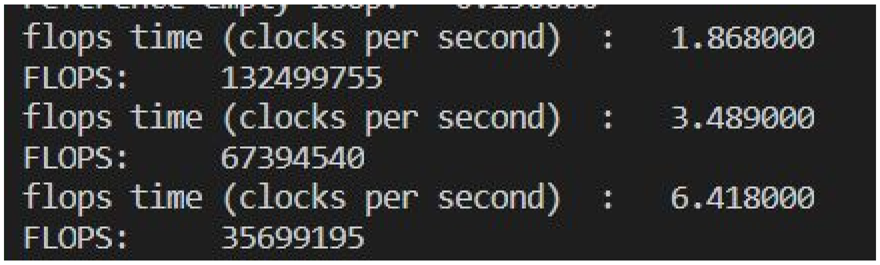
Jeremy Venne
Algorithms
Q2



```
flops time (clocks per second)  :   1.868000
FLOPS:     132499755
flops time (clocks per second)  :   3.489000
FLOPS:     67394540
flops time (clocks per second)  :   6.418000
FLOPS:     35699195
```

A recursive function has a base case, or the condition that terminates the recursive function. In getNumberEqual, this correlates to the check of n being less than or equal to zero. Since n corresponds to the size of the array, it makes sure every element has been checked. Since it is subtracted by one each recursive loop, it is equal to zero after the last element of the array is checked. A recursive function also breaks a problem into subproblems in order to reach the base case. As described, n is subtracted until it reaches the base case. As this is occuring, the count of the element of the array is tracked. Therefore, the individual counts can be combined at the end to get the total number of occurrences in the array. This clearly shows that the function represents a recursive method.

Since every element of the array is accessed once, this means this function has a time complexity of O(n) = n. This is supported by the screenshot above. This is the clocks per second and the FLOPS of the algorithm for three different arrays. These arrays have size 3, 6, and 12. Clearly, the array size is increasing linearly. Similarly, the FLOPS are decreasing linearly while the clocks per second are increasing linearly.