Part A: contains()

Container/ Amount	Vector	List	Hash	Table
1000	0	2.05 * 10^-7	1.22 * 10^-4	7.60 * 10^-3
2000	2.10 * 10^-7	0	8.42 * 10^-5	1.63 * 10^-2
3000	0	0	1.11 * 10^-4	1.63 * 10^-2
4000	2.05 * 10^-7	2.05 * 10^-7	1.49 * 10^-4	3.68 * 10^-2

Part B: query (name & weight)

Container	Vector	List	Hash	Table
1000	5.68 * 10^-2	5.75 * 10^-2	5.87 * 10^-2	5.58 * 10^-2
2000	9.85 * 10^-2	1.01 * 10^-1	1.02 * 10^-1	9.60 * 10^-2
3000	2.43 * 10^-1	2.47 * 10^-1	2.53 * 10^-1	2.50 * 10^-1
4000	6.06 * 10^-1	6.13 * 10^-1	6.25 * 10^-1	6.16 * 10^-1

Container	Vector	List	Hash	Table
1000	4.85 * 10^-3	7.89 * 10^-3	7.57 * 10^-3	6.05 * 10^-3
2000	7.47 * 10^-3	1.44 * 10^-2	1.43 * 10^-2	1.42 * 10^-2
3000	1.87 * 10^-2	3.21 * 10^-2	3.19 * 10^-2	2.40 * 10^-2
4000	3.35 * 10^-2	6.18 * 10^-2	6.43 * 10^-2	4.55 * 10^-2

Analysis

My Tree-based query didn't pass the time requirement on Gradescope, so that will affect my response. I am surprised that the AVL tree has such long times, especially for the query. The fact that the Hash-based inventory performed similarity to the list for query. This shows me that the list is an inefficient data time to iterate and search through because Hash-based containers aren't made for searches to the value stored, instead it exceeds for lookups. The hash function also outperformed the query when based on name comparison, which is also a mark against a Tree-based inventory. Overall, I would pick a Hash-based inventory because overall it performs better than Tree-based. As noted in my first sentence, my choice could be changed if I was able to write my Tree-based query more efficiently.