The two different types of searching algorithms are linear and binary. An algorithm is a series of events that are taken to accomplish a task (Lysecky, Vahid, Lysecky, & Givargis, 2015). A linear search algorithm searches a list of values consecutively from the beginning to the end. One drawback to this algorithm is the runtime. The amount of time the algorithm takes to execute is the runtime (Lysecky et al., 2015, sect. 1.1). If an array has a large number of values, it will have a longer runtime than an array with only a few elements. For example, if you are doing a linear search for titles in a library, the database will search for the name by starting at the beginning and iterating through each title until it finds the correct one.

Binary searches have a faster runtime when compared to linear searches. The elements of an array must be sorted to perform a binary search. In the above example, the titles would first have to be sorted alphabetically. A binary search will split the list in half and see whether the title came before or after that point. It continues this process until it finds the element.

Along with searching through elements, you can also sort these elements. According to Shaffer (2013), there are three sorting algorithms, selection, insertion, and bubble. Selection sorts work by selecting the smallest value and putting it first and continuing this process until all the values are in the correct order. An insertion sort “iterates through a list of records” (Schaffer, 2013, p. 225). Insertion sorts take values and insert them into the correct position until all values all sorted. Bubble sorts iterate through all the values and push lower values to the top. The bubble sorting algorithm is the slowest when it comes to runtime.

A factor in which type of algorithm is used is the time and space complexity. Time complexity is determined by the size of the information, which will determine how many steps are needed to complete the task while space complexity is how much storage is required for the size of the values (Sorting, searching, and algorithm analysis, 2014). For the average person, space, and time complexity would not matter. Regardless of which algorithm is executed, a person would not be able to decipher if one algorithm was milliseconds, nanoseconds, etc. faster then another algorithm. When it comes to a supercomputer, it does matter which algorithm is faster, as they need to complete their tasks as quickly as possible, so milliseconds and nanoseconds would matter.

**References**

Lysecky, R., Vahid, F., Lysecky, S., & Givargis, T. (2015). Data structures essentials. Retrieved from <https://zybooks.zyante.com/#/zybook/DataStructuresEssentials>

Shaffer, C. A. (2013). Data structures and algorithm analysis (3.2 ed.). Retrieved from <http://people.cs.vt.edu/~shaffer/Book/JAVA3elatest.pdf>

Sorting, searching and algorithm analysis. (2014). Retrieved from http://python-textbok.readthedocs.io/en/latest/Sorting\_and\_Searching\_Algorithms.html