UNIVERSITETET
I OSLO

IN2010: Forelesning 7 – Sortering I

Ragnhild Kobro Runde

Sorteringsalgoritmer i IN2010

- Boblesortering (bubble-sort)
- Selection-sort
- **Instikksortering** (insertion-sort)
- Heapsortering

- Flettesortering (merge-sort)
- **Quicksort**

- Bøttesortering (bucket-sort)
- **Radix-sortering**

Krav til en sorteringsalgoritme

- Riktig
- Rask
  - For alle typer fordelinger
  - For alle datatyper (int, double, tekst, …)
- Stabil
- Bruker lite ekstra plass

Hva avgjør tidsforbruket ved sortering

- Sorteringsalgoritmen
- Antall elementer
- Fordelingen av elementene
- Effekten av caching
- Optimalisering av jvm eller tilsvarende
- Muligheter for parallellisering

Definisjon sortering

Ønsker å sortere en array A med n elementer som kan sammenlignes.

To krav til sluttresultatet:

1. Etter sorteringen er A[i] ≤ A[i+1] for alle i.

2. Elementene i A er de samme før og etter sorteringen.

Hvorfor ikke bare bruke Arrays.sort?

UNIVERSITETET I OSLO

- Ulike varianter av Arrays.sort
- Tilfeller hvor egen implementasjon er raskere?
- Når vi sorterer andre ting enn arrayer
- Språk som ikke har sortering innebygget

- «Allmenndannelse»
- Programmeringstrening
- Generelle prinsipper
- Trening i O-notasjon

- Sorteringsalgoritmer er gøy ☺

UNIVERSITETET I OSLO

Øvelse 1

Gå sammen to og to. Hvert par får en kortstokk.

1. Stokk kortene.

2. Del ut 10 kort til hver person.
3. Sorter de 10 kortene basert på tallverdi, uten hensyn til farge.
4. Forklar hverandre hvilken metode dere brukte for å sortere kortene.

Sorterte dere kortene på samme måte?

Øvelse 2

1. Stokk kortene.

2. Person 1:
   Sorter kortene etter farge og tall.

   Person 2:
   Legg merke til hvilken metode person 1 bruker.

Sorterte dere kortene på samme måte som i øvelse 1?

«Lego-sortering»

Min variant:

```
while (ikke ferdig sortert)
    gå til den første blokken
    while (ikke står på siste blokk)
        if (neste blokk er lavere enn denne)
            bytt denne og neste blokk
        else
            gå videre til neste blokk
```

Boblesortering

https://pixabay.com/en/people-group-crowd-line-silhouette-312122/

NB!

Java: Kan implementeres vha grensesnittet **`Comparable`** og metoden **`compareTo`**.

```
Input:  An array A of n comparable elements,
        indexed from 1 to n.
Output: An ordering of A so that its elements are
        in nondecreasing order.

Algorithm BubbleSort(A):
  for i ← 1 to n-1 do
      for j ← n down to i+1 do
          if A[j-1] > A[j] then
              swap A[j-1] and A[j]
  return A
```

Hva blir kjøretiden (i O-notasjon)?

1	2	3	4	5	6	7	8	9	10

Selection-sort

```
Algorithm SelectionSort(A):
  for i ← 1 to n-1 do
      s ← i
      for j ← i+1 to n do
          if A[j] < A[s] then
              s ← j
      swap A[i] and A[s]
  return A
```

Hva blir kjøretiden (i O-notasjon)?

1	2	3	4	5	6	7	8	9	10

Innstikk-sortering


```
Algorithm InsertionSort(A):
  for i ← 2 to n do
      x ← A[i]
      j ← i
      while j > 1 and x < A[j-1]
          A[j] ← A[j-1]
          j ← j-1
      A[j] ← x
  return A
```

Hva blir kjøretiden (i O-notasjon)?

1	2	3	4	5	6	7	8	9	10

«PQ-sort»

Sortering ved hjelp av (prioritets)kø (s. 158):

1. Legg alle elementene inn i en tom (prioritets)kø.
2. Ta ut elementene en etter en ved hjelp av removeMin.

Tre varianter:

- Prioritetskø som usortert liste
  ≈ Selection Sort
- Prioritetskø som sortert liste
  ≈ Instikksortering
- Prioritetskø som heap
  ≈ Heapsortering

Heapsortering

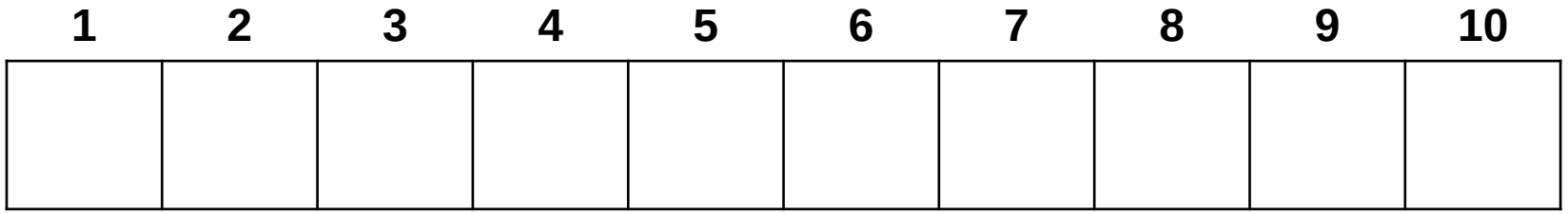
Krever i utgangspunktet den ekstra datastrukturen til en heap.

Men:

- Hver gang et element fjernes fra heapen blir det en ledig plass i heap-arrayen. Denne plassen kan brukes til elementet som ble fjernet!
- Dette vil gi en array som er sortert med det største elementet først.
- For å få til sortering med minste element først, kan vi bruke en maksimums-heap og deleteMax i stedet.

Heapsortering (maxheap)

```
Algorithm HeapSort(A):
  for i ← n/2 down to 1 do
      downHeap(a,i,n)
  for i ← n down to 2 do
      swap A[i] and A[1]
      downHeap(a,1,i-1)
  return A
```



Stabile sorteringsalgoritmer

Noen ganger ønsker vi at like elementer skal beholde sin innbyrdes rekkefølge etter sortering.

Når?

Formelt krav:

- Gitt elementer på formen (k, e) hvor k er nøkkelen det skal sorteres etter og e er selve elementet.
- Hvis $k_i = k_j$ og (k_i, e_i) kommer foran (k_j, e_j) før sorteringen (dvs $i < j$), så skal (k_i, e_i) komme foran (k_j, e_j) også etter sorteringen.

Neste forelesning: 19. oktober

SORTERING II