Criterion C: Development

1. Enum for Periodic Table
2. Data Collection and Exception Handling
3. Storage in a 2D Array
4. Calculation Methods
5. Displaying Results
6. Displaying Steps to get to results.

1. I initially thought that there was a readily available periodic table class or package that I could import into my project, but research proved there was no readily available Periodic Table class that I could import into my project. As a result, I attempted to use a package to hold files for every element in the periodic table, but the estimated storage for that would make in infeasible to do so. I did research while coding my own enum and found one on GitHub by Felix Divo that had all the elements and molar masses available, saving me much needed time in the coding process.
2. In the program, the information is taken in from the user. One of the greatest problems associated with this is the number of elements in each formula varying. When tasked with solving this problem, I could either leave JTextFields empty and ignore them in solving, but I figured out how to use a JComboBox to give the user only select options for the number of elements. After this, there is a problem with the user entering an atomic symbol that is incorrect or numbers for each percent that do not add up to 100. I decided to throw exceptions if these where the case when the ActionListener in the calculate button is pressed to avoid these and only allow equations that can be solved to be entered.
3. For doing the steps to get to the result, I needed to have a way to not only get the result but also have the steps to get to it for displaying in a later stage. I chose to do this with the 2D array in Java to store the steps needed to get the result.
4. I needed methods to do the step by step calculations when calculating the Empirical Formula to not only calculate the result but populate the arrays at the same time. Since the methods in the enum were static, I used a set of static methods created in the FormulaCollect class to prevent the need for initializing an instance of a class full of only methods.
5. For displaying results, I needed to do the number of each element in a subscript and found that this could be easily done using html since it is supported in JLabels. I figured out how to loop through the result and populate the values at the same time in the html string by appending it.
6. For displaying the results, I was trying to decide how to print the steps of the array in a regular pattern across the string and came across JTable which could do this for me. I opted to use this for displaying the steps to get to the result for its simplicity.