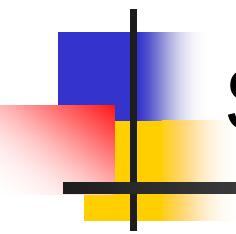


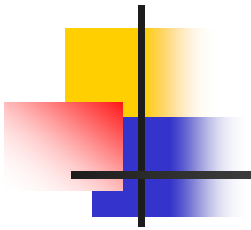
INF1000 (Uke 12)
Sortering og eksamensoppgaver

Grunnkurs i programmering
Institutt for Informatikk
Universitet i Oslo

Are Magnus Bruaset og Anja B. Kristoffersen

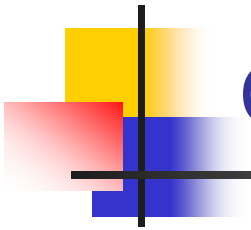
Orakel-tjeneste for Oblig 4

- Orakel på Abelstua, N.H. Abels hus:
  - Onsdag 26. april: 10 – 13
    - Anne (10-12.30) og Magnus (10 -13)
  - Torsdag 27. april: 12 – 15
    - Anne og Magnus



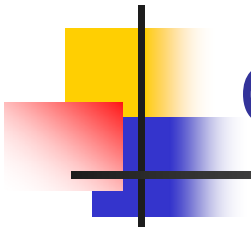
Innhold

- Om sortering
  - Sortering av heltall og tekster
- Litt om dokumentasjon av kode
- Deler av eksamen H03



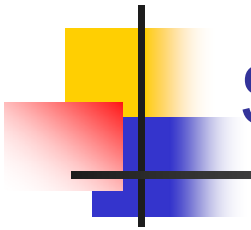
Oversikt I

- Lære å løse et vanskelig problem
- Sortering – mange metoder, her Innstikksortering
  - Sortere hva:
    - Heltall
    - Tekster
    - En tabell (2-dim) etter verdiene i første kolonne
      - Eks : telefonkatalogen (sortert på navn)



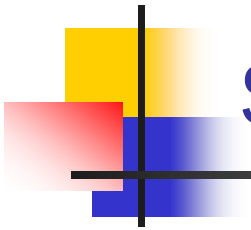
Oversikt II

- Lære abstraksjon
  - Når vi har løst ett problem, kan lignende problemer løses tilsvarende
- Lære å lage ”proff” programvare ved å lage en generell klasse for sortering
  - Hvordan deklarere en slik klasse
  - Javadoc – lage dokumentasjon
  - Testing
  - Hvordan utvikle programmet



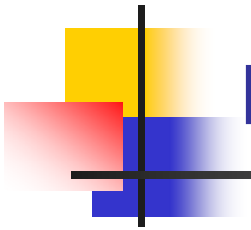
Sortering

- Mange datatyper kan sorteres
  - Krav: operatorer som <, <=, ==, >, >=, != må ha mening
- Eksempler
  - Tall
  - Tekster (leksikografisk = i samme rekkefølge de ville stått i et leksikon)
  - Tabeller av tekster eller tall



Sortering

- Vi må ha en algoritme (oppskrift) for sortering
  - Det finns mange titalls (hundretalls) metoder å velge blant
  - Vi skal se på innstikksortering
  - Dette er den raskeste metoden når vi skal sortere få elementer (typisk færre enn 50 elementer)

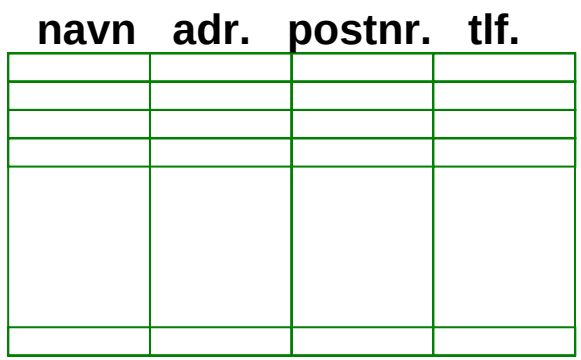


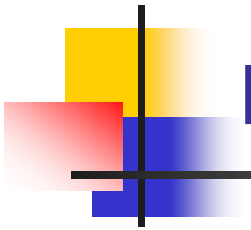
Hvorfor sorterer vi

- For å få noen tall i en bestemt (stigende eller synkende) rekkefølge
  - Eksempel: lotto-tallene
- Sortere tekster (navnelister) for raskere oppslag
- Sortere et sett av opplysninger som hører sammen, ved å sortere på en av opplysningene
  - Eksempel: Telefonkatalogen (navn, adresse, telefonnummer – informasjonen sortert på navn)

Vi skal først lære å sortere heltall

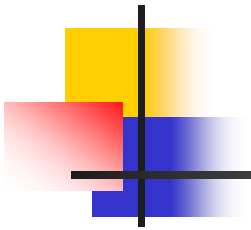
- Dette skal vi (med minimale endringer) bruke til å sortere:
  - String-arrayer (tekster)
  - Sammenhengende opplysninger i en 2-dim array av tekster (hver linje er opplysninger om ett objekt)
    - Eks : Telefonkatalogen





En felles klasse for sortering

- Vi ønsker en klasse med tre varianter av sortering:
  - Heltall
  - Tekster
  - To-dimensjonal tekst-arrays (sortert på data i 1. kolonne)

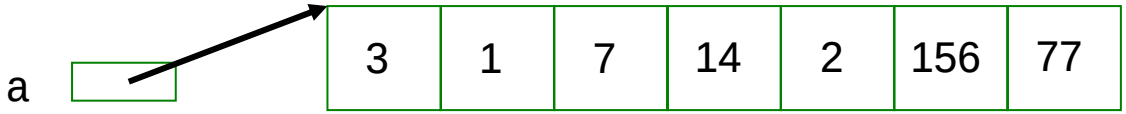


Class ISort

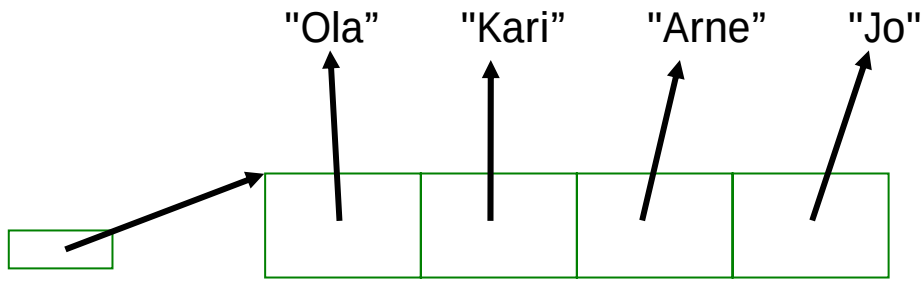
```
public class ISort {

  public static void sorter(int [] a) {

  }
  public static void sorter(String [] a) {

  }
  public static void sorterEtterKol1(String [] [] a) {

  }
} // end class ISort
```

```
class TestInnstikkSortering
{

  public static void  main ( String[] args)  {

      int [] a = {3,1,7,14,2,156,77};
      String [] navn = {"Ola", "Kari", "Arne", "Jo"};
      String [][] telefonliste = { {"Per","22852451"},
                                   {"Arne","33445566"},
                                   {"Kari","44452611"},
                                   {"Jo","55010102"}};

        // sorter heltall - skriv ut
        ISort.sorter(a);
        for (int i = 0; i < a.length; i++)
          System.out.println("b[" + i +"]= " + a[i]);

          System.out.println("\n Test tekst-sortering:");

        // sorter Stringer - skriv ut
        ISort.sorter(navn);
        for (int i = 0; i < navn.length; i++)
          System.out.println("navn[" + i +"]= " + navn[i]);

          System.out.println("\n Test 2dim tekst-sortering:");

         // sorter Tabell - skriv ut
        ISort.sorterEtterKol1(telefonliste);
        for (int i = 0; i < navn.length; i++)
          System.out.println("navn[" + i +"]= " + telefonliste[i][0]
                             + ", med tlf.: " + telefonliste[i][1] );


}}
```

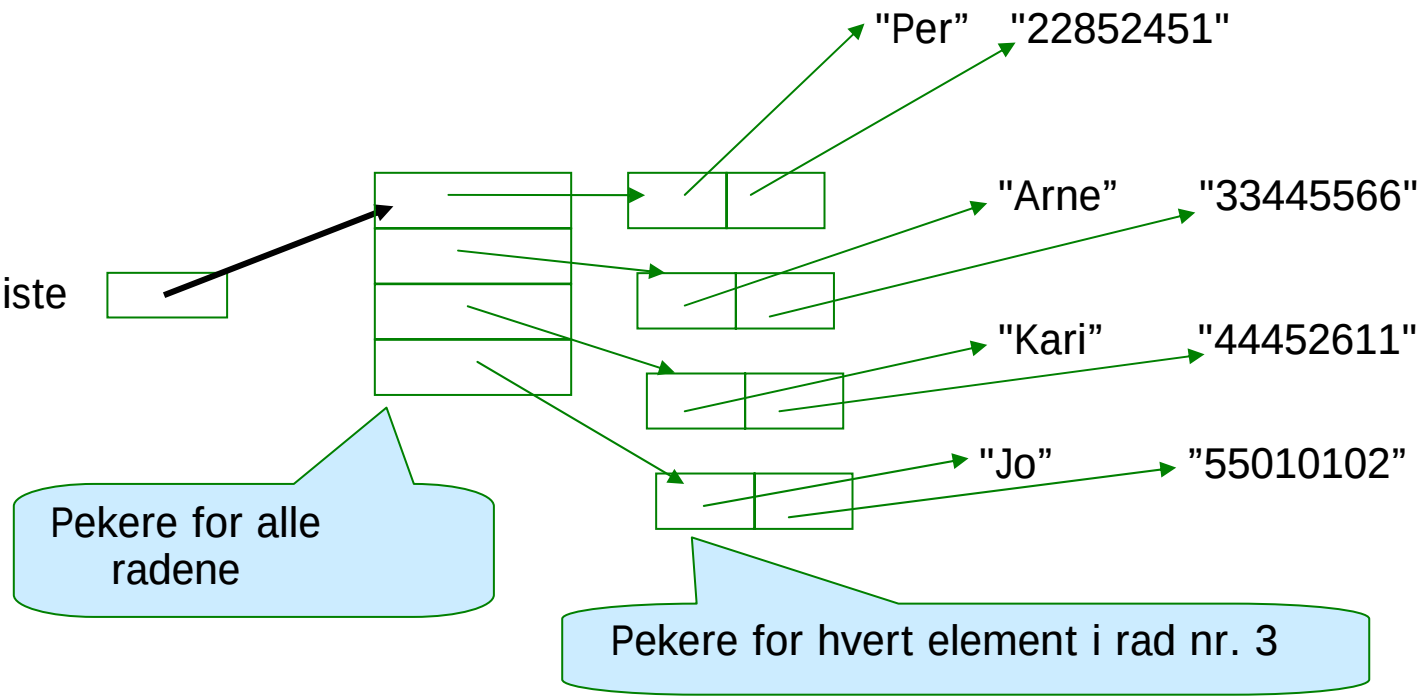
Heltalls-array



En-dimensjonal String-array



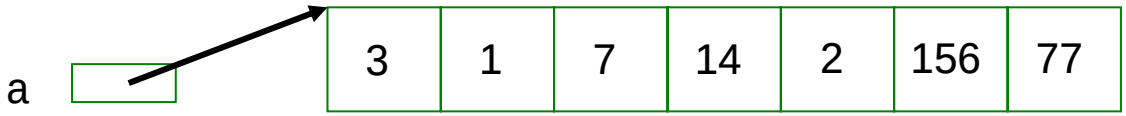
To-dimensjonal String-array



Test av test-programmet med **tomme** sortering-metoder

```
>java InnstikkSortering
b[0]= 3
b[1]= 1
b[2]= 7
b[3]= 14
b[4]= 2
b[5]= 156
b[6]= 77

 Test tekst-sortering:
navn[0]= Ola
navn[1]= Kari
navn[2]= Arne
navn[3]= Jo

 Test 2dim tekst-sortering:
telefonliste[0]= Per, med tlf.: 22852451
telefonliste[1]= Arne, med tlf.: 33445566
telefonliste[2]= Kari, med tlf.: 44452611
telefonliste[3]= Jo, med tlf.: 55010102
```

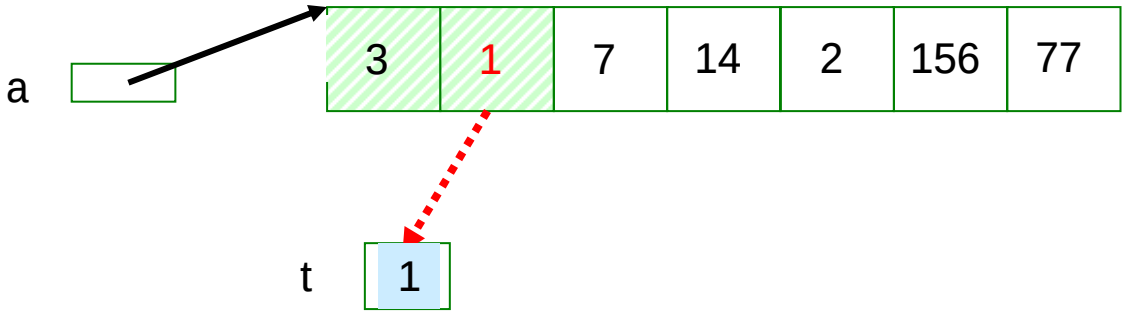
Sortering av heltall – innstikksmetoden



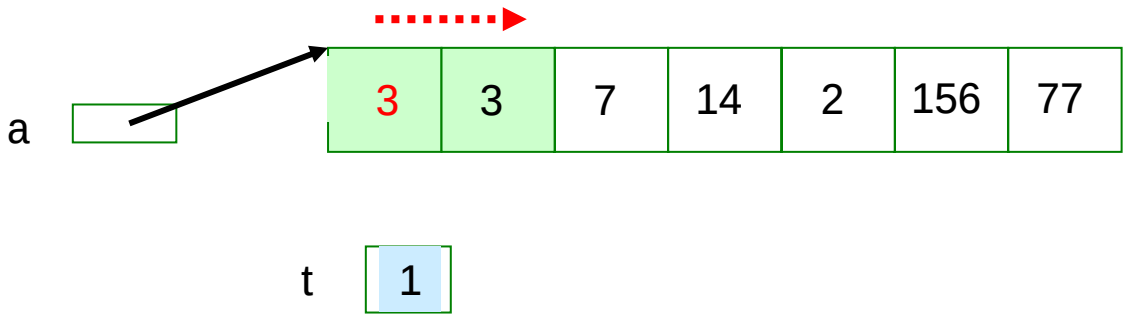
- Se på arrayen element for element fra venstre
- Sorterer det vi hittil har sett på ved :
  - Hvis det nye elementet vi ser på **ikke** er sortert i forhold til de vi allerede har sett på:
    - Ta ut dette elementet (gjem verdien i en variabel **`t`**)
    - Skyv de andre elementene vi her sett på, en-etter-en, ett hakk til høyre. Slutt når elementet i **`t`** kan settes inn på sortert plass
    - Den sorterte delen er nå ett element lenger (sett fra venstre)
  - Når vi har sett på alle elementene, er hele arrayen sortert

Sorter 1 på plass i forhold til 3

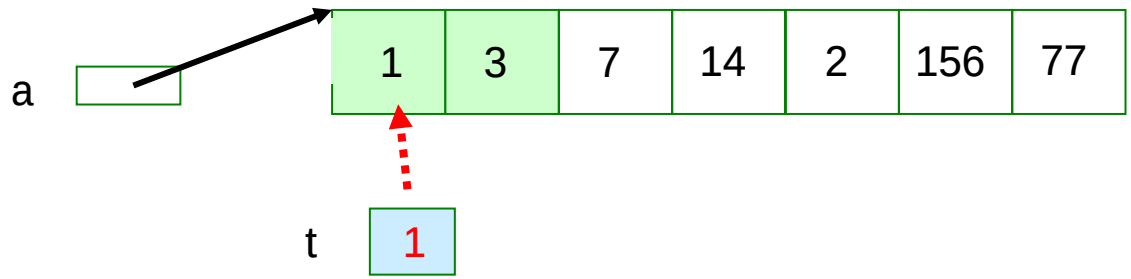
steg 1



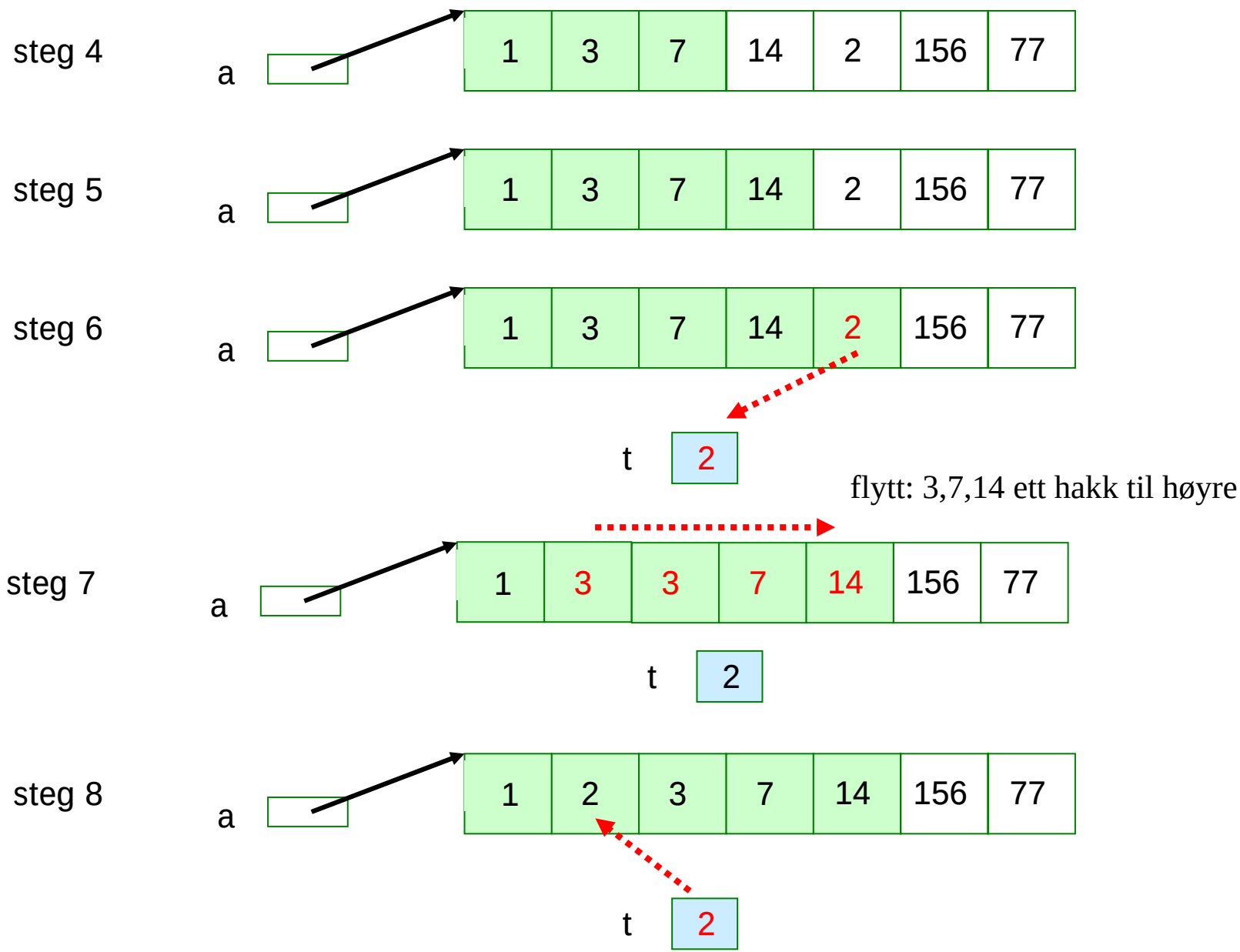
steg 2

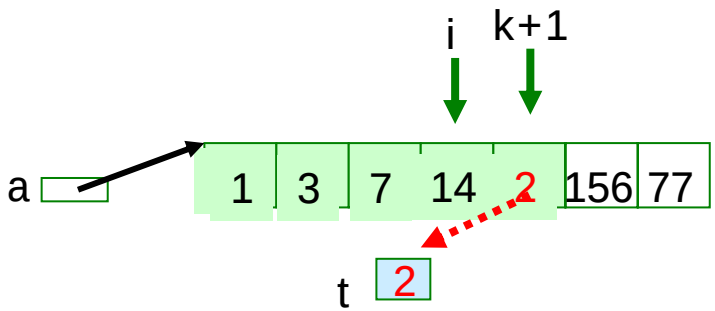


steg 3



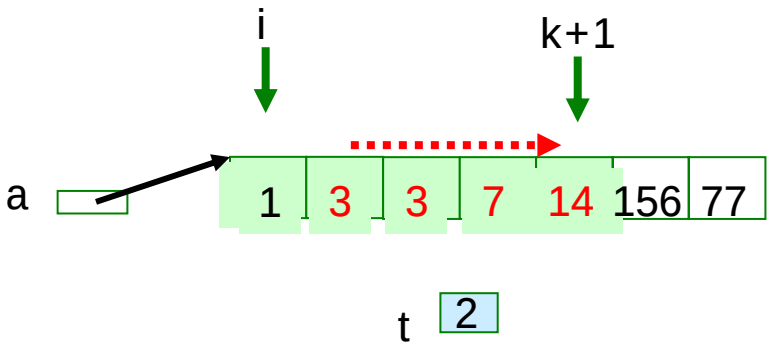
7 og 14 står riktig, Sorter 2 på plass i forhold til : 1,3,7,14



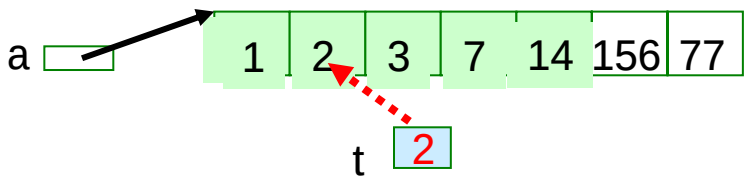


Kode for å flytte **ett** element på plass :

```
// a[k +1 ] står muligens på
// feil plass, ta den ut
int t = a[k + 1], i = k;
```




```
// skyv a[i] mot høyre ett hakk til
// vi finner riktig plass til t
while (i >= 0 && a[i] > t) {
    a[i + 1] = a[i];
    i--;
}
```



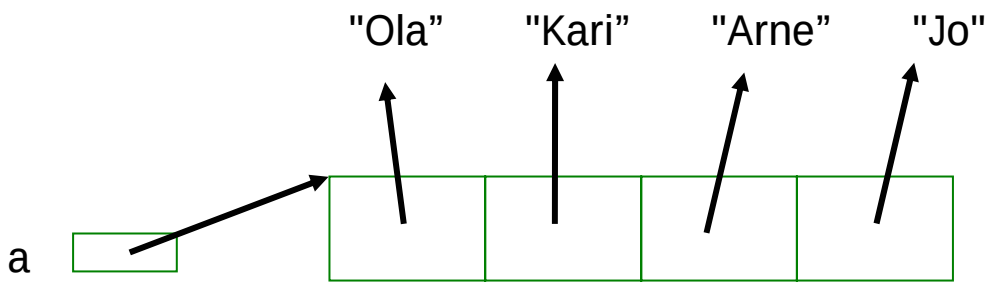

```
// sett t inn på riktig plass
  a[i + 1] = t;
```

```
public class ISort {


  public static void sorter(int [] a) {
       for (int k = 0 ; k < a.length-1; k++) {
              // a[k +1 ] står muligens på feil plass, ta den ut
              int t = a[k + 1], i = k;
              // skyv a[i] mot høyre ett hakk til
              // vi finner riktig plass til t
              while (i >= 0 && a[i] > t) {
                     a[i + 1] = a[i];
                     i--;
              }
              // sett t inn på riktig plass
              a[i + 1] = t;
        }
  } // end heltall-sortering
```

```
>java InnstikkSortering
b[0]= 1
b[1]= 2
b[2]= 3
b[3]= 7
b[4]= 14
b[5]= 77
b[6]= 156

 Test tekst-sortering:
navn[0]= Ola
navn[1]= Kari
navn[2]= Arne
navn[3]= Jo

 Test 2dim tekst-sortering:
telefonliste[0]= Per, med tlf.: 22852451
telefonliste[1]= Arne, med tlf.: 33445566
telefonliste[2]= Kari, med tlf.: 44452611
telefonliste[3]= Jo, med tlf.: 55010102
```

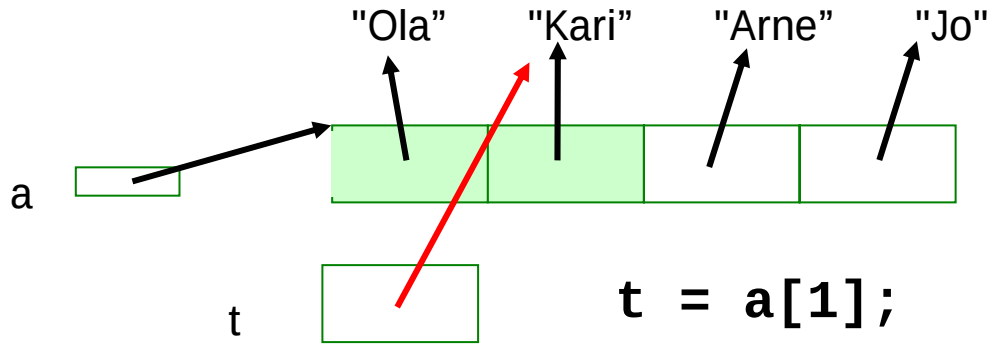
Resultat av sortering med heltalls-metoden kodet, de to andre tomme

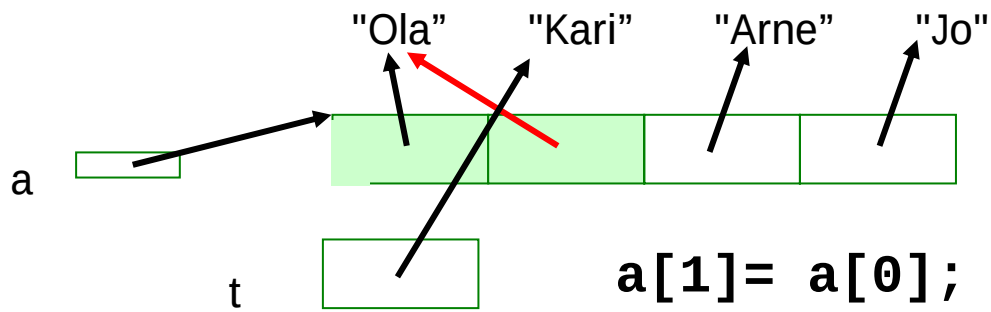
Sortering av tekster (String)

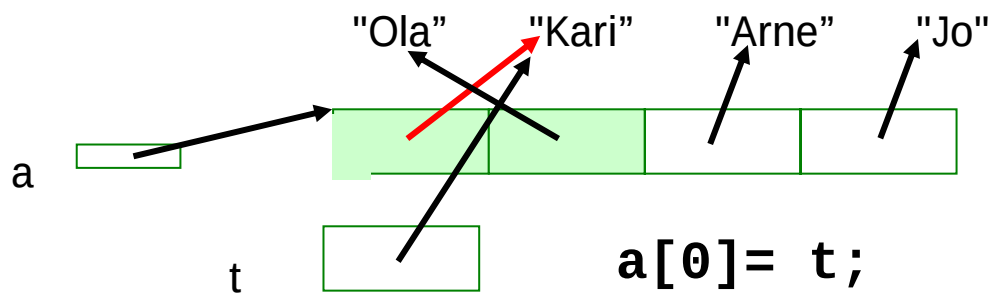


- Vi skal sortere denne ved å bytte om på pekerne (la a[0] peker på ”Arne”, osv.) med innstikkmetoden

Sortere de to første elementene ved å bytte om pekere





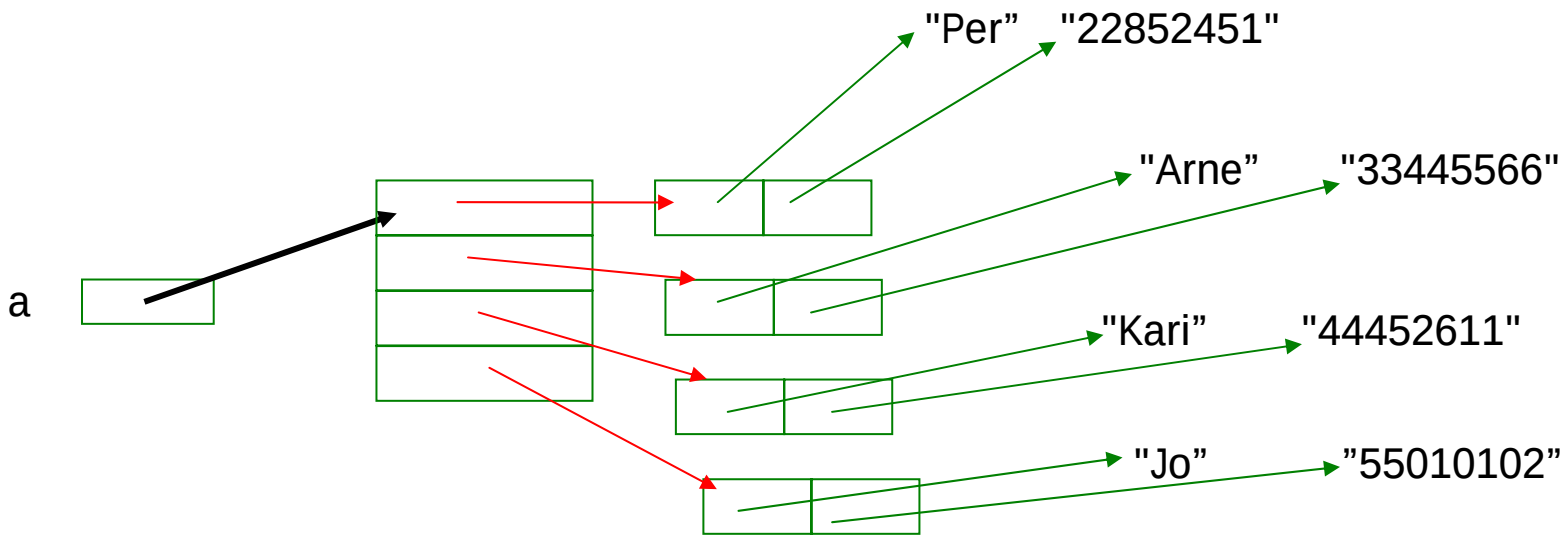


```
public static void sorter(int [] a) {
       // Sorterer heltallsarrayaen 'a'.
       for (int k = 0 ; k < a.length-1; k++) {
              int t = a[k + 1], i = k;
              while (i >= 0 && a[i] > t) {
                     a[i + 1] = a[i];
                     i--;
              }
              a[i + 1] = t;
       }
  } // end heltall-sortering

  public static void sorter(String [] a) {
       // Sorterer String-arrayen 'a'.
       for (int k = 0 ; k < a.length-1; k++) {
              String t = a[k + 1];
              int i = k;
              while (i >= 0 &&  ( a[i].compareTo(t) > 0) ){
                     a[i + 1] = a[i];
                     i--;
              }
        a[i + 1] = t;
       }
  } // end String-sortering
```

```
>java InnstikkSortering
b[0]= 1
b[1]= 2
b[2]= 3
b[3]= 7
b[4]= 14
b[5]= 77
b[6]= 156

 Test tekst-sortering:
navn[0]= Arne
navn[1]= Jo
navn[2]= Kari
navn[3]= Ola

 Test 2dim tekst-sortering:
telefonliste[0]= Per, med tlf.: 22852451
telefonliste[1]= Arne, med tlf.: 33445566
telefonliste[2]= Kari, med tlf.: 44452611
telefonliste[3]= Jo, med tlf.: 55010102
```

Test med heltall og enkel String-sortering kodet, 2dim sortering tom

Sortering av 2-dim String array

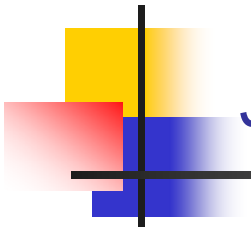


- Vi kan sortere denne på to måter:
  - Bytte om på pekerne til radene (la a[0] peker på ”Arne”-raden,..osv)
    - Enklest
  - Bytte om på pekerne til hvert element i hver rad
    - Mye mer arbeid, vanskeligere kode, langsommere

```
public static void sorter(String [] a) {
        // Sorterer String-arrayen 'a'.
        for (int k = 0 ; k < a.length-1; k++) {
                String t = a[k + 1];
                int i = k;
                while (i >= 0 &&  ( a[i].compareTo(t) > 0) ){
                        a[i + 1] = a[i];
                        i--;
                }
                 a[i + 1] = t;
         }
  } // end String-sortering

  public static void sorterEtterKol1(String [] [] a) {
        // Sorterer den 2-dim String-arrayen 'a' ettter verdi i kol.1.
        for (int k = 0 ; k < a.length-1; k++) {
                String [] tRad = a[k + 1];
                int i = k;
                while (i >= 0 && ( a[i][0].compareTo(tRad[0]) > 0) ){
                        a[i + 1] = a[i];
                        i--;
                }
         a[i + 1] = tRad;
       }
    } // end 2-dim String-sortering
```

```
M:\INF1000\prog2>java InnstikkSortering
b[0]= 1
b[1]= 2
b[2]= 3
b[3]= 7
b[4]= 14
b[5]= 77
b[6]= 156

 Test tekst-sortering:
navn[0]= Arne
navn[1]= Jo
navn[2]= Kari
navn[3]= Ola

 Test 2dim tekst-sortering:
telefonliste[0]= Arne, med tlf.: 33445566
telefonliste[1]= Jo, med tlf.: 55010102
telefonliste[2]= Kari, med tlf.: 44452611
telefonliste[3]= Per, med tlf.: 22852451
```

Alle sorterings metodene skrevet



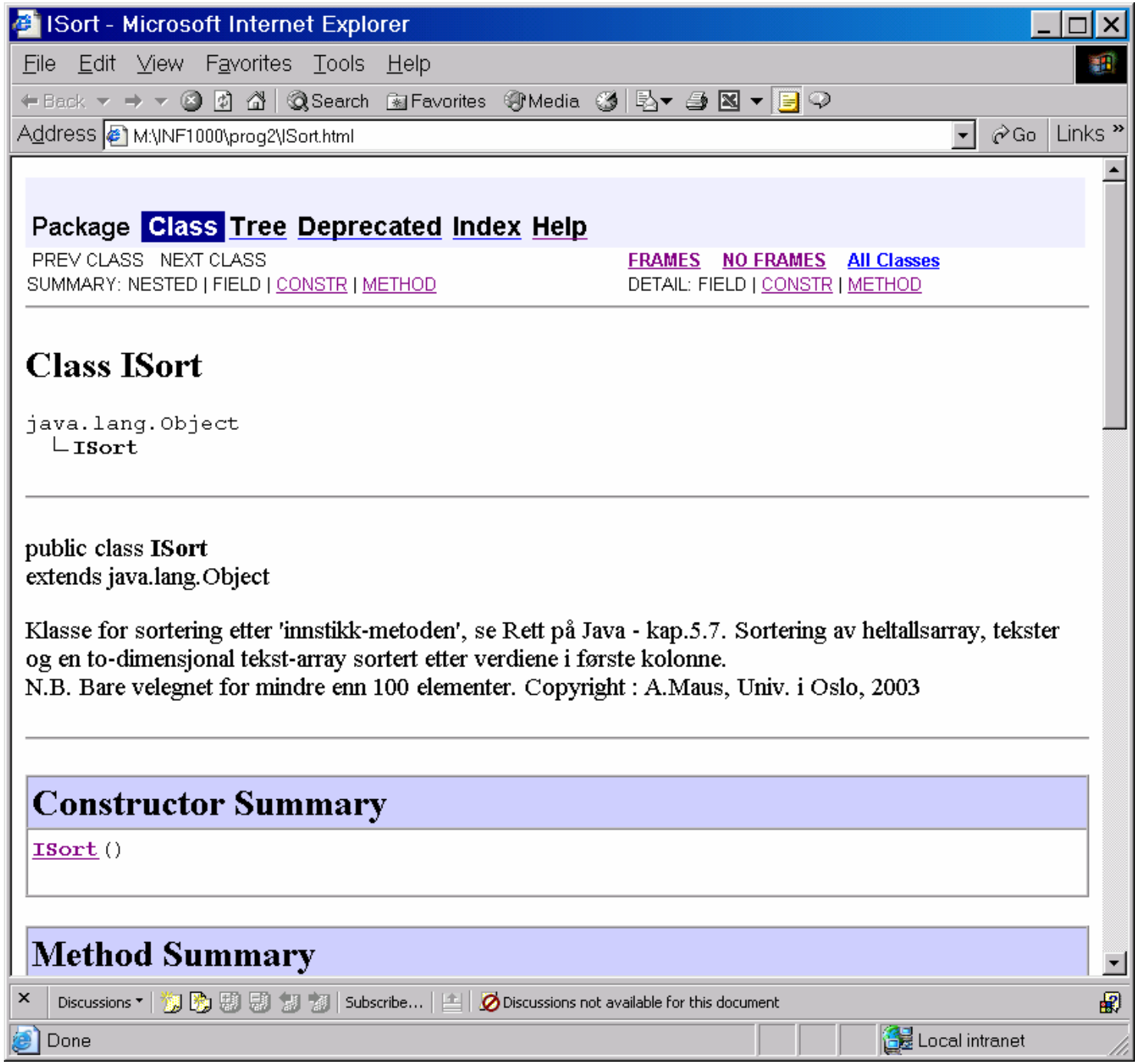
Javadoc – proff dokumentasjon av klassene

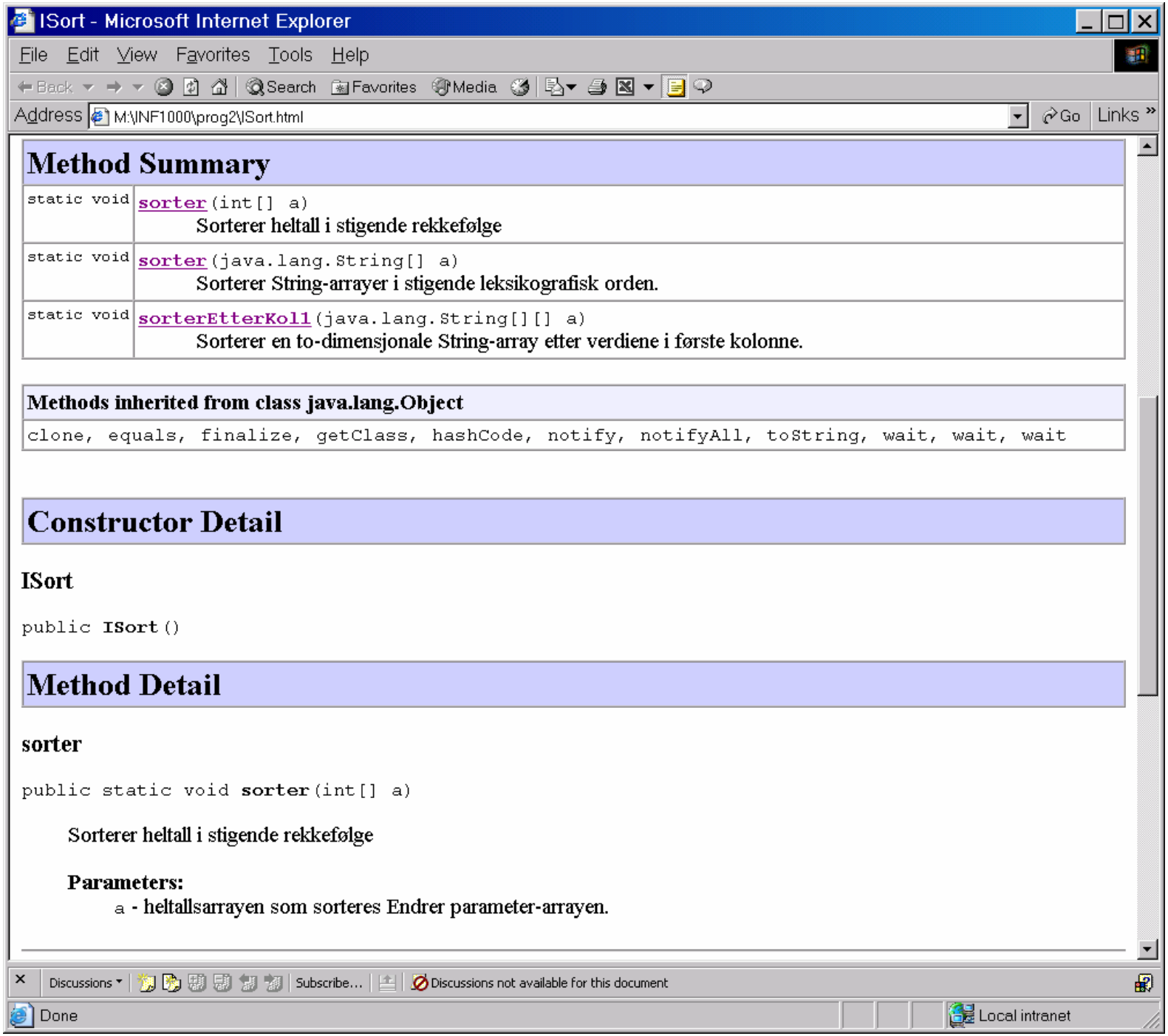
- Legg inn spesielle kommentarer i programmet ditt (over hver metode og klasse)
- Kjør programmet ’javadoc’ som automatisk genererer en oversiktlig dokumentasjon

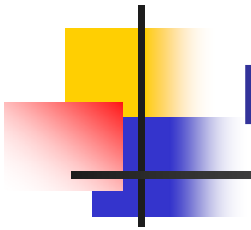
```java
/**
* Klasse for sortering etter 'innstikk-metoden', se
* Rett på Java - kap 5.7.
* Sortering av heltallsarray, tekster og en to-dimensjonal
* tekst-array  sortert etter verdiene i første kolonne.<br>
*
* N.B. Bare velegnet for mindre enn 100 elementer.
*
* Copyright : A.Maus, Univ. i Oslo, 2003
*************************************************/
public class ISort {

  /**
  * Sorterer heltall i stigende rekkefølge
  * @param a heltallsarrayen som sorteres
  * Endrer parameter-arrayen.
  ****************************************/
  public static void sorter(int [] a) {
  }
  /**
  * Sorterer String-arrayer i stigende leksikografisk orden.
  * @param a arrayen som sorteres
  * Endrer parameter-arrayen
  ****************************************/
  public static void sorter(String [] a) {

  }

  /**
  * Sorterer en  to-dimensjonale String-array
  * etter verdiene i første kolonne.
  * Nytter pekerombytting av radpekerne.
  * Antar at  alle radene har minst ett element
  * @param a en to-dimensjonal array som sorteres
  * Endrer parameter-arrayen.
  ******************************************************/
  public static void sorterEtterKol1(String [] [] a) {
  }
} // end class ISort
```

Dokumentasjon av klassen og metodene - javadoc

```
>javadoc ISort.java
Loading source file ISort.java...
Constructing Javadoc information...
Standard Doclet version 1.4.2
Generating constant-values.html...
Building tree for all the packages and classes...
Building index for all the packages and classes...
Generating overview-tree.html...
Generating index-all.html...
Generating deprecated-list.html...
Building index for all classes...
Generating allclasses-frame.html...
Generating allclasses-noframe.html...
Generating index.html...
Generating packages.html...
Generating ISort.html...
Generating package-list...
Generating help-doc.html...
Generating stylesheet.css...
```

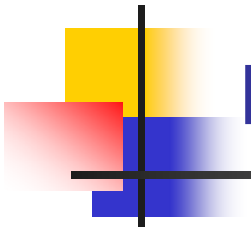

ISort - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address M:\INF1000\prog2\ISort.html
Go Links
Package Class Tree Deprecated Index Help
PREV CLASS NEXT CLASS
FRAMES NO FRAMES All Classes
SUMMARY: NESTED | FIELD | CONSTR | METHOD
DETAIL: FIELD | CONSTR | METHOD
Class ISort
java.lang.Object
└ ISort
public class ISort
extends java.lang.Object
Klasse for sortering etter 'innstikk-metoden', se Rett på Java - kap.5.7. Sortering av heltallsarray, tekster og en to-dimensjonal tekst-array sortert etter verdiene i første kolonne.
N.B. Bare velegnet for mindre enn 100 elementer. Copyright : A.Maus, Univ. i Oslo, 2003
Constructor Summary
ISort ()
Method Summary
Discussions Subscribe... Discussions not available for this document
Done
Local intranet


ISort - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address M:\INF1000\prog2\ISort.html
Go Links
Method Summary
static void sorter(int[] a)
Sorterer heltall i stigende rekkefølge
static void sorter(java.lang.String[] a)
Sorterer String-arrayer i stigende leksikografisk orden.
static void sorterEtterKol1(java.lang.String[][] a)
Sorterer en to-dimensjonale String-array etter verdiene i første kolonne.
Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait
Constructor Detail
ISort
public ISort()
Method Detail
sorter
public static void sorter(int[] a)
Sorterer heltall i stigende rekkefølge
Parameters:
a - heltallsarrayen som sorteres Endrer parameter-arrayen.
Discussions Subscribe... Discussions not available for this document
Done
Local intranet



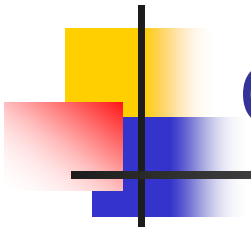
Eksamen høsten 2003

- Den store bøygen i INF1000 er de obligatoriske oppgavene
- Hovedformålet med eksamen er å skille ut de som har skjønt litt om programmering fra de som ikke har skjønt noe
- For de som har løst de obligatoriske oppgavene uten mye hjelp, bør eksamen være ganske enkel



Eksamen høsten 2003

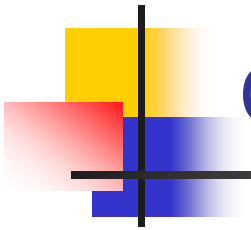
- I dag tar vi oppgave 1
- Resten av oppgavesettet gjennomgås senere



Oppgave 1.1

- Hvilke utsagn er riktige om en *variabel deklarert i en objektmetode?*
  - Før vi foretar en tilordning til variabelen har den ingen verdi
  - Den kan ha en aksessmodifikator (f eks private eller public)
  - Andre objektmetoder i samme klasse har tilgang (aksess) til variabelen
  - Objektmetoder i andre klasser har tilgang (aksess) til den via prikk-notasjon

Oppgave 1.1 – vurderinger



- **NB!** Les oppgaveteksten *nøye!*
  . . . i en objekt**metode**?

- Variable i metoder er alltid utilgjengelige utenfra.
  Derfor er det også uaktuelt med aksessmodifikator

- Som alle variable er metodevariable uten verdi i starten. De kan initieres ved deklarasjonen:

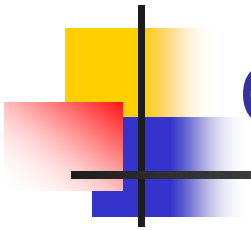
```
int x = 0;
```

  eller ved en tilordning senere

Oppgave 1.1 – svar

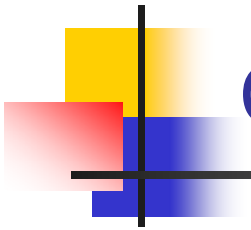
- Svaret på oppgave 1.1 er altså **alternativ 1:**

  “Før vi foretar en tilordning til variabelen har den ingen verdi”



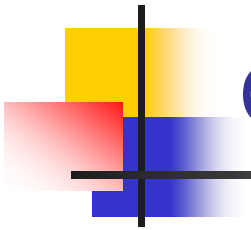
Oppgave 1.2

- Hvilke utsagn er riktige om en *objektvariabel?*
  - Før vi foretar en tilordning har den ingen verdi
  - Den kan ha en aksessmodifikator (f eks private eller public)
  - Objektmetoder i samme klasse har tilgang (aksess) til variabelen
  - Objektmetoder i andre klasser har tilgang (aksess) til den via prikk-notasjon dersom variabelen er deklarert som public



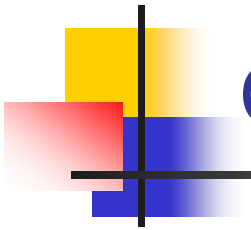
Oppgave 1.2 – vurderinger

- Vi har følgende aksessmodifikatorer:
  - Private: skjult for alle andre klasser
  - Protected: skjult for alle andre klasser (unntatt *subklasser*)
  - — kun tilgjengelig for klasser i samme *pakke*
  - public åpen for alle
- Dere vet ikke hva pakker og subklasser er; det kommer i INF1010
- Dere trenger bare å vite:
  - private skjult
  - public åpen



Oppgave 1.2 – svar

- Svaret på oppgave 1.2 er da **alle alternativene 1, 2, 3, 4**

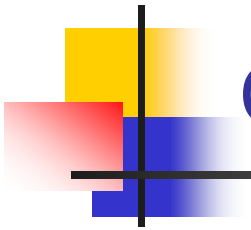


Oppgave 1.3

- Hvor mange heltall settes det av plass til (array-lengde) i setningen

```
int[] tallene = new int[100];
```

- **Alternativer: 99, 100, 101 heltall**
- **Svar: 100 heltall**

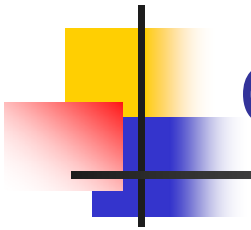


Oppgave 1.4

- Hvor mange ganger skrives ”Eksamen” ut?

```
for (int i = 0; i < 100; i++) {
  for (int j = 0; j < 99; j++) {
    System.out.println(”Eksamen”);
  }
}
```

- Svar: 100 * 99 = 9900 ganger



Oppgave 1.5

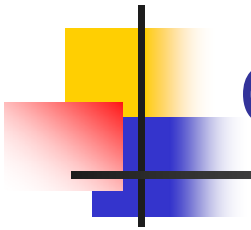
- Hvor mange ganger skrives ”INF 1000” ut?

```
for (int i = 0; i < 3; i++) {
  for (int j = 0; j <= i; j++) {
    System.out.println("INF 1000");
  }
}
```

Oppgave 1.5 – svar

i	j
0	0
1	0, 1
2	0, 1, 2

- Svar: 6 ganger



Oppgave 1.6

- Hvilken verdi har **`alder`** etter denne koden?

```
int alder = 4;
int nyalder = alder--;
// A
alder += nyalder;
// B
alder++;
// C
```

Oppgave 1.6 – svar

Utskrift	alder	nyalder
A	3	4
B	7	4
C	8	4

- Svar: `alder` har verdien 8 etter at alle setningene er utført

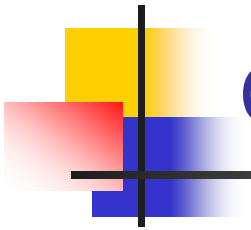
Oppgave 1.7

- Anta at vi har et program hvor en av klassene blant annet har følgende objektvariabeldeklarasjon:

```
HashMap personer = new HashMap();
```

- Klassen inneholder blant annet metoder for å legge inn objekter av klassen Person (med en passende nøkkel, f eks personnummer) i HashMap-en og for å løpe gjennom alle Person-objektene i HashMap-en. Sistnevnte metode, som skal kalle på en metode SkrivUt() i hvert av objektene i HashMap-en, ser slik ut:

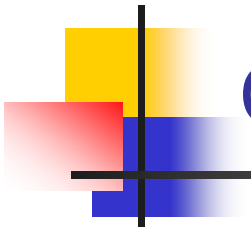
```
void skrivAlle () {
   Iterator liste = personer.values().iterator();
   while (liste.hasNext()) {
      ...
      b.skrivUt();
   }
}
```



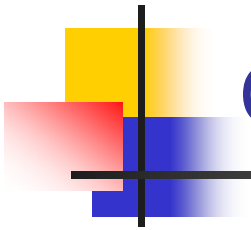
Oppgave 1.7

- Innholdet i while-løkken ovenfor er ikke ferdig utfylt. Hvilke(t) av følgende alternativer kan vi erstatte . . . med slik at metoden virker slik den skal?
  - `Person b = it.next();`
  - `Person b = (Person)it.next();`
  - `Person b = liste.next();`
  - `Person b = (Person)liste.next();`
  - `Bil b = (Bil)it.next();`
  - Ingen av alternativene ovenfor

Oppgave 1.7 – vurderinger



- **NB!** En del spørsmål vil være formet slik at de sjekker forståelse. Målet er at de som kopierer blindt fra læreboken eller lysark, skal tabbe seg ut
- De fleste eksemplene kaller iteratoren for **`it`**, men i dette eksemplet heter den **`liste`**
- Det er mange eksempler med **`Bil`** i læreboken, men de har ingenting med dette eksemplet å gjøre
- Derimot demonstrerer eksemplene at man må typekonvertere når man bruker en iterator



Oppgave 1.7 – svar

- Svaret på oppgave 1.7 blir da **alternativ 4:**

  “**`Person b = (Person)liste.next();`**”

Oppgave 1.8

- Hva skrives ut?

```
int i = 11;
int j = i;
int k = 32;

if (k > j * i || k < i) {
   System.out.println("A");
} else {
   if (k < j * i && k > i) {
      System.out.println("B");
   } else {
      System.out.println("C");
   }
}
```

j er 11, j*i er 121

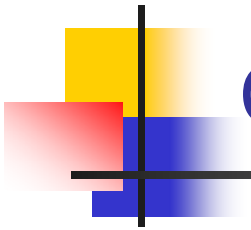
32 > 121 er false

32 < 11 er false

32 < 121 er true

32 > 11 er true

Svar: B



Oppgave 1.9

- Hvordan beregne summen av tallene i **a**?

```
int[] a = new int[77];

int sum = 0;
```

Oppgave 1.9 – svar

```
int i = 0;
while (i < a.length) {
  sum = a[i]; i++;
}
```

Feil sum, får sum = a[76]

```
int i = 0;
while (i < a.length) {
  sum += a[i]; ++j;
}
```

Feil sum, evig løkke, sum = a[0]+a[0]+...

```
int i = 0;
while (i < a.length) {
  sum += a[i]; i++;
}
```

JA – riktig sum

```
int i = 0;
while (i++ < a.length) {
  sum += a[i-1];
}
```

JA – riktig sum

Oppgave 1.9 – svar

```
for (int i = 0; i < a.length; i++) {
  sum += a[i];
}
```

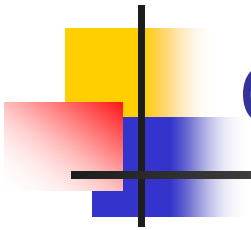
JA – riktig sum

```
for (int i = 1; i <= a.length; i++) {
  sum += a[i-1];
}
```

JA – riktig sum

```
for (int i = 0; i < a.length; ++i) {
  sum = sum + a[i];
```

JA – riktig sum



Oppgave 1.10

- Hvilken verdi får **k**?

```
int i = 11;
int k = i/3;
```

- Alternativer: 3, 3.67, 4, ingen av alt.
- Svar: **k** blir 3 (heltallsdivisjon)

Oppgave 1.11

- Hva blir **k**?
- Alternativer: 12, 24, 6
- Svar: 12 (ikke retur)

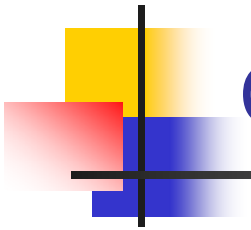
```
void dobleVerdi(int k) {
  k = k * 2;
}
```

```
int k = 12
dobleVerdi(k)
System.out.println(”Verdien til k er ” + k);
```

Oppgave 1.12

- Anta at følgende program utføres:

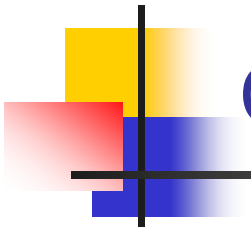
```
class Studentregister {
  public static void main (String[] arg) {
    Student s = new Student("Ole", "Karl Johans gt 1");
    Student p = new Student("Marit","Karl Johans gt 2");
    System.out.println(s.fåNavn()+" og "+p.fåNavn());
  }
}

class Student {
  String navn = "Grete";
  String adresse = "Blindernveien 3";

  Student (String navn, String adresse) {
    this.navn = navn;
    this.adresse = adresse;
  }

  String fåNavn () {return navn;}
}
```



Oppgave 1.12

- Hva blir utskriften på skjermen?
  - Grete og Grete
  - Ole og Ole
  - Marit og Marit
  - navn og navn
  - Ole og Marit
  - s.fåNavn() og s.fåNavn()
  - Marit og Ole
  - Ingen av alternativene over

Oppgave 1.12 – vurderinger



- Det er ingen klassevariable (angitt med static) i denne oppgaven, kun vanlige objektvariable

- Deklarasjonene

```
String navn = "Grete";
String adresse = "Blindernveien 3";
```

  utføres først hver gang et nytt objekt lages, men overskrives så av det som skjer i konstruktøren:

```
Student (String navn, String adresse) {
  this.navn = navn;
  this.adresse = adresse;
}
```

- Metoden **`fåNavn`** er en vanlig objektmetode og utføres “inne i” det objektet som angis i kallet:

  `s.fåNavn()` og **`p.fåNavn()`**

Oppgave 1.12 – svar

- Det riktige svaret på oppgave 1.12 blir da **alternativ 5:**

  “Ole og Marit”