This project focuses on aspects of optimizing and measuring performance.

- The first problem provides a baseline implementation for a function and asks for a more performant version which runs faster. To do so, one must exploit knowledge of the memory hierarchy and processor pipeline.
- The second problem provides a series of functions for classic search algorithms and requires implementation of a main() function to benchmark the performance of those functions.

The first object was to make a matrix multiplication function as fast as possible, if you were able todo this a Sonic the Hedge Hog would pop up in the terminal. I first attempted to this by looking at the mathematical way to speed up the function, but untimetnly went in a different direction. I ended up making 4 versions of the function to try and max out my speed. My best solution focused on optimizing the function via, using memset only once and gather data in larger portions that smaller. This utilized the stack as much as possible instead of using the cashe. By using the stack instead of the cashe I was able to get a raw score of 42, above the 35 expected points, I also did get the Sonic which was Awesome!!

The second object was to compare the differentes between a multitude of different data structures. These 4 data structures were arrays, lists, binary, and trees. We provided extensive data to explain the difference between these 4 data structures, and some graphs and explanations to go with it.

Length	array	list	binary	tree
2	1.00	1.06	1.59	2.00
4	1.00	1.10	1.17	1.04
8	1.63	1.69	1.30	1.00
16	1.68	1.68	1.50	1.00
32	1.71	1.67	1.30	1.00
64	2.14	2.33	1.15	1.00
128	4.53	4.58	1.15	1.00
256	9.33	8.64	1.16	1.00
512	16.88	20.19	1.19	1.00
1024	26.33	90.24	1.00	1.03
Average				
	6.62	13.32	1.25	1.11