Design and Analysis of Algorithms (DAA)

Design and Analysis of Algorithms (DAA) ①-②

Algorithm:- A sequence of logically related instructions to solve a Computational problem

Computational problems
↳ I/P & o/P

I/P: Array A, x	I/P: Array A
? Does $x \in A$	? Sort A
o/P: Yes / No	

Design and Analysis of Algorithms (DAA) ①-② -③

Algorithm:- A sequence of logically related instructions to solve a Computational problem

Computational problems
↳ I/P & o/P

I/P: Array A, x

? Does $x \in A$

o/P: Yes / No

I/P: Array A

? Sort A

Any Algorithm must satisfy (characteristics)

↳ I/P, O/P

↳ Finiteness:- It must terminate after a finite amount of time (1s, 1hr, 1day, 1month)

↳ Definiteness:- Each instruction must be unambiguous → $\left[\begin{array}{ll} x=5 & ✓ \\ y=5 \text{ or } -4 & ✗ \end{array}\right]$

↳ Must involve 1 basic arithmetic instructions!

Ⓟ

Design and Analysis of Algorithms (DAA)

→ MAX in an integer array

```
i=0
Max= A[i]
For i=1 to n
    if(A[i]> Max)
        max=A[i]
Return Max
```

Design and Analysis of Algorithms (DAA) ④-⑤

→ MAX in an integer array

Ⅰ

```
i = 0
Max = A[i]
For i=1 to n
    if(A[i] > Max)
        max = A[i]
return Max
```

Ⅱ

```
Sort A
return A[n]
```

Ⅲ

$$\underbrace{a_1 \text{ ---- } a_{n/2}}_{Max_1} \qquad \underbrace{a_{\frac{n}{2}+1} \text{ --- } a_n}_{Max_2}$$

Max

↳ Many approaches (Algorithms)

If the problem is solvable (There exists an Algorithm)

↳ How many different algorithms are possible.

P

Design and Analysis of Algorithms (DAA)

CS

↓

A set of Computational problems

Solvable (There exists an Algo)

unsolvable (No Algo exists)

Design and Analysis of Algorithms (DAA)

⑥-⑦

CS

↓

A set of Computational problems

Solvable (There exists an Algo)

unsolvable (No Algo exists)

↳ How many different algos are possible

↳ How do we Compare Algo

↳ How to pick the best one (Efficiency)

Design and Analysis of Algorithms (DAA) ⑥-⑦-⑧

CS

↓

A set of Computational problems

Solvable — unsolvable

(There exists an Algo) — (No Algo exists)

↳ How many different algos are possible

↳ How do we Compare Algo

↳ How to pick the best one (Efficiency)

① Correctness [Is the Strategy/Approach/Algorithm designed performing the intended task]

② How to measure the time spent / Space occupied (time Complexity) / (Space Complexity)

③ Optimality — A_1 is least

$$\begin{cases} A_1 - 10\,s \\ A_2 - 20\,s \\ A_3 - 30\,s \end{cases}$$

Can we better A_1

Is there an algorithm whose time is $< 10\,s$

Suppose A_4 (2 s) ; Is A_4 best ever ? Ⓟ