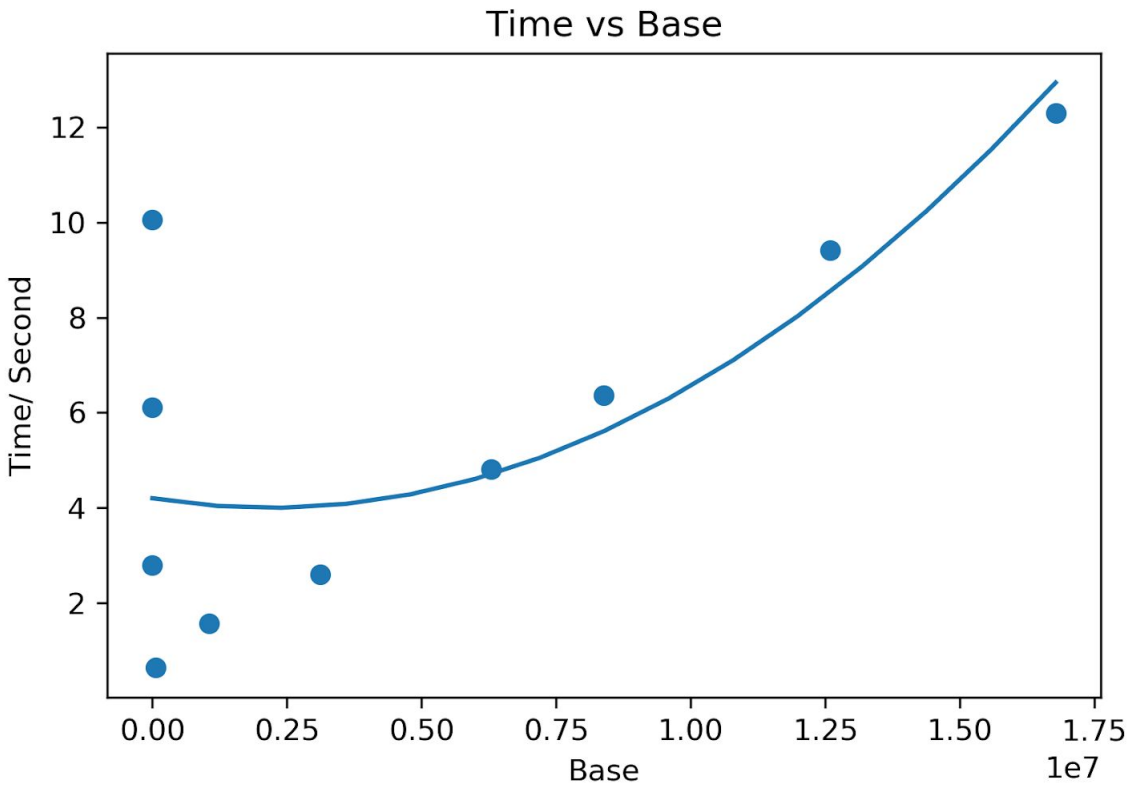
Task 2- Data Analysis



Data Collection:

Base	**Simplification (if any)**	**Time Taken (Second)**
2	2^1	9.9533965999980860
3		6.0817414000011920
10		2.8725971999992908
65536	2^16	0.6348494000012579
1048576	2^20	1.5049007000015990
3123141		2.4960748000012245
6291461		4.5003235000003770
8388608	2**23	5.6508916999991925
12582917		8.2908277000024100
16777216	2**24	11.088056500000675

Explanation:

Selection of bases:

Base 2 is chosen to be one of the values to be tested because it is the smallest possible number for a base. This implies that only 2 “containers” will be created in the “counting_sort” algorithm. However, due to the smaller base, the number of digits required to represent the number will be relatively longer. This is because only 1 and 0 are available to represent the value. Hence, by using a base of 2, we are able to observe the processing speed with minimal “containers” and maximum length of digits.

Base 3 is chosen as a value between 2 to 10 due to a large time gap that exists between computing base 2 and base 10. This helps with the observation of the trend with smaller bases.

Base 10 is chosen to be tested because it is one of the most common bases we use in everyday life.

A series of numbers with a base 2 and the exponent of 16, 20, 23 and 24 are also chosen to be tested. A series of numbers with base 2 are chosen because computers use binary for calculation and processing. However, as bitwise operations are not implemented in the code, no significant improvement in processing time can be observed.

Random large numbers such as 3123141 and 6291461 were also tested as bases. The number 16777216 is tested as it is a number close to the largest number that can be handled by my computer system. This number reflects the processing time for cases that have an unreasonably high number of “containers” created in the “counting_sort”.

Time complexity

The radix sort involves the implementation of counting sort that has a complexity of O(n+b).
Where n is the size of the input list and b is the base used to perform counting sorts on.
The radix sort utilises ‘m’ iterations of stable counting sorts to perform its sorting, where ‘m’ is varied according to the magnitude of the number in the list. To be precise, the number of digits ‘m’ is based on $\log_b M$, where M is the maximum value in the list and b is the base. Together, the total time complexity of the radix sort algorithm is O((n+b)m), where n is the size of the list, b is the base and m is the length of the integer (“columns” to perform counting sorts on).

According to the scatter plot, the general trend of the scatter plot is a decreasing trend of processing time (between base 2 to base 2^16) followed by an increasing trend (between base 2^16 to 2^24) with a turning point at around base 2^16. As can be seen from the scatter plot, the radix sort performed for smaller bases (base 2 & 3) resulted in relatively longer processing times. This is because the number of digits ‘m’ to be sorted will be high. Although a smaller base implies a lower number of “containers” needed to be created in the counting sort iterations, the number of digits ‘m’ will be higher. This suggests that there are more counting sort iterations needed as the number of “vertical columns” increases. As the base increases, the number of “containers” needed to be created also increases, however

the number of digits ‘m’ needed to represent the value also decreases, resulting in fewer sortings (counting sorts) required and thus speeding up the processing time (between base 2 to base 16). This will eventually lead to an optimal point where processing time is the fastest as the number of "columns" and "containers" has reached a perfect balance (around base 2^16).

Moreover, as can be observed from the scatter plot, a large base will also result in a relatively longer processing time. However, as the base increases, the balance will be disrupted and thus the processing time will rise. As can be seen from the graph, there is an increasing trend of processing time used at around base 2^16 to 2^24 and so on . As the base rises, the number of "column" rises until it reaches a certain point where an increment of base will no longer result in any significant decrement of "column" to sort. Eventually, the number of digits used to represent the values become 1 or close to 1 as the base continue to grow, and any increment in base will only result in more processing time as the increment of the base will no longer result in a significant decrement in the amount of sorting required and will only increase the time as more containers in the counting_sort of radix sort now need to be created and looped through. The scatter plots are expected to grow continuously as the bases increase after base 2^24.