

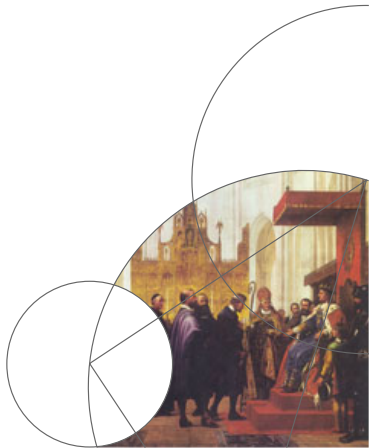


Det Naturvidenskabelige Fakultet

# Gymnasiebesøg

## Algoritmer og problemløsning

Gymnasietjenesten DIKU  
Department of Computer Science



- ① Dagens program
- ② Introduktion til algoritmer
- ③ Algoritme vs Algoritme
- ④ Algoritme design
- ⑤ Øvelser
- ⑥ Opsamling og Spørgsmål



# Program for idag

## Indtil frokost

- Algoritme design og metoder.



# Program for idag

## Indtil frokost

- Algoritme design og metoder.
- Hvordan man kan sammenligne forskellige løsninger.



# Program for idag

## Indtil frokost

- Algoritme design og metoder.
- Hvordan man kan sammenligne forskellige løsninger.
- Hvad er grænserne for algoritmer?



# Program for idag

## Indtil frokost

- Algoritme design og metoder.
- Hvordan man kan sammenligne forskellige løsninger.
- Hvad er grænserne for algoritmer? **Historie tid!**



# Program for idag

## Indtil frokost

- Algoritme design og metoder.
- Hvordan man kan sammenligne forskellige løsninger.
- Hvad er grænserne for algoritmer? **Historie tid!**

## Fra 13 til 14.30

- Øvelser i algoritmer



# Program for idag

## Indtil frokost

- Algoritme design og metoder.
- Hvordan man kan sammenligne forskellige løsninger.
- Hvad er grænserne for algoritmer? **Historie tid!**

## Fra 13 til 14.30

- Øvelser i algoritmer
- Sjove gåder





# Program for idag

## Indtil frokost

- Algoritme design og metoder.
- Hvordan man kan sammenligne forskellige løsninger.
- Hvad er grænserne for algoritmer? **Historie tid!**

## Fra 13 til 14.30

- Øvelser i algoritmer
- Sjove gåder
- Opsamling og spørgsmål



# Hvad er en Algoritme?

*An algorithm is a self-contained step-by-step set of operations to be performed that can be expressed within a finite amount of space and time and in a well-defined formal language.*



# Hvad er en Algoritme?

*An algorithm is a self-contained step-by-step set of operations to be performed that can be expressed within a finite amount of space and time and in a well-defined formal language.*

## På dansk

En algoritme er en **opskrift** på hvordan et bestemt problem kan løses.



# Havregryns algoritme

## Eksempel

### Algorithm 1

Indgangsbetingelser: En skål, mælk, havregryn

Udgangsbetingelser: Morgenmad

**while** Skålen ikke er fyldt med gryn **do**

    Hæld Gryn i Skålen

**end while**

**if** Jeg er tyk **then**

    mælk = Minimælk

**else**

    mælk = Letmælk

**end if**

**while** Skålen ikke er fyldt med mælk **do**

    Hæld mælk i Skålen

**end while**

# Krav til en algoritme?

## Krav til en algoritme

**Veldefineret** Ingen tvetydigheder og vendinger som: “så tager du det bedste resultat ...”



# Krav til en algoritme?

## Krav til en algoritme

**Veldefineret** Ingen tvetydigheder og vendinger som: “så tager du det bedste resultat ...”

**Terminerer** Den må ikke køre for evigt, du skal garantere at den rent faktisk finder sit svar.



# Krav til en algoritme?

## Krav til en algoritme

**Veldefineret** Ingen tvetydigheder og vendinger som: “så tager du det bedste resultat ...”

**Terminerer** Den må ikke køre for evigt, du skal garantere at den rent faktisk finder sit svar.

**Input og output** Jeg skal vide at hvis jeg giver den  $A$  så returnerer den  $B$ .



# Krav til en algoritme?

## Krav til en algoritme

**Veldefineret** Ingen tvetydigheder og vendinger som: “så tager du det bedste resultat ...”

**Terminerer** Den må ikke køre for evigt, du skal garantere at den rent faktisk finder sit svar.

**Input og output** Jeg skal vide at hvis jeg giver den  $A$  så returnerer den  $B$ .

**Kan bevises** Det er muligt både at bevise korrekthed og køretid for algoritmen.





# Eksempler på algoritmer

Algoritmer bruges inden for alle former for problemløsning.



# Eksempler på algoritmer

Algoritmer bruges inden for alle former for problemløsning.

## Bioinformatik

*Longest common subsequence:* Sammenligning af DNA strenge for at se hvor beslægtede to strenge er.



# Eksempler på algoritmer

Algoritmer bruges inden for alle former for problemløsning.

## Bioinformatik

*Longest common subsequence:* Sammenligning af DNA strenge for at se hvor beslægtede to strenge er.

## Primtals faktorisering

Bruges i *kryptering*: Basis for at vi kan have sikker kommunikation.



# Eksempler på algoritmer

Algoritmer bruges inden for alle former for problemløsning.

## Bioinformatik

*Longest common subsequence:* Sammenligning af DNA strenge for at se hvor beslægtede to strenge er.

## Primtals faktorisering

Bruges i *kryptering*: Basis for at vi kan have sikker kommunikation.

## Machine Learning

En samling af algoritmer der selv kan lære og finde egenskaber i store data sæt. Gør det muligt at løse problemer der før var uden for menneskers kunnen.



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer

### Hold B



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer
- Bruger algoritmen  
*Merge Sort*

### Hold B



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer
- Bruger algoritmen *Merge Sort*
- De kan sortere 10 millioner tal på under 20 minutter

### Hold B





# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer
- Bruger algoritmen *Merge Sort*
- De kan sortere 10 millioner tal på under 20 minutter
- De kan sortere 100 millioner tal på 4 timer.

### Hold B



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer
- Bruger algoritmen *Merge Sort*
- De kan sortere 10 millioner tal på under 20 minutter
- De kan sortere 100 millioner tal på 4 timer.

### Hold B

- Har en computer der er 1000 gange hurtigere end hold A



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer
- Bruger algoritmen *Merge Sort*
- De kan sortere 10 millioner tal på under 20 minutter
- De kan sortere 100 millioner tal på 4 timer.

### Hold B

- Har en computer der er 1000 gange hurtigere end hold A
- Bruger algoritmen *Insertion Sort*



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer
- Bruger algoritmen *Merge Sort*
- De kan sortere 10 millioner tal på under 20 minutter
- De kan sortere 100 millioner tal på 4 timer.

### Hold B

- Har en computer der er 1000 gange hurtigere end hold A
- Bruger algoritmen *Insertion Sort*
- De kan sortere 10 millioner tal på 5 timer.



# Valg af algoritme

## Sorterings algoritmer

Givet en liste af  $n$  tal ønsker vi at returnere en sorteret liste af længde  $n$ .

### Hold A

- Har en computer
- Bruger algoritmen *Merge Sort*
- De kan sortere 10 millioner tal på under 20 minutter
- De kan sortere 100 millioner tal på 4 timer.

### Hold B

- Har en computer der er 1000 gange hurtigere end hold A
- Bruger algoritmen *Insertion Sort*
- De kan sortere 10 millioner tal på 5 timer.
- De kan sortere 100 millioner tal på **23 dage!**



# Sammenligning af algoritmer

*Vi bruger begrebet Køretid for at beskrive  
hvordan tiden en algoritme bruger stiger med  
størrelsen af input.*



# Sammenligning af algoritmer

*Vi bruger begrebet Køretid for at beskrive hvordan tiden en algoritme bruger stiger med størrelsen af input.*

## Definition på køretid

En øvregrænse for den tid der bliver brugt på at løse et problem af størrelse  $n$ . Skrives som

$$O(n), O(n^2), O(n \lg n), O(n!), O\left(\frac{a}{b}\right)$$



# Sammenligning af algoritmer

*Vi bruger begrebet Køretid for at beskrive hvordan tiden en algoritme bruger stiger med størrelsen af input.*

## Definition på køretid

En øvregrænse for den tid der bliver brugt på at løse et problem af størrelse  $n$ . Skrives som

$$O(n), O(n^2), O(n \lg n), O(n!), O\left(\frac{a}{b}\right)$$

## Algoritme for minimums funktionen

Givet en liste  $X = [x_1, x_2, \dots, x_n]$  ønsker vi at returnere det mindste tal i listen. Hvad er algoritmen og hvad er køretiden?





# Minimums algoritme

## Eksempel

### Algorithm 2

Input: En liste  $X = [x_1, x_2, \dots, x_n]$

Ouput: Det mindste tal i listen.

---

$\text{min} = x_1$

**for**  $x_i$  in  $X$  **do**

**if**  $x_i < \text{min}$  **then**

$\text{min} = x_i$

**end if**

**end for**

---



# Minimums algoritme

## Eksempel

### Algorithm 3

Input: En liste  $X = [x_1, x_2, \dots, x_n]$

Output: Det mindste tal i listen.

---

$\text{min} = x_1$

**for**  $x_i$  in  $X$  **do**

**if**  $x_i < \text{min}$  **then**

$\text{min} = x_i$

**end if**

**end for**

---

## Analyse af algoritmen

Køretid?



# Minimums algoritme

## Eksempel

### Algorithm 4

Input: En liste  $X = [x_1, x_2, \dots, x_n]$

Output: Det mindste tal i listen.

---

$\text{min} = x_1$

**for**  $x_i$  in  $X$  **do**

**if**  $x_i < \text{min}$  **then**

$\text{min} = x_i$

**end if**

**end for**

---

## Analyse af algoritmen

Køretid?  $O(n)$



# Minimums algoritme

## Eksempel

### Algorithm 5

Input: En liste  $X = [x_1, x_2, \dots, x_n]$

Output: Det mindste tal i listen.

---

$\text{min} = x_1$

**for**  $x_i$  in  $X$  **do**

**if**  $x_i < \text{min}$  **then**

$\text{min} = x_i$

**end if**

**end for**

---

## Analyse af algoritmen

Køretid?  $O(n)$

Er den optimal?



# Minimums algoritme

## Eksempel

### Algorithm 6

Input: En liste  $X = [x_1, x_2, \dots, x_n]$

Output: Det mindste tal i listen.

---

$\text{min} = x_1$

**for**  $x_i$  in  $X$  **do**

**if**  $x_i < \text{min}$  **then**

$\text{min} = x_i$

**end if**

**end for**

---

## Analyse af algoritmen

Køretid?  $O(n)$

Er den optimal? **Jeps!**



# Eksempler på køretid

## Bogo Sort

- Køretid på  $O(n!)$

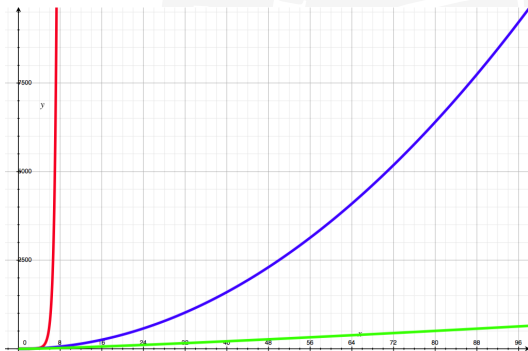
## Insertion Sort

- Køretid?  $O(n^2)$

## Merge sort

- Køretid på  $O(n \lg n)$

Figure: Graf over køretider



# Hvordan finder man på en algoritme

## Algoritme for algoritmer

- 1 Beskriv problemet med egne ord.



# Hvordan finder man på en algoritme

## Algoritme for algoritmer

- 1 Beskriv problemet med egne ord.
- 2 Del problemet op i mindre dele.





# Hvordan finder man på en algoritme

## Algoritme for algoritmer

- 1 Beskriv problemet med egne ord.
- 2 Del problemet op i mindre dele.
- 3 Definer output



# Hvordan finder man på en algoritme

## Algoritme for algoritmer

- 1 Beskriv problemet med egne ord.
- 2 Del problemet op i mindre dele.
- 3 Definer output
- 4 Definer input



# Hvordan finder man på en algoritme

## Algoritme for algoritmer

- 1 Beskriv problemet med egne ord.
- 2 Del problemet op i mindre dele.
- 3 Definer output
- 4 Definer input
- 5 Beskriv trin for at gå fra input til output



# Øvelser

## Algoritme for øvelserne

- 1 Der præsenteres et problem med eksempler.



# Øvelser

## Algoritme for øvelserne

- 1 Der præsenteres et problem med eksempler.
- 2 I finder på en algoritme for problemet (Arbejd gerne sammen)



# Øvelser

## Algoritme for øvelserne

- 1 Der præsenteres et problem med eksempler.
- 2 I finder på en algoritme for problemet (Arbejd gerne sammen)
- 3 Vi løser den sammen på tavlen.



# Øvelser

## Algoritme for øvelserne

- 1 Der præsenteres et problem med eksempler.
- 2 I finder på en algoritme for problemet (Arbejd gerne sammen)
- 3 Vi løser den sammen på tavlen.



# Søgning

## Mål

Givet en sorteret liste og et element, bestem om element er i listen, ved at kigge på så få elementer som muligt.





# Søgning

## Mål

Givet en sorteret liste og et element, bestem om element er i listen, ved at kigge på så få elementer som muligt.

## Eksempel

Lad en liste være givet ved  $[2, 4, 5, 7, 8, 11, 25]$ , hvor vi ønsker at finde ud af om elementet 11 er i listen. Svaret skulle gerne være ja. (det første element har indeks 0).



# Sortering

## Mål

Sorter en givet usorteret liste.



# Sortering

## Mål

Sorter en givet usorteret liste.

## Eksempel

Lad en liste være givet ved  $[7, 4, 5, 12, 1]$ , denne vil vi gerne sortere! Den sorterede liste skulle gerne være  $[1, 4, 5, 7, 12]$ .



# Nøgle gåde

Hvordan holder jeg min garagedør lukket?

I har en gargeport med en IR modtager der kan modtage et signal og en nøglering der kan sende et signal.



# Nøgle gåde

## Hvordan holder jeg min garagedør lukket?

I har en gargeport med en IR modtager der kan modtage et signal og en nøglering der kan sende et signal.

## Mål

Hvordan kan vi gøre den sikker? Hvad skal "Computeren" i nøgleringen kunne?



# Nøgle gåde

## Hvordan holder jeg min garagedør lukket?

I har en gargeport med en IR modtager der kan modtage et signal og en nøglering der kan sende et signal.

## Mål

Hvordan kan vi gøre den sikker? Hvad skal "Computeren" i nøgleringen kunne?

## Eksempel

Send en kode til IR modtageren?



# Fibonacci tal

## Eksempel

Det  $n$ 'te *Fibonacci* tal er defineret som summen af de to forgående.

$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \dots$



# Fibonacci tal

## Eksempel

Det  $n$ 'te *Fibonacci* tal er defineret som summen af de to forgående.

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \dots$$

Hvordan bestemmer vi dem?





# Sommer i Sunny beach

## Beskrivelse

Det er sommer i *Sunny Beach* og en kæde af barer langs kysten har et problem,



# Sommer i Sunny beach

## Beskrivelse

Det er sommer i *Sunny Beach* og en kæde af barer langs kysten har et problem, de kan ikke bestille flere øl!



# Sommer i Sunny beach

## Beskrivelse

Det er sommer i *Sunny Beach* og en kæde af barer langs kysten har et problem, de kan ikke bestille flere øl!

Hver bar langs strandvejen har  $b_i$  øl tilbage, der er ikke nogen der ved hvilken bar der kan sælge flest øl på en given aften.



# Sommer i Sunny beach

## Beskrivelse

Det er sommer i *Sunny Beach* og en kæde af barer langs kysten har et problem, de kan ikke bestille flere øl!

Hver bar langs strandvejen har  $b_i$  øl tilbage, der er ikke nogen der ved hvilken bar der kan sælge flest øl på en given aften. Det er besluttet at alle barer skal have lige mange øl.



# Sommer i Sunny beach

## Beskrivelse

Det er sommer i *Sunny Beach* og en kæde af barer langs kysten har et problem, de kan ikke bestille flere øl!

Hver bar langs strandvejen har  $b_i$  øl tilbage, der er ikke nogen der ved hvilken bar der kan sælge flest øl på en given aften. Det er besluttet at alle barer skal have lige mange øl.

De har en lastbil hvor der kan være en uendelig mængde af øl,



# Sommer i Sunny beach

## Beskrivelse

Det er sommer i *Sunny Beach* og en kæde af barer langs kysten har et problem, de kan ikke bestille flere øl!

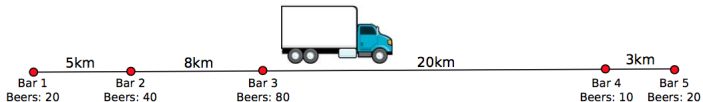
Hver bar langs strandvejen har  $b_i$  øl tilbage, der er ikke nogen der ved hvilken bar der kan sælge flest øl på en given aften. Det er besluttet at alle barer skal have lige mange øl.

De har en lastbil hvor der kan være en uendelig mængde af øl, dog er vognførerne på denne lastbil glade for øl. Hver gang der er kørt en kilometer så drikkes der to øl.



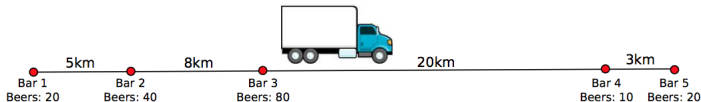
# Eksempel

## Figur



# Eksempel

## Figur



## Hvad i får som input

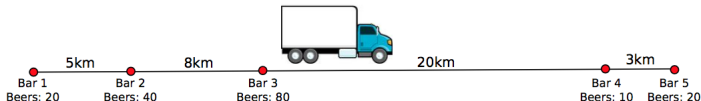
- En liste over hvor mange øl der er på hver bar  $b_1, b_2, \dots$
- En liste over afstanden mellem dem  $a_1, a_2, \dots$





# Eksempel

## Figur



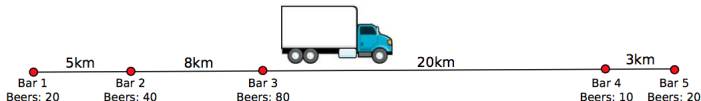
## Hvad i får som input

- En liste over hvor mange øl der er på hver bar  $b_1, b_2, \dots$
- En liste over afstanden mellem dem  $a_1, a_2, \dots$
- Det antal øl de gerne vil have på hver bar



# Eksempel

## Figur



## Hvad i får som input

- En liste over hvor mange øl der er på hver bar  $b_1, b_2, \dots$
- En liste over afstanden mellem dem  $a_1, a_2, \dots$
- Det antal øl de gerne vil have på hver bar

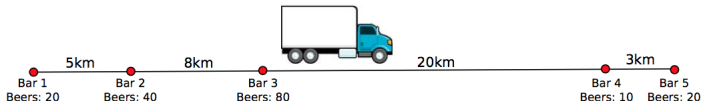
## Hvad i skal svare

- Ja, eller nej.



# Eksempel

## Figur



## Hvad i får som input

- En liste over hvor mange øl der er på hver bar  $b_1, b_2, \dots$
- En liste over afstanden mellem dem  $a_1, a_2, \dots$
- Det antal øl de gerne vil have på hver bar

## Hvad i skal svare

- Ja, eller nej.



# Spørgsmål

*Nogen spørgsmål?*

