Hanoi Performance:

The performance of Hanoi is not something we have seen before. I would describe it as exponential, this would make sense as the most efficient way to solve the game in real life takes (2^n) -1 steps, where n is the number of discs. I think this is caused by the fact that you have two recursive calls in the function.

Command Line Arguments:

An argument in the command line when running a program could allow for choice of using either an array deque or linked list deque when using deques in either stack, queue, or deque programs. The parameter accepted in the command line could determine which type of deque is chosen in the get_deque function. A possible benefit of this is that when working with large set of data, a data structure based off a linked lists is preferable to that of one based off an array because the cells do not have to be next to each other in a linked list.

Testing:

Testing for Hanoi consisted of two different things. Making sure the output matched the example exactly and making sure for bigger n's no rules were broken. This consisted of matching the output for n=3 and examining the output for larger n's like 5 to make sure no bigger discs were placed on smaller ones.

Delimiter Check testing was a bit more rigorous. For delimiter check there are three things that need to be checked. First that it returns true for balanced delimiters. I did this by testing two files, one very simple and one much more complex. The second thing to check is that it returns false for imbalanced delimiters. However, there are two things to check here as delimiters can be imbalanced for two reasons. Either the stack was not empty or there was an incorrect open close pairing. I again did this by checking one simple file and one more complex file for each.