Joshua Gordon

Time Complexity Analysis

Time complexity is something that is always regarded when trying to make an efficient algorithm. Without algorithms being time efficient, many things would take exponentially longer to complete. When conducting my experiment using the Bubble and Selection Sorts; I was able to see firsthand just how exponential this gap in time could be when regarding efficiency.

The Bubble Sort algorithm proved itself to be much slower than its counterpart, the Selection Sort. When comparing small numbers like 1 and 10 (as shown in my first two examples of the program) we see that the difference in time is very minimal. However, once we start to calculate arbitrary large numbers, this difference in time becomes much more extreme. Just from using the highest number stated in the project – 5000, there was a three second difference in time. These numbers would vary from OS and the computer being used to run the program of course, but it can be argued that a difference would be noticed. For my experiment I used a 2015 MacBook Pro 13-inch with a 2.7 Ghz Dual-Core Intel i-5 processor. Although not the newest or most up to date in terms of specs, it’s obvious that these sorts of algorithms are necessary regardless of the hardware and software equipped. The fact that my 6-year-old computer can still maintain efficiency just based on the algorithm that is used is very powerful.

We determine that an algorithm is efficient if it runs faster. That’s it. No other outside factors are considered. Even though both sorts use Big O Notation, one is still faster than the other and that is entirely due to how the algorithm sorts through the information it is given. One doesn’t make use of the keywords like >, <, =, ==, +, -, etc. That is not all that makes an algorithm; however, these were the main components of what made these two, and by logic alone, sorting through one by using those keywords more logically is what gave us better timing.