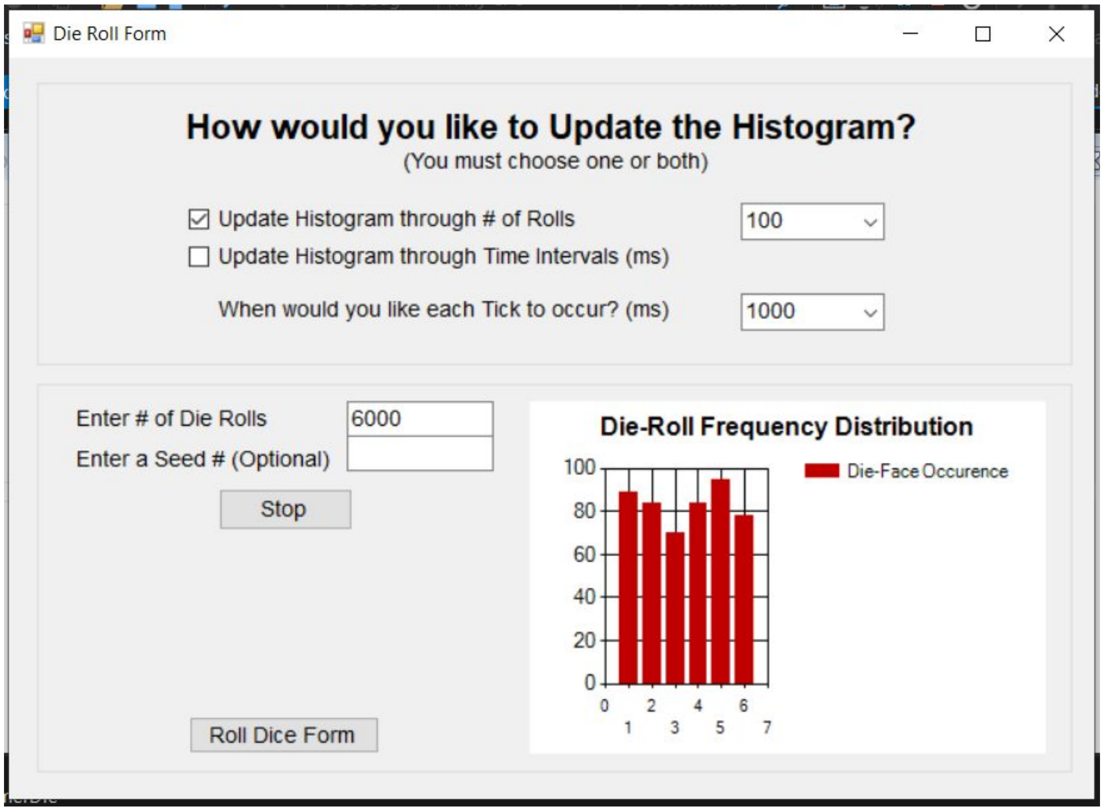
☑ Update Histogram through # of Rolls — 1000

☑ Update Histogram through Time Intervals (ms) — 100

When would you like each Tick to occur? (ms) — 10000

You can make various selections via all three combo boxes to show real continuous updates to the frequency distribution. The tick interval must be less than or equal to the update time intervals.

After entering valid input for the top group box and the number of die/dice rolls, the user can click the "Generate Frequency Distribution" button to continuously populate the chart on the right side as shown below.



Die Roll Form

How would you like to Update the Histogram?
(You must choose one or both)

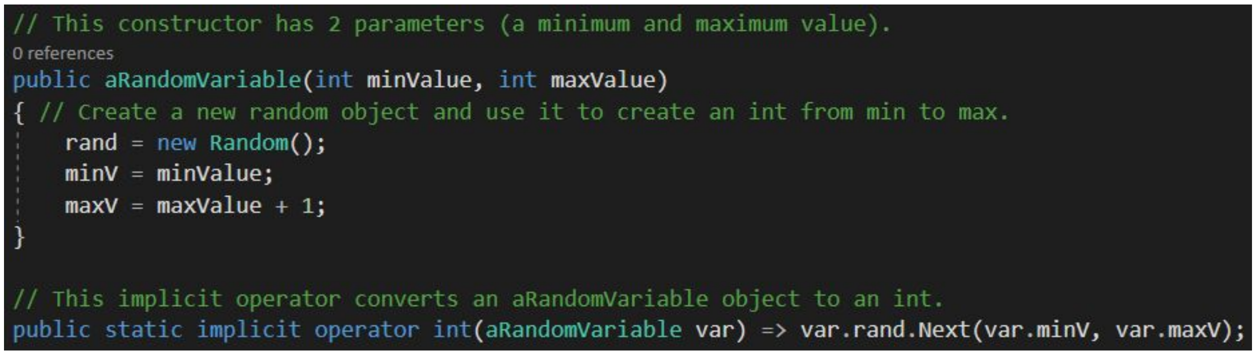
- [✓] Update Histogram through # of Rolls: 100
- [ ] Update Histogram through Time Intervals (ms)

When would you like each Tick to occur? (ms): 1000

Enter # of Die Rolls: 6000
Enter a Seed # (Optional):

Currently, updates happen every 1000 ms, which is the tick. 100 random Y values are plotted to the graph each second for all 6000 die rolls (this is 1 minute total of 60 updates). Next, let's say the tick interval is 1000 ms and the update time interval is 10000ms. There is an update each second and there would be 10 rolls per update (10000 / 1000 = 10). The 'Generate Frequency Distribution' becomes a 'Stop' button if you want to stop the continuous plotting and restart with an empty graph. You can press the 'Generate Frequency Distribution' button again after this or when the chart finishes graphing.

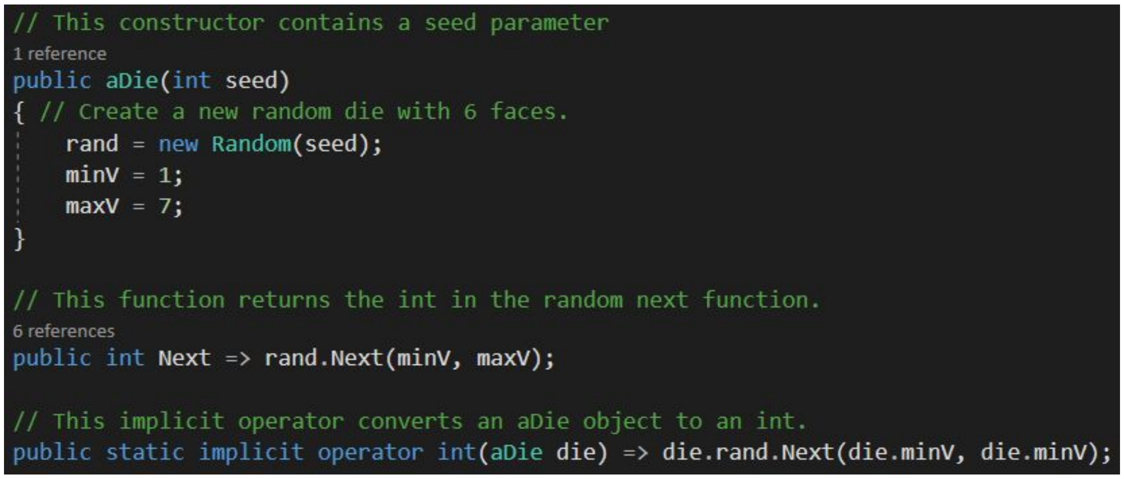
Finally, there were 2 new classes that were created: the aRandomVariable class and the aDie class. The aRandomVariable class has 2 constructors. Both constructors create the Random() object and set minimum and maximum values for the implicit int operator to use as shown below. The first constructor sets default min and max values while the second constructor uses 2 int parameters for those values.



```
// This constructor has 2 parameters (a minimum and maximum value).
0 references
public aRandomVariable(int minValue, int maxValue)
{ // Create a new random object and use it to create an int from min to max.
    rand = new Random();
    minV = minValue;
    maxV = maxValue + 1;
}

// This implicit operator converts an aRandomVariable object to an int.
public static implicit operator int(aRandomVariable var) => var.rand.Next(var.minV, var.maxV);
```

The aDie class has 2 constructors and inherits from the aRandomVariable class. They behave the same as the aRandomVariable constructors. The difference between them is that the second constructor uses one int parameter for the seed value to be used in the Random object as shown below. The next function for this class behaves like the Random.Next() function with the given min and max values.



```
// This constructor contains a seed parameter
1 reference
public aDie(int seed)
{ // Create a new random die with 6 faces.
    rand = new Random(seed);
    minV = 1;
    maxV = 7;
}

// This function returns the int in the random next function.
6 references
public int Next => rand.Next(minV, maxV);

// This implicit operator converts an aDie object to an int.
public static implicit operator int(aDie die) => die.rand.Next(die.minV, die.minV);
```