1. Divide and conquer
   1. Pseudo

Set min equal to large number

Set max equal to 0

Get array

Search function catches array

If array size is greater than 1

Find middle var

Make new array of left and right of original array

Recursively call search function passing left array

Recursively call search function passing right array

Else

Check if array[0] is less than minimum

Set new minimum if true

Check is array[0] is more than maximum

Set new maximum if true

Call search function, pass array

* 1. T(n) = 2T(n/2) + c
  2. Asymptotic Running Time = n
     1. Iterative is n^2 which would make it slower as it would need to read through it multiple times rather than linear time that runs through the array entirety once.

1. Merge sort 3
   1. Pseudo

Get array

Call sort3, pass array

Sort3 catches array

Check size of array is divisible by 3

Add 1 to % if odd number

Set m1 to division of 3

Set m1 to 1 if below 1

Set m2 to m1\*2

Make array1 equal to values of array below m1

Make array2 equal to values of array above m1 and below m2

Make array3 equal to values of array above m2

Call sort3 for array1

Call sort3 for array2

Call sort3 for array3

Make tmp array and indexes

Loop array1 and array2, while indexes are below array# length

Append tmp array with lower value between array1 and array2 at their indexes

Increment winners index

While loop array1 to add excess numbers to tmp

While loop array2 to add excess numbers to tmp

Loop tmp array and array3, while indexes are below array# length

Set array with lower value between tmp array and array3 at their indexes

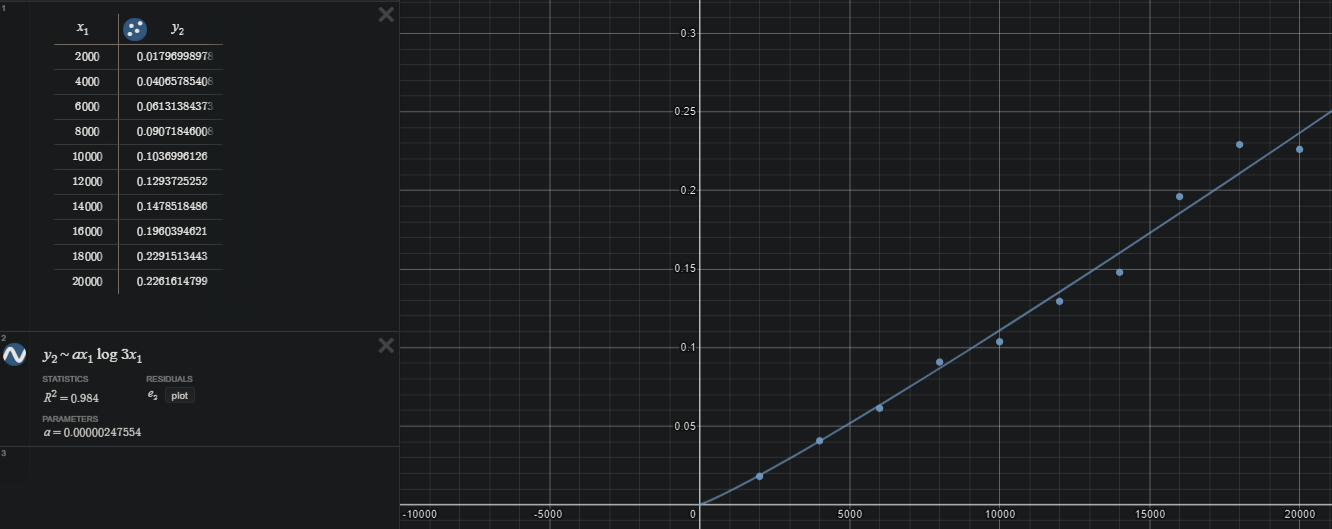
Increment winners index

While loop tmp array to add excess numbers to array

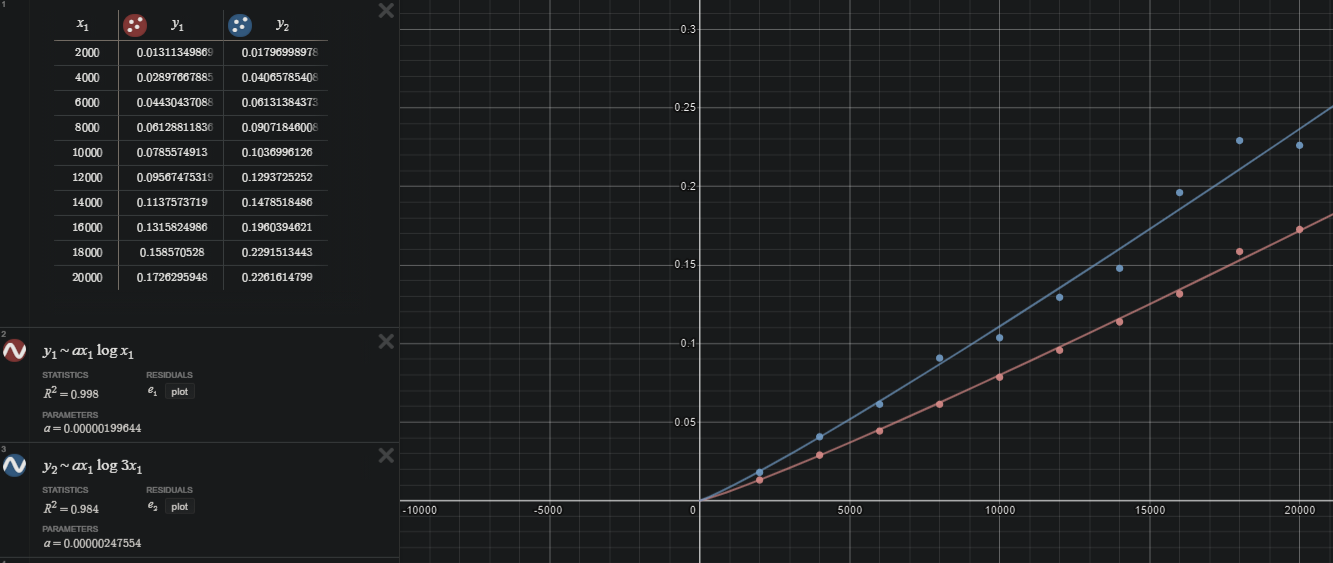
While loop array3 to add excess numbers to array

* 1. T(n) = 3T(n/3) + O(n)
  2. Asymptotic Running Time = O(n log 3n)

1. mergesort3.py
2. mergesort3Time.py
   1. Implement
   2. Collect data



* 1. Plot: red is merge, blue is merge3



* 1. Merge sort is faster and more consistent. Mergesort3 varies much more as the size of the array bounces between being divisible by 3 or not. I took 10 samples for merge3 and averaged them to make that graph, by itself, merge3 is much more scattered.