DAA -01 (20/01/25)

Algorithm:

→ Seq of steps to solve a prob

→ Sorting:

(inc, dec, lexicographic)

1) ⇒ Search

↳ success $\begin{cases} \text{best}(1) \\ \text{worst}(n) \end{cases}$ Avg ≈ n-1 ≈ n

↳ unsuccess (n)

→ Refine linear search

i) sort.

↳ If greater val found while linear search ⇒ search unsuccess.

ii) sort + binary search

log(n)

↳ sorting ($N, N^2, n\log n$).

Insertion sort (iterative)

WC → n^2 , BC → n

Merge sort (recursive)

WC, BC → $n\log n$

→ for memory constraint ⇒ insertion sort

DATE: DATE:

PC1: 10s PC2: 100s

↳ tell worst case:

Tester → PC3 : 10000s

⇒ Algorithm Analysis:

↳ machine independent.

↳ language independent.

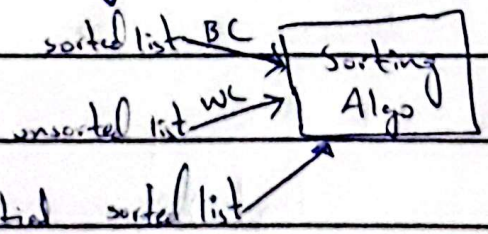
↳ based on algo/pseudo code.

↳ based on time & assymptotic behaviour.

↳ estimation.

(O, π, Θ)

2) Sorting:



Insertion Sort	Merge Sort
BC → n	BC → nlogn
WC → n^2	WC → nlogn
Avg → n^2	Avg → nlogn

Algo:

1) Seq of steps to solve a prob

2) Algo must halt (with correct O/P)

DAY: _____ DATE: _____

3) Unambiguous steps

4) Handle edge cases or clearly state unhandled cases.

n^2+n-1 (run-time)

$n^2 \rightarrow$ Big O

$\rightarrow$ Omega Ω

$\rightarrow \Theta$

DAA-02 (24/01/25)

Analysis of Algo:

Algorithm (Program independent)

↳ ~~Stg~~ Seq of steps

↳ unambiguous steps/statements

↳ Must HALT (with correct O/P)

↳ Edge case (handled)

↳ Mathematical tool → must be correct.

↳ used for analysis	↳ mathematical induction.
• space	↳ proofs
• time	

→ Heap (Priority Queue) (Heap sort Algo) $n\log n$.

Limitation:

→ can't go beyond specified.

DATE: _____ DATE: _____

↳ space complexity ⇒ extra memory used

↳ time complexity ⇒ runtime

Δt → change in time with respect to increase in size

Problem size ∝ Time

Design:

↳ Divide & Conquer

↳ Dynamic ↳ Greedy.

⇒ Greatest Common Divisor Problem:

1) Euclidean:

GCD(m,n)

if (m%n) == 0 return n

else return GCD(n, mmodn)

2) 60, 12

↳ if small value divides both ⇒ GCD

else decrement small value & check again

60%12 == 0 && 12%12 == 0

↳ GCD(60,12) ⇒ 12.

prob: If m & n are relatively prime & extremely large

⇒ if relatively prime ⇒ return 1

↳ Euclidean is better.

DATE: _____ DATE: _____

3) ↳ common prime factors.

↳ requires list of prime factors.

↳ seive of Eratosthesis (Algo for P.M b/w 2 nums)

24 → 2.2.2.3
18 → 2×3×3

n = 25

(Skip 1) ~~1~~ 2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~

~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~

17 ~~18~~ ~~19~~ ~~20~~ ~~21~~ ~~22~~ 23

~~24~~ ~~25~~

1) Eliminate factors

↳ p ⇒ first eliminate p·p

2 → 4 | 3 → 9

$\sqrt{n}$ iterations req.

$\sqrt{25} = 5$.

⇒ If not perfect square ⇒ $\lfloor \sqrt{n} \rfloor$

N/2, N/3, N/5

$\left(\frac{n}{2}, \frac{n}{3}, \frac{n}{5}, \dots\right)$

$n\left(\frac{1}{2}, \frac{1}{3}, \frac{1}{5}, \dots\right)$

Harmonic Progression.

$\sqrt{n}(\lg\lg n)$

nxn (initialized with 1)

DATE: _____ DATE: _____

↳ Euclidean is better in a sense it does handle relatively prime (1 iteration)

I/P → [Sorting — insertion sort] → O/P sorted.

Input Types:
- Best Case — sorted
- Worst Case — Rev sorted
- Avg Case — Semi sorted.

```
int i = 0        → unit time
a = b + c.       → unit time
for i = 0 - m    → n-unit time
    c = 5
```

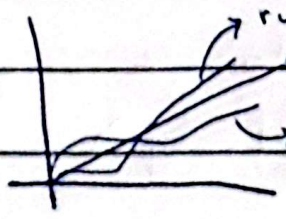
ALU → add, sub only (arithmetic)
comparison (logical unit)

int → 4 byte → 4 machine cycle.
or 2 machine cycle

delay → distance & cycles.

i) Registers → fastest.

ii) Cache



DAY: _____ DATE: _____

⇒ We cannot evaluate system qualitative time.

∵ n^2+n+1 ⟶ machine dependent.

↓ Assymptotic behaviour

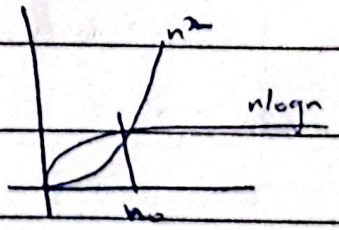
→ Order of Growth.

→ Efficiency class.

- linear → quadratic
- cubic → polynomial
- exponential → log.

n	n^2	kn^2
10^{10}	10^{20}	$10^{200} = 10^{180+20}$
10^{20}	10^{40}	$10^{200} = 10^{180+20}$
10^{40}	10^{80}	$10^{100} = 10^{20+80}$

Merge sort ⇒ $an\log n + bn + c$



⇒ merge sort beats insertion sort after n_0.

DATE _____ DATE _____

DAA-03 (27/01/25)

Basic Operation

sometime magnitude of value matters

10 101024 (even or odd?)

↳ takes more time.

In sorting ⇒ comparison operation

⇒ $T(n) = COP\ C(n)$

Total time — Cost of Basic Operation — (No of times basic operation being executed)

⇒ $T(n) = C(n)$ (If $COP = 1$)

Machine 'A' Algo 'X'

1) A = 10 ins/s.

size-n → 1s

B = 100 ins/s.

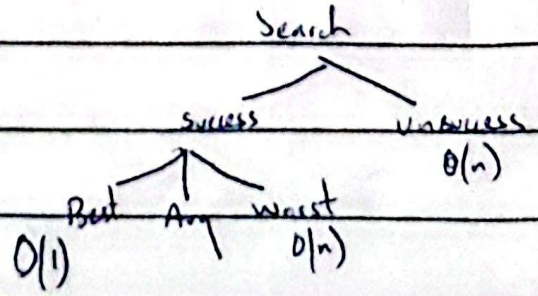
size-n → ? (0.1s)

2) A = 100 ins/s.

size-n → 1s

B = 100,000 ins/s.

size-n ⇒ ? (0.001 sec)



DATE: DATE:

Sorting:

Insertion Sort
- sorted → n
- unsorted → n^2
- semi-sorted → n^2

Merge sort → $O(n \log n)$

single element array (supposed BC)

Avg case has multiple possibilities.

⇒ Avg Analysis ✗

⇒ Probability that element exists is p

~~~~ does not exists $1-p$

(Max no of comparisons) successful + unsuccessful

$$1p/n + 2p/n \ldots + ip/n + \ldots + n(p/n) + (1-p)\,n$$

$$\frac{p}{n}(1+2+3 \ldots + i \ldots + n) + (1-p)\,n.$$

$$\frac{p}{\cancel{n}}\left(\frac{\cancel{n}(n+1)}{2}\right) + (1-p)\,n.$$

$$\frac{p(n+1)}{2} + (1-p)\,n.$$

⇒ $p = 0$ → $0 + n$

⇒ $p = 1$ → $\frac{n+1}{2} + 0$.
~~~~

DATE _____ DATE: _____

DAA-04 (31/01/25)

1.2-2

$8n^2 \leq 64 n \lg_2 n$.

$n \leq 8 \lg_2 n$.

$n=1$ $\quad 1 \leq 8 \lg_2 1 \rightarrow \leq 8(0)$ ✗

$n=2$ $\quad 2 \leq 8 \lg 2 \rightarrow \leq 8$ ✓

powers of 2 $\quad 4 \leq 8 \lg 4 \rightarrow \leq 16$ ✓

$\lg_2 46 = \frac{\log 46}{\log 2}$ $\quad 8 \leq 8 \lg 8 \rightarrow \leq 24$. ✓

$16 \leq 8 \lg 16 \rightarrow \leq 32$. ✓

$32 \leq 8 \lg 32 \rightarrow \leq 40$. ✓

$64 \leq 8 \lg 64 \rightarrow \leq 48$ ✗

$32-64 \rightarrow$ mid = 46 (random try).

$46 \leq 8 \lg 46$.

$46 \leq 44$ ✗

$42 \leq 8 \lg 42 \rightarrow \leq 43$ ✓

$43 \leq 8 \lg 43 \rightarrow \leq 43.41$ ✓

1-1 Comparison of running times:

$f(n)$ ms $\rightarrow \lg n$ ms = 1s.

$\lg n \; 10^{-6} s = 1 s$

$\lg_2 n = 10^6$

$n = 2^{10^6}$

DATE: _____ DATE: _____

Insertion Sort:

- Decrease & conquer
- Space complexity [ ] O(n)
- Time Complexity → $O(n^2)$ worst case
  → $O(n)$ best case.

Avg case is taken as close to worst case.

→ Place an element from unsorted part to sorted part of the array.

~~Bubble Sort~~: for i = n: → n+1 iterations (1 terminating check)
→ loop body → n times.

⇒ Cost is cause of the machine.

⇒ $c_3 = 0$ → # comment will be read but not passed as machine ins.

(Best case: Time)

$T(n) = c_1 n + c_2(n-1) + 0 + c_4(n-1) + c_5(n-1) + c_8(n-1)$

// c_6 & c_7 not executed.

$= c_1 n + c_2 n - c_2 + c_4 n - c_4 + c_5 n - c_5 + c_8 n - c_8$

$= (c_1 + c_2 + c_4 + c_5 + c_8)n - (c_2 + c_4 + c_5 + c_8)$

$= a'n - b \Rightarrow \cong O(n)$

DATE: _____ DATE: _____

Worst Case:

loop invariant

Mathematical Induction.

Base step – True

Inductive step.

IH – Assume True (k)

prove k+1

⇒ one counter example ⇒ denied whole.

Algo Correctness:

→ Initialization.

→ Maintenance.

→ Termination.

→ loop invariant (finite)

Mathematical Induction (infinite).

DATE: _____ DATE: _____

Avg Case:

10	30	20	70

t_i ⇒ avg time of while loop till i

$$\sum_{i=2}^{n} t_i \leq i$$

$$\sum_{i=2}^{n} t_i = 2 + 3 + \cdots + n.$$

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

$$\sum_{i=2}^{n} = (t_i - 1)$$

$$\Rightarrow \sum_{i=1}^{n-1} i = \frac{n(n-1+1)}{2}$$