As you observe the timings for Hanoi, what do you notice? Is this a performance class we have seen before or a different one? How would you describe the rate at which the runtime of Hanoi is increasing? No formal analysis is necessary here; just describe what you see and what is causing it. Including Delimiter_Check, we have now seen several examples of python programs that accept arguments/parameters when they are invoked on the command line. Describe how you could employ this together with factory method infrastructure provided by Deque_Generator, to allow a deque, stack or queue program to use either an array-based deque or linked list-based deque depending how it is invoked from the command line. Finally, describe the process through which you tested Hanoi and Delimiter_Check to ensure that they function correctly.

The performance of Hanoi is exponential, O(2^n) which we have not seen before in previous projects. The number of steps can be computed by -- the number of steps of n-1 + 2^n) – this is likely due to the recursion. Each new step involves doing all the steps of n-1 and additional steps of 2^n.

We can make a constructor that is similar to the Deque_Generator

```
Linked_List = 0

Array = 1

def __init__(self, type):

        self.type = type

        self.__dq = get_deque(type)
```

This way we can change which type of stack, queue, or deque we want, whether it be array or linked list by choosing 0 or 1.

For testing Delimiter_Check, I made a few files where some were supposed to be correct, and some were supposed to cause errors and ran it through the program. The results returned were exactly what I was looking for.

For testing Towers of Hanoi, I recorded the average time it took to run n=1, n=2, n=3, n=4, n=5 and compared them. My computer is pretty slow so the timing was inconsistent, but after multiple trials it was clear that the performance was, for the most part, exponential. I also counted the steps printed and it was clearly exponential. Additionally, I followed through the printed steps of Hanoi and it matched what I was expected (for example n=3 was identical to the described result in the project instruction)