**CSGY 6033 Algorithms Cheat Sheet**

Contents

[Asymptotic Function Ordering 2](#_Toc51098613)

[Equations 2](#_Toc51098614)

[Big O: 2](#_Toc51098615)

[Big Omega: 3](#_Toc51098616)

[Theta: 3](#_Toc51098617)

[Master Method: 3](#_Toc51098618)

[Induction: 3](#_Toc51098619)

[Linear Probing: 3](#_Toc51098620)

[Quadratic Probing: 3](#_Toc51098621)

[Summation: 3](#_Toc51098622)

[Log N Formula: 3](#_Toc51098623)

[Merge-Sort 4](#_Toc51098624)

[Appendix 4](#_Toc51098625)

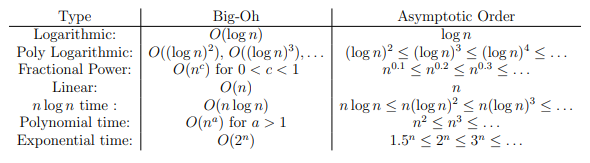
[Summation Tricks 4](#_Toc51098626)

[Logarithm Tricks 5](#_Toc51098627)

# Asymptotic Function Ordering

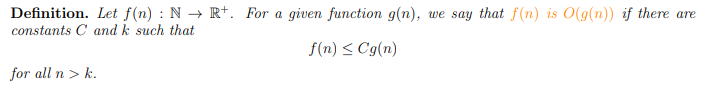
|  |  |  |  |
| --- | --- | --- | --- |
| # | Ex 1. | Ex 2. | Ex 3. |
| 1 | log(n) | log (n0.5) = (0.5) log(n) | log(n2) = 2\*log(n) |
| 2 | n0.3 \* log(n) |  |  |
| 3 | n \* log n | n log (n2) = 2\*n log n |  |
| 4 | n \* (log n) 2 |  |  |
| 5 | n1.3­ = n \* n3 |  |  |
| 6 | n2log n |  |  |
| 7 | 4n/3 = (41/3)n |  |  |
| 8 | 2n |  |  |
| 9 | 3n |  |  |
| 10 | n! |  |  |
| 11 | nn |  |  |

Asymptotically Function (increasing order)

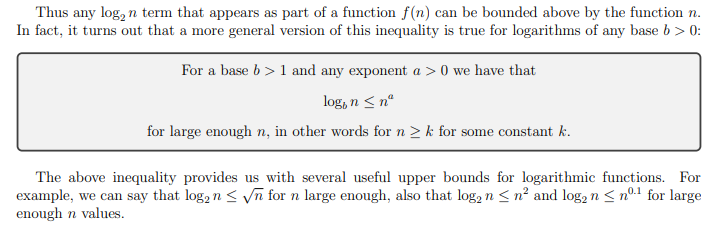


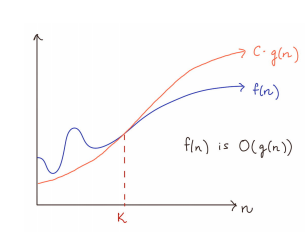
# Equations

Big O:Will give us an upper bound for function f(n) when n is very large/asymptotically. This is used extensively to describe the worst-case scenario for the number of operations used by an algorithm. The notation used in the definition is: O(g(n)) which is read “big-oh of g of n”

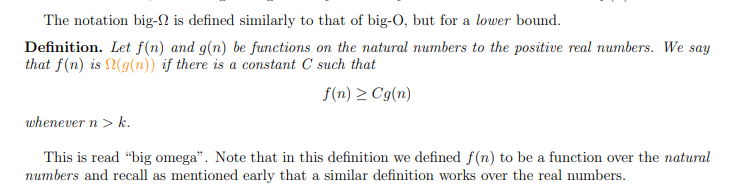








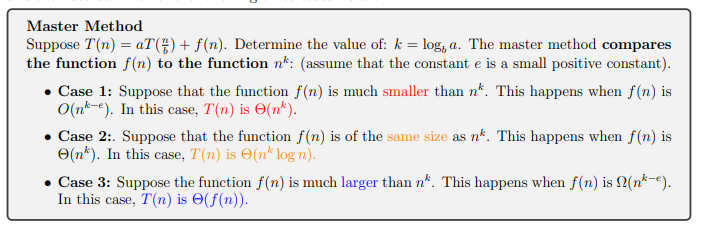
## Big Omega:



## Theta:



Master Method:the algorithm takes a problem of size n and calls itself on a problem of size . So as long as b > 1, it is working on a smaller problem size. These types of situations have the following equation for the run time:



## Induction:

when induction, only reason it works is working out exactly what induction hypothesis is.

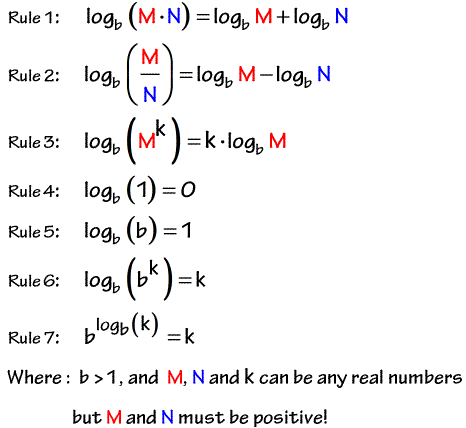
**Rules:** Must work out Big O exactly. If we do not get induction hypothesis exactly, doesn’t work out.

Linear Probing: h(k) + c1i

Quadratic Probing: h(k) + c1i + c2i^2

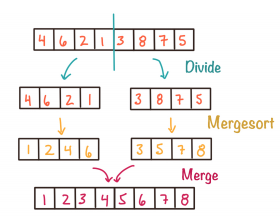
## Summation:

## Log N Formula:



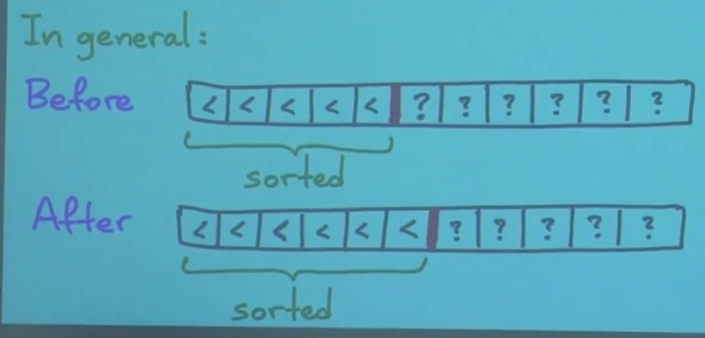
Merge-Sort sorts an array of n items using a divide-and-conquer approach. The three steps of merge-sort can be described as follows:

1. Divide the array of length n into two subsections of size n/2.
2. Conquer: Sort the two smaller sections using mergesort on each half.
3. Combine: Merge the two smaller sections together into one large sorted sequence



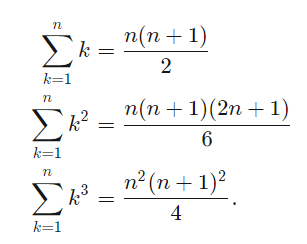
## Insertion-Sort

1. Start with sorted prefix of size 1 (sorted on left, unsorted on right)
2. Extend size of sorted prefix by 1
3. Repeat

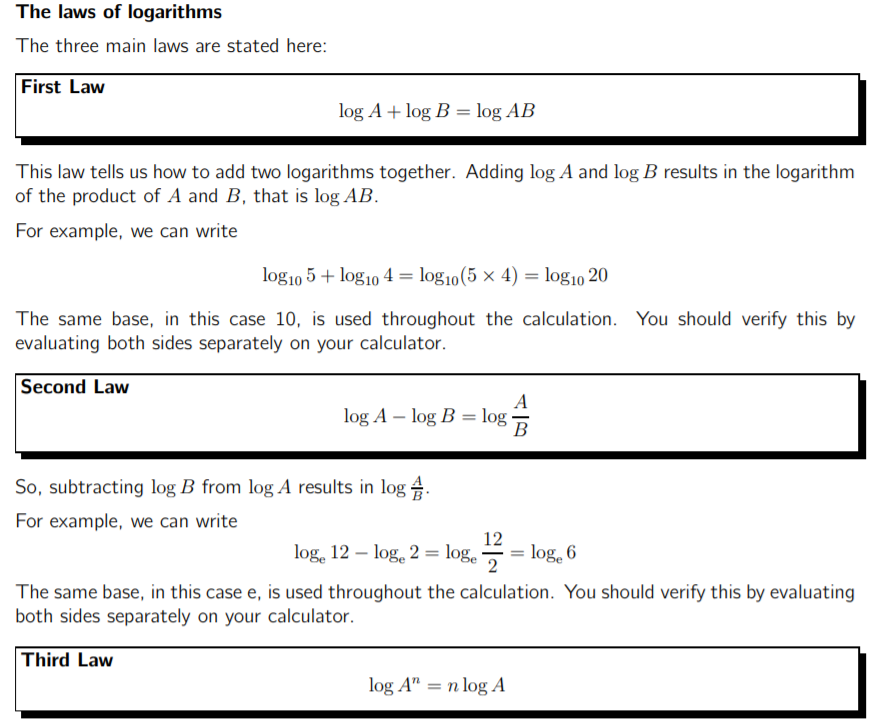


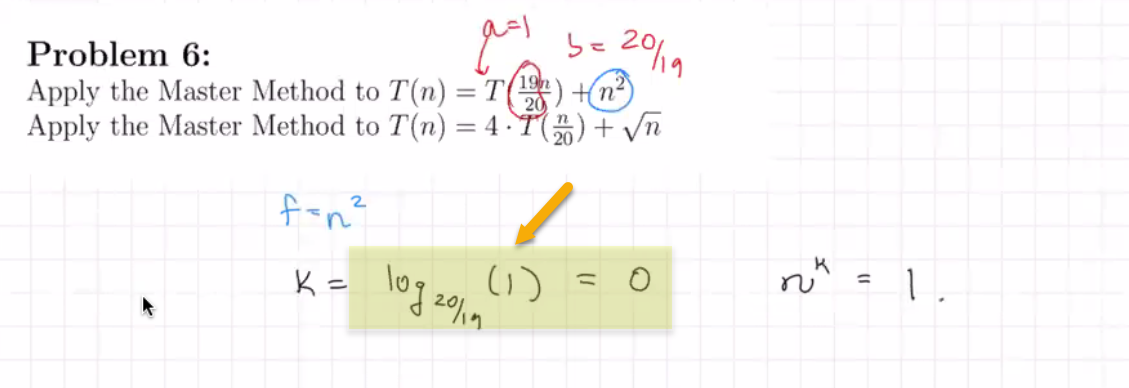
# Appendix

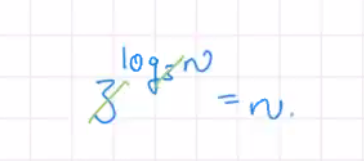
## Summation Tricks

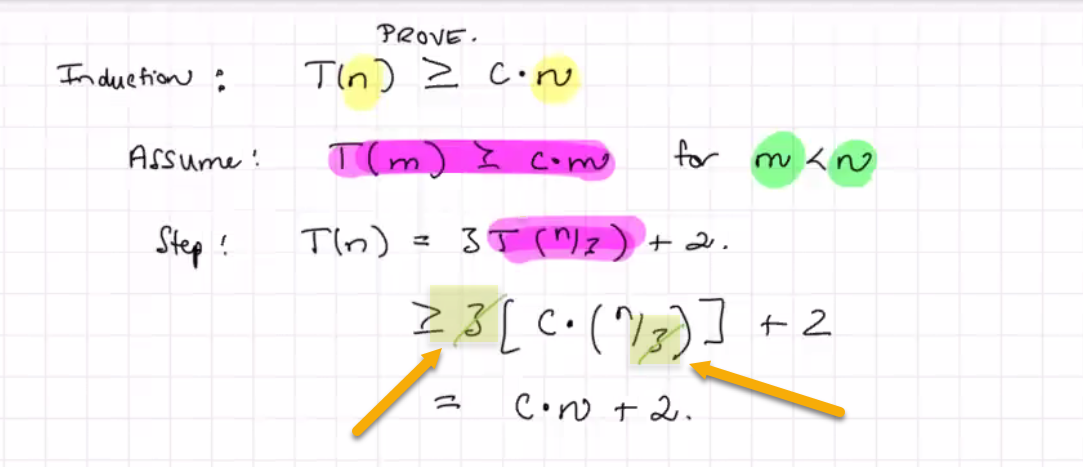


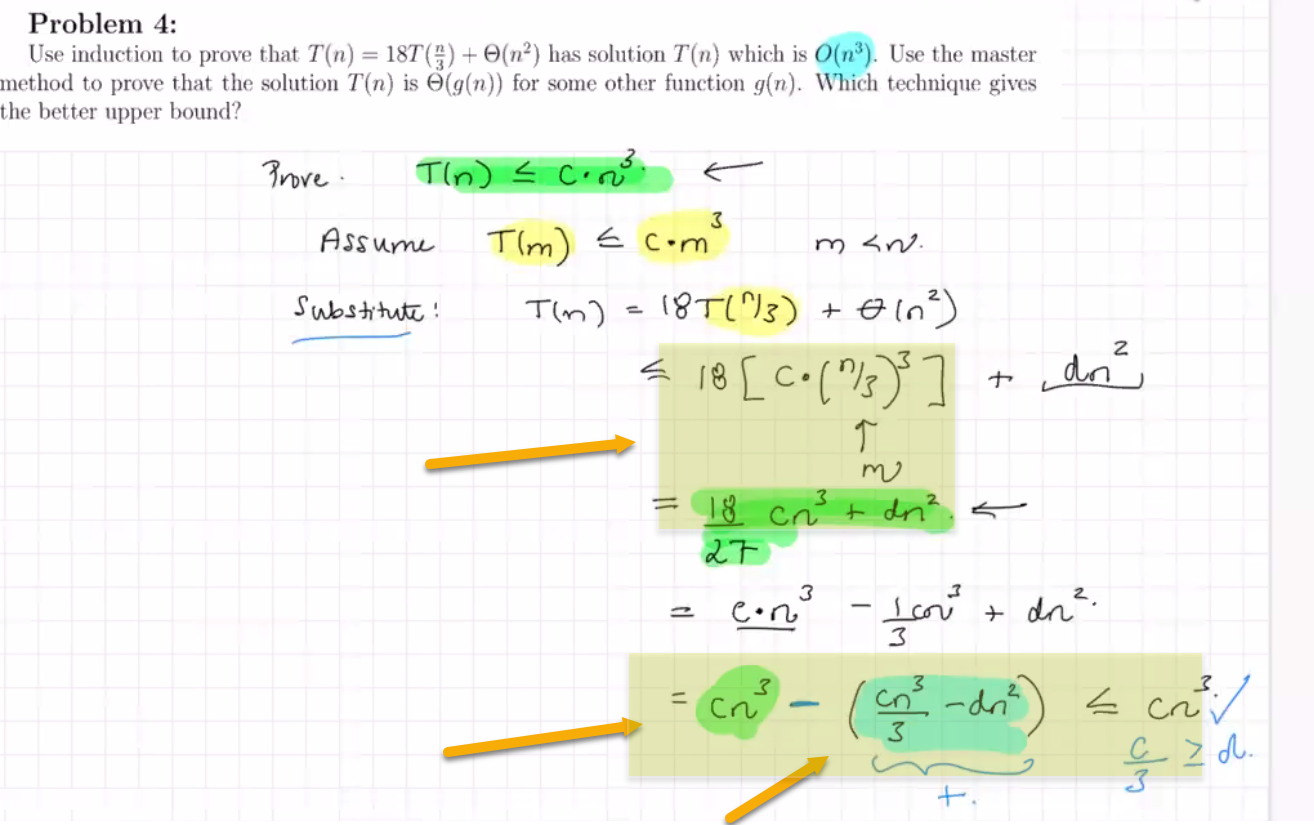
## Logarithm Tricks

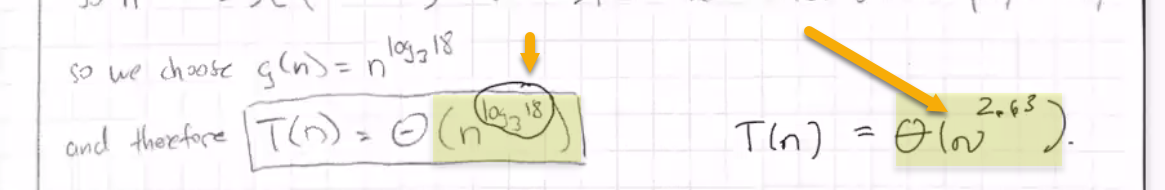


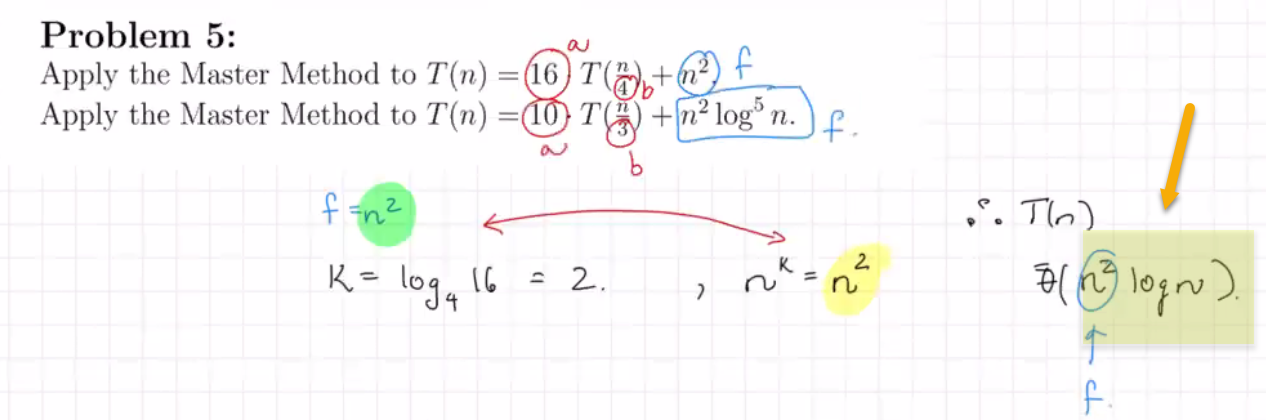


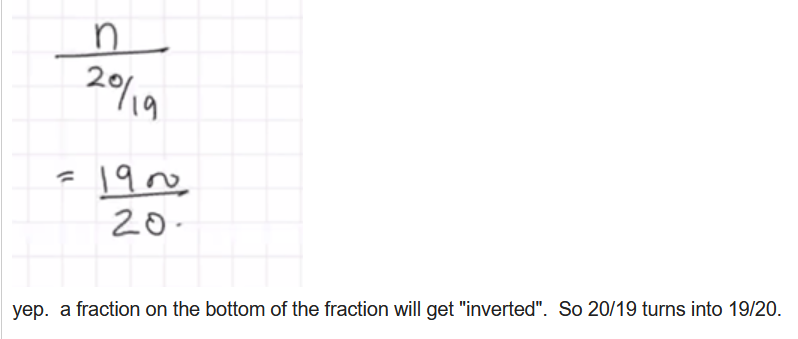












**.**