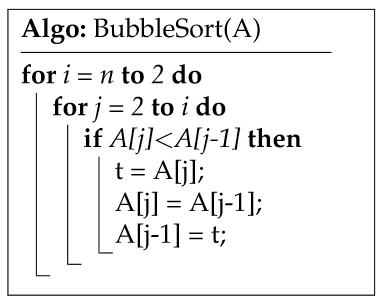
# Algorithms

Finite sequence of unambiguous instructions

Step-by-step outline

# Sorting

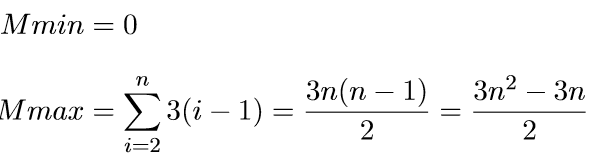
**Bubble Sort:** scan sequence and swap unsorted adjacent elements



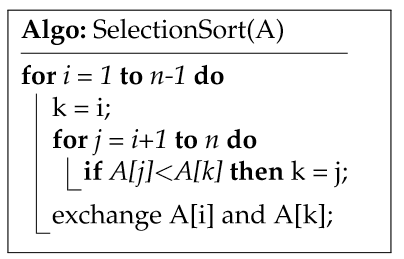
Comparisons:



Exchanges:



**Selection sort**: Select smallest element and swap with first element.



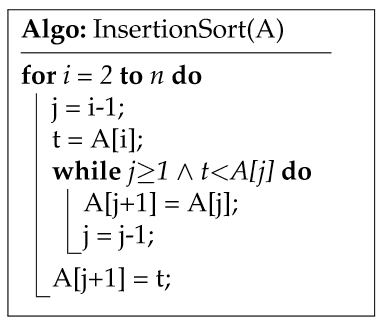
Comparisons:



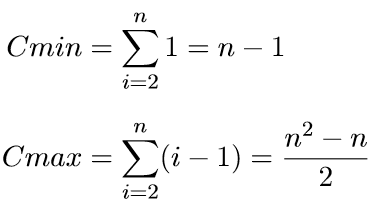
Exchanges:



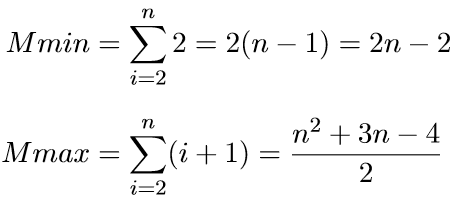
**Insertion sort**:Take first element and consider it as sorted. Put new elements in correct position.



Comparisons:



Exchanges:

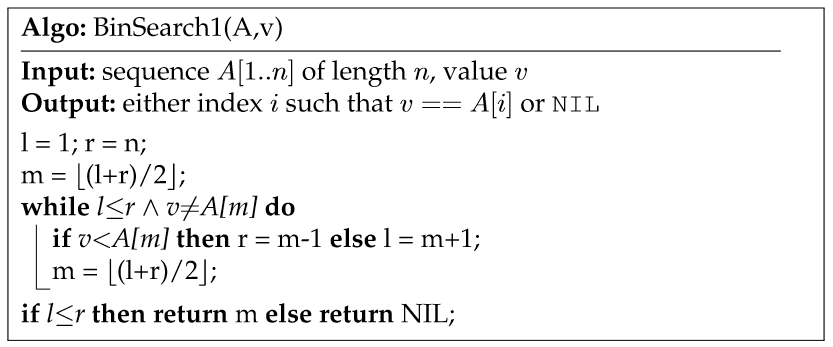


# Recursion

Contains itself as part of it

# Algorithmic complexity

Binary search: sorted array, look at middle element and continue searching like this in halfes.

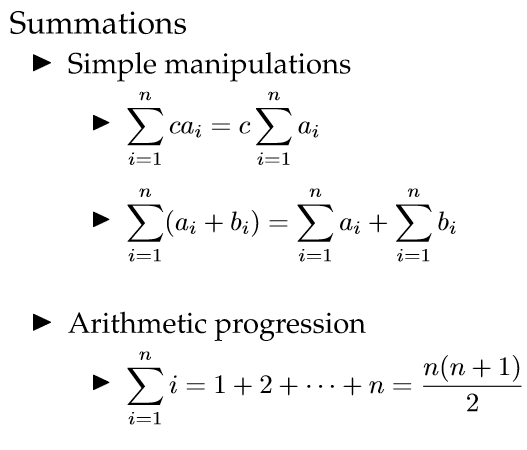


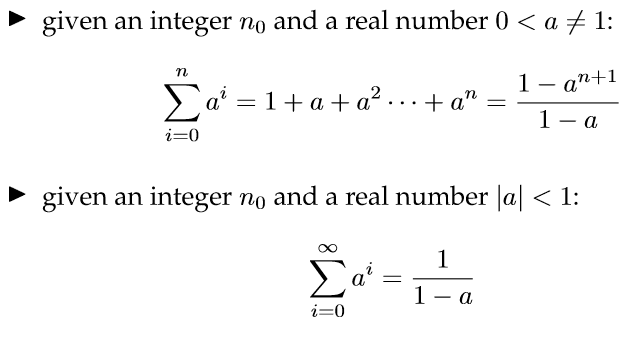
# Correctness

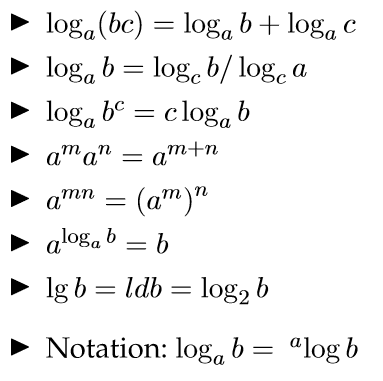
Partial correctness: desired output is delivered

Total correctness: desired output is delivered and the algorithm terminates.

# Asymptotic complexity







**BIG-O notation**: worst case

**Big-Ω notation**: best case

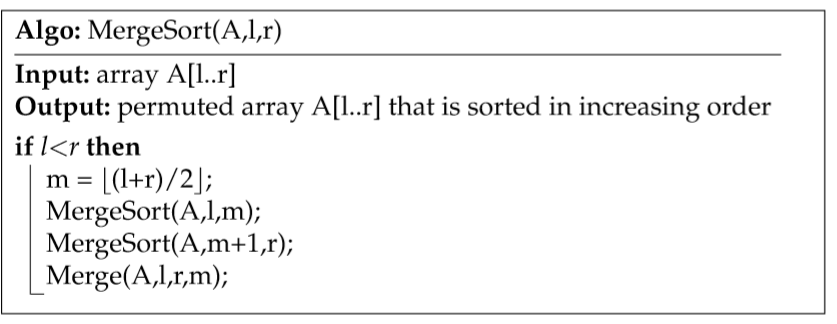
**Big Θ notation**: asymptotic complexity

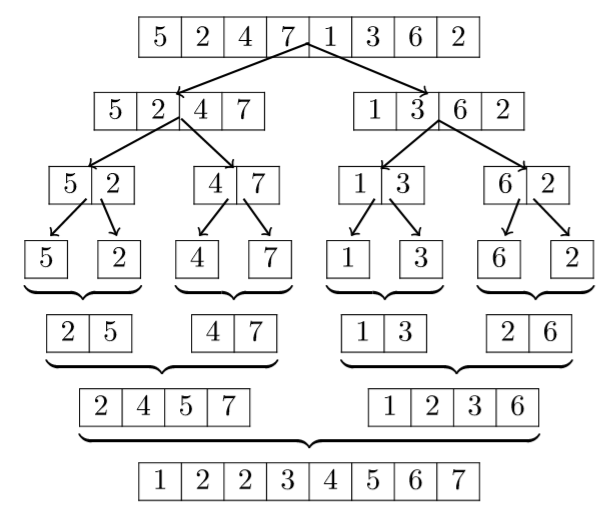
How to determine asymptotic complexity:

1. Determine fastest growing part with n🡪infinity
2. Remove constants out of that part

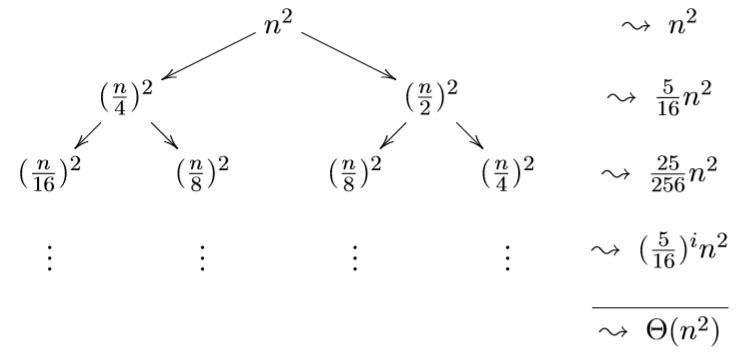
# Merge sort

Divide the array into two and sort within these smaller arrays





# Recurrences

**Recursion tree**:

# Master method

