Lab 4

1) The order of functions by increasing asymtotic rate (from most to least efficient).

1) 2^{10}
2) $2^{\log_2(n)}$
3) $3n + 100\log_2(n)$
4) $4n$
5) $n\log_2(n)$
6) $4n\log_2(n)$
7) $n^2 + 10n$
8) n^3
9) 2^n

2)

n time

For i in range (len(S)):

$\in(S[i])$

$\downarrow$

$O(i)$

$i = 0 + 1 + 2 + \cdots + n$

$= \frac{n(n+1)}{2} \rightarrow \frac{n^2+n}{2}$

And, Since, in the binary search we take the highest power, therefore the above algorithm is running in $O(n^2)$ time.

3) Running time of each algorithm is as following;

example 1) $O(n)$ time

example 2) $O(\frac{n}{2})$ time

example 3) $O(n^2)$ time

example 4) $O(n)$ time

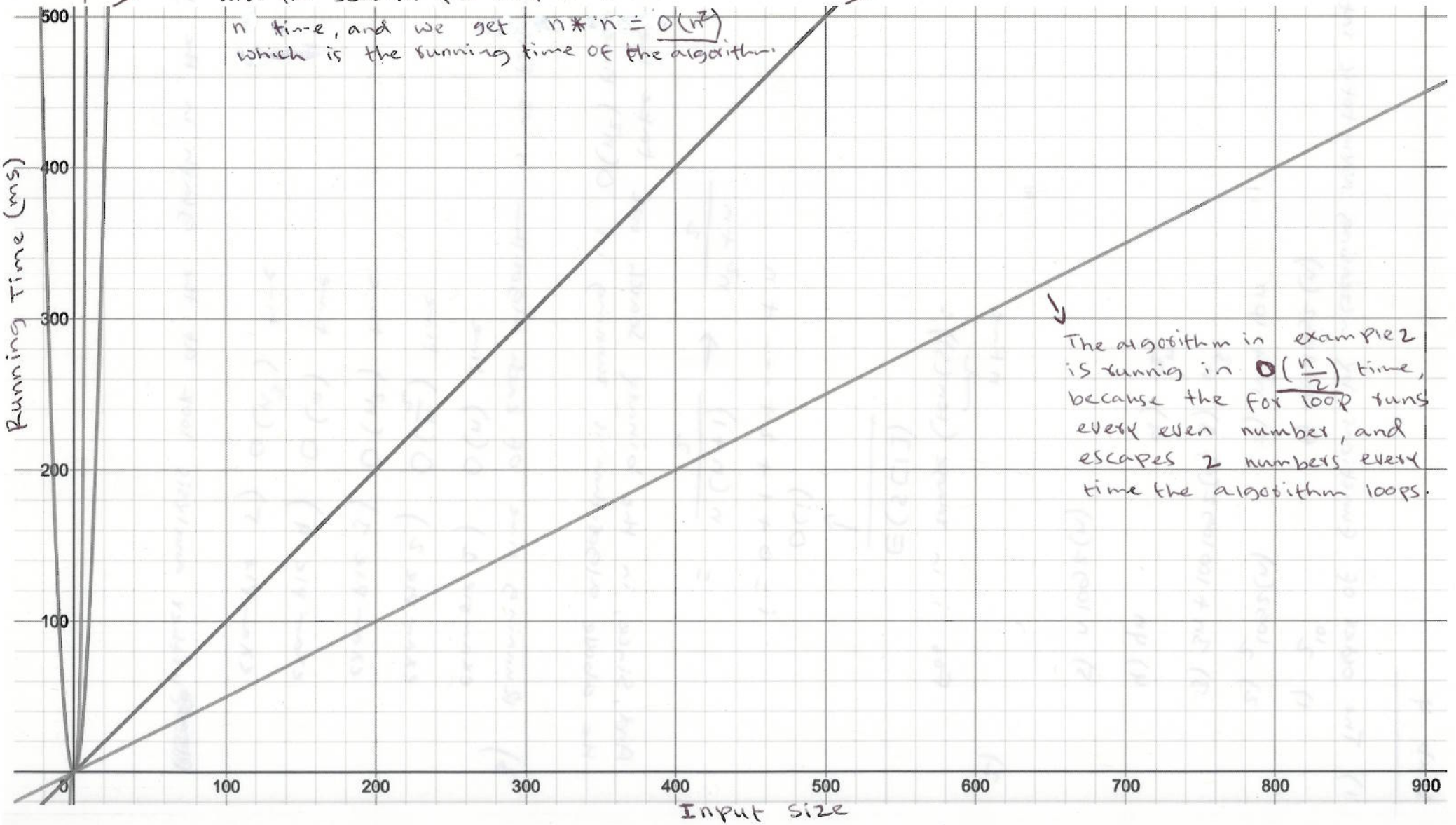
example 5) $O(n^3)$ time

For further analysis look at the graph in the next page.

In the example 5 the algorithm is running in $O(n^3)$ time, because the algorithm is running three for loops and each of the for loop is running in n time and the result of $n * n * n = O(n^3)$

The algorithms in example 1 and example example 4 are both running in $O(n)$ time, because each of the algorithms are running one for loop that is running in n time

The algorithm in example 3 is running in $O(n^2)$ time, because the first for loop runs in n time and the second for loop runs in n time, and we get $n * n = O(n^2)$ which is the running time of the algorithm



The algorithm in example 2 is running in $O(\frac{n}{2})$ time, because the for loop runs every even number, and escapes 2 numbers every time the algorithm loops.