List

Array	front_insert	back_insert	traverse	front_delete	back_delete
Length = 5	0:00:00.000923	0:00:00.000926	0:00:00.000895	0:00:00.000923	0:00:00.000915
Length = 10	0:00:00.000991	0:00:00.001008	0:00:00.000907	0:00:00.000955	0:00:00.000972
Length = 15	0:00:00.001121	0:00:00.001115	0:00:00.000975	0:00:00.001051	0:00:00.001066

Pointer	front_insert	back_insert	traverse	front_delete	back_delete
Length = 5	0:00:00.001049	0:00:00.001007	0:00:00.000922	0:00:00.000955	0:00:00.000948
Length = 10	0:00:00.001198	0:00:00.001148	0:00:00.001036	0:00:00.001099	0:00:00.001116
Length = 15	0:00:00.001491	0:00:00.001502	0:00:00.001253	0:00:00.001427	0:00:00.001465

Stack

Array	Iter_pop	Iter_push
Length = 5	0:00:00.000917	0:00:00.000936
Length = 10	0:00:00.000945	0:00:00.001004
Length = 15	0:00:00.000984	0:00:00.001130

Pointer	Iter_pop	Iter_push
Length = 5	0:00:00.000898	0:00:00.000921
Length = 10	0:00:00.000968	0:00:00.000987
Length = 15	0:00:00.001007	0:00:00.001123

Conclusion:

Overall, it can be seen that the execution time was dependent on the size of the input data structure. As the length went up so did the execution time. This was the case for every implementation. Where the results get a bit interesting is what the functions were implemented on: stacks or lists. For the upper 2 tables which represent the lists the pointer implementation had a noticeable lag compared to the arrays. This is most likely due to the arrays ability for random access the pointers required going through nearly every element in sequence for most implementations. In the stack implementation random access is no longer an issue as a stack is LIFO (last in first out) so elements can only really be inputted and retrieve in one way. The way the pointer was implemented for this Python implementation favored that as a special class was written to hold the current element data and the following data as well. This project would have most likely been a bit easier in a C++ implementation since pointers do not really exist in Python and the overall execution times better as Python is an interpreted language.