# Algoritme:

En veldefineret computer procedure der omdanner input til output.

Der kan være flere rigtige algoritmer til hver opgave

Kriterier: for hvert input giver den det korrekte output

## Kompleksitet:

Space: hvor meget hukommelse bruger den

Tid: hvor lang tid tager algoritmen

God eller dårlig algoritme

Vil måske afhænge af input: best, worst og average case.

Incorrect algoritme: giver måske ikke output på input, eller giver forkert output

Brugbar i tilfælde hvor der ikke er nogen bedre korrekt algoritme, men man kender fejlprocenten og margin.

## Lineær søgning:

Defineret af størrelse på input

Kompleksitet som funktion af input størrelse n

Space used: array størrelse

Best case: A[1] = q, running time =1, konstant tid

Worst case: A[n] = q, running time = n, lineær tid

Avarage case: q er forventet I midten af sekvensen, running time = n/2, stadig lineær tid

Ikke godt hvis der er rigtig mange data. Big Data æra

## Binær søgning

Virker hvis data er sorterede.

Ideer bag: del og hersk (divide and conqure), en af nøgle design teknikkerne.

Rækkefølgen af data sørger for at elementer man leder efter ikke er ignoreret.

**Sorterings problem:**

Input sekvens er kaldt en instans af sorterings problem.

Der kan være uendeligt mange input

Nogle algoritmer er hurtigere end andre.

**Fremgangsmåde**

1. Analyser specifikationer på forholdet mellem input og output
2. Design algoritmer og datastrukturer
3. Implementer algoritmer i rigtigt programmeringssprog

**Algoritme mål**

* Korrekthed
* Effektivitet

Datastruktur:

Gemmer og ograniserer date for at få tilgang til og

Datastruktur: Hvordan organiserer vi data der skal håndteres

Algoritme: Beskriv processen hvor data er håndteret