INF1000 : Forelesning 4

Kort repetisjon av doble (nestede) løkker
Mer om 1D-arrayer
Introduksjon til 2D-arrayer
Metoder

Ole Christian Lingjærde
Biomedisinsk forskningsgruppe
Institutt for informatikk
Universitetet i Oslo

Repetisjon: nesting av løkker

```
for (int i=0; i<3; i++) {
   for (int j=0; j<5; j++) {
      System.out.print(i*j);
   }
   System.out.println();
}
```

	j=0	j=1	j=2	j=3	j=4
i=0					
i=1					
i=2					

Repetisjon: arrayer

- Deklarere og opprette array - eksempler:

```
int[] a1 = new int[100];
String[] a2 = new String[100];
```

I begge tilfellene over får arrayen 100 elementer, nummerert 0, 1, ..., 99.

- Bruke array - eksempler:

```
for (int i=0; i<100; i++) {
   a1[i] = i;
}

for (int i=0; i<a2.length; i++) {
   a2[i] = "Posisjon nr " + i;
}
```

Er denne lovlig?

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

Er denne lovlig?

```
int a = new int[5];
```

Er denne lovlig?

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

Er denne lovlig?

```
int[] a = new int[99];
a[99] = (int) 3.14;
```

Er denne lovlig?

```
double[] a = new int[100];
```

Er denne lovlig?

```
String[] s = new String[3];
s = {"juni", "juli", "august"};
```

Er denne lovlig?

```
String[] s = new String[3];
s = new String[]{"juni", "juli", "august"};
```

Automatisk initialisering av arrayer

- Når en array blir opprettet, blir den automatisk initialisert (dvs verdiene er ikke udefinerte når den er opprettet).

```
int[] k = new int[100];          // Nå er alle k[i] == 0
double[] x = new double[100];    // Nå er alle x[i] == 0.0
boolean[] b = new boolean[100];  // Nå er alle b[i] == false
char[] c = new char[100];        // Nå er alle c[i] == '\u0000'
String[] s = new String[100];    // Nå er alle s[i] == null
```

- Merk: String-arrayer initialiseres med den spesielle verdien `null`. Dette er *ikke* en tekststreng og må ikke blandes sammen med en tom tekst: `""`.

- For å kunne bruke verdien `s[i]` til noe fornuftig må du først sørge for å gi `s[i]` en tekststreng-verdi, f.eks. `s[i] = "Per"` eller `s[i] = "".`

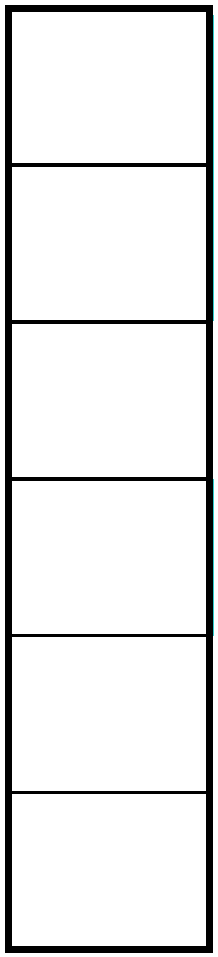
Egendefinert initialisering av en array

- Det er ikke alltid den automatiske initialiseringen av en array gir det vi ønsker. Vi kan da initialisere arrayen med våre egne verdier, slik som i disse eksemplene:

```
int[] primtall = {2, 3, 5, 7, 11, 13};

double[] halve = {0.0, 0.5, 1.0, 1.5, 2.0};

String[] ukedager = {”Mandag”, ”Tirsdag”, ”Onsdag”,
       ”Torsdag”, ”Fredag”, ”Lørdag”, ”Søndag”};
```

Finne minste verdi i en array

```
double minste = a[0];
for (int i=1; i<6; i++) {
     if (a[i] < minste) {
          minste = a[i];
     }
}
```



minste: 

Utvidelse - hvor ligger minste verdi?

```
double minste = a[0];
minPos = 0;

for (int i=1; i<6; i++) {
    if (a[i] < minste) {
        minste = a[i];
        minPos = i;
    }
}
```

Fullstendig program

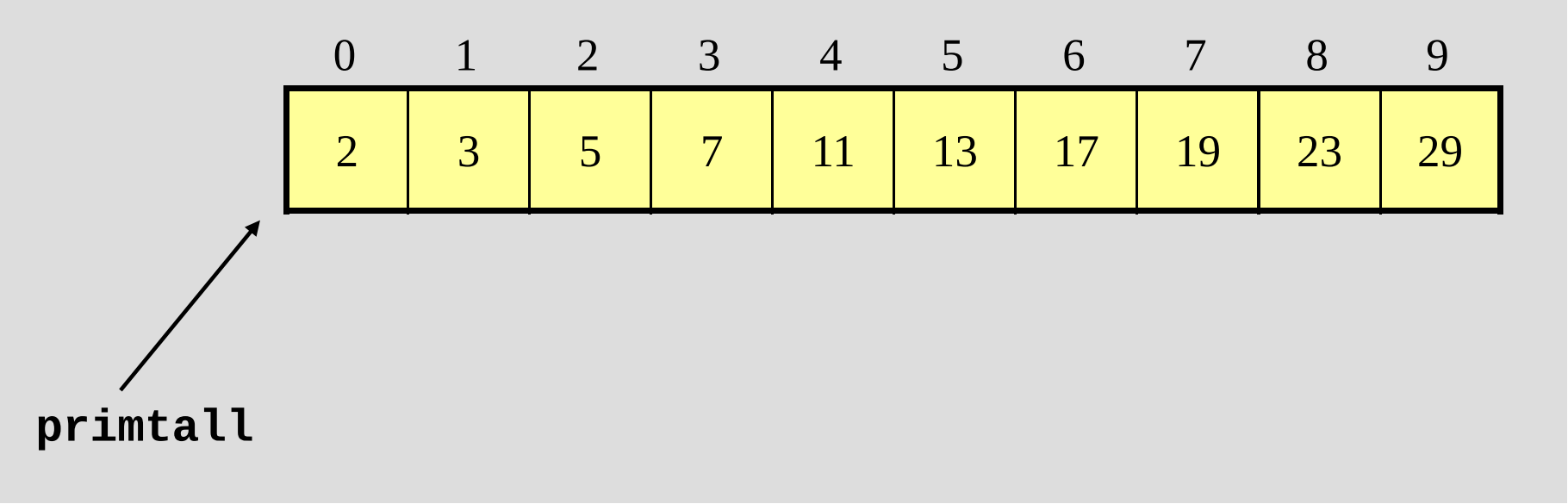
```
import easyIO.*;

class MinsteVerdi {
   public static void main (String[] args) {
      In tastatur = new In();
      double[] a = new double[5];

      for (int i=0; i<a.length; i++) {
         System.out.print("Oppgi en verdi: ");
         a[i] = tastatur.inDouble();
      }

      double minste = a[0];
      int minPos = 0;
      for (int i=1; i<a.length; i++) {
         if (a[i] < minste) {
            minste = a[i];
            minPos = i;
         }
      }

      System.out.println("Minste verdi er " + minste);
      System.out.println("Den ligger på plass " + minPos);
   }
}
```

En array-variabel er en adresse

- Når du har opprettet en array så inneholder arrayvariabelen adressen til stedet i hukommelsen hvor verdiene ligger lagret.
- Eksempel: resultatet etter at vi har utført

```
int[] primtall = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29};
```

kan visualiseres slik:



Oppgave (viktig å forstå!)

Hva blir utskriften fra følgende program?

```
class ToArrayer {
  public static void main (String [] args) {
     int[] x = new int[10];
     int[] y = x;

     for (int i=0; i<10; i++) {
        x[i] = i;
        y[i] = 10-i;
     }

     for (int i=0; i<10; i++) {
        System.out.println(x[i] + "\t" + y[i]);
     }
  }
}
```

Kompilering og kjøring

```
> javac ToArrayer.java
> java ToArrayer
10      10
9       9
8       8
7       7
6       6
5       5
4       4
3       3
2       2
1       1
```

Merk: samme innhold i de to arrayene x og y!!

Hva som skjedde

Etter å ha utført instruksjonen ... **... så er situasjonen denne:**

```
int[] x = new int[10];
```

```
int[] y = x;
```

Forklaring

```
class ToArrayer {
 public static void main (String [] args) {
   int[] x = new int[10];
   int[] y = x;

   for (int i=0; i<10; i++) {
    x[i] = i;
    y[i] = 10-i;
   }

   for (int i=0; i<10; i++) {
    System.out.println(x[i] + "\t" + y[i]);
   }
 }
}
```

Kopiering av arrayer

Vi kan ikke lage en kopi av en array x ved å skrive

```
int[] y = x;
```

siden dette bare medfører at adressen til arrayen kopieres til y.

Skal vi lage en kopi, må vi først opprette en array til (f.eks. y), og så kopiere over verdiene en for en, for eksempel slik:

```
double[] y = new double[x.length];
for (int i=0; i<x.length; i++) {
    y[i] = x[i];
}
```

Tips: det finnes også ferdige verktøy for å kopiere arrayer, bl.a.

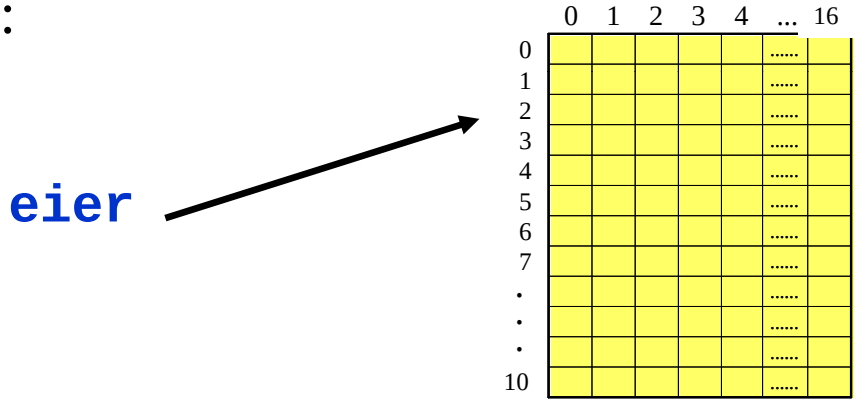
```
int[] y = (int[]) x.clone();
```

Flerdimensjonale arrayer

- Eksempel (fra oblig 2):

```
final int ANT_RADER = 11;
final int ANT_KOLONNER = 17;
String[][] eier = new String[ANT_RADER][ANT_KOLONNER];
```

- Resultat:



- Eksempel på bruk:

```
for (int i=0; i<ANT_RADER; i++) {
  for (int j=0; j<ANT_KOLONNER; j++) {
     <gjør noe med eier[i][j]>
  }
}
```

Ta inn og ut av 2D-array

Vi skal lage et program som illustrerer hvordan man

- deklarerer og oppretter en to-dimensjonal array
- legger inn verdier i arrayen
- henter ut verdier fra arrayen

Programmet skal:

- be om og lese inn 9 desimaltall og legge tallene inn i en double-array med 3 rader og 3 kolonner
- skrive ut double-arrayen som en 3x3-tabell, men slik at verdiene i tabellen har fått fratrukket gjennomsnittet av alle de 9 innleste verdiene.

Det ferdige programmet

```
import easyIO.*;

class ArrayEksempel {
   public static void main (String[] args) {
      In tast = new In();
      Out skjerm = new Out();
      double[][] x = new double[3][3];

      double sum = 0;
      for (int i=0; i<3; i++) {
         for (int j=0; j<3; j++) {
          skjerm.out("x[" + i + "][" + j + "]=");
          x[i][j] = tast.inDouble();
          sum += x[i][j];
         }
      }

      double snitt = sum / 9;
      for (int i=0; i<3; i++) {
         for (int j=0; j<3; j++) {
            skjerm.out(x[i][j] - snitt, 2, 7);
         }
         skjerm.outln();
      }
   }
}
```

Eksempel: finne antall solgte oljefelt i oblig 2

```
int antallFelter = <totalt antall oljefelter>;
int antallSolgte = 0;

for (int i=0; i<11; i++) {
   for (int j=0; j<17; j++) {
      if (eier[i][j] != null) {
         <skriv ut en "x">
         antallSolgte++;
       } else {
         <skriv ut en ".">
       }
   }
}

double prosentSolgte = 100.0 * antallSolgte / antallFelter;
<skriv ut antall solgte felter og prosent solgte felter>
```

Hvis et felt ikke er solgt:

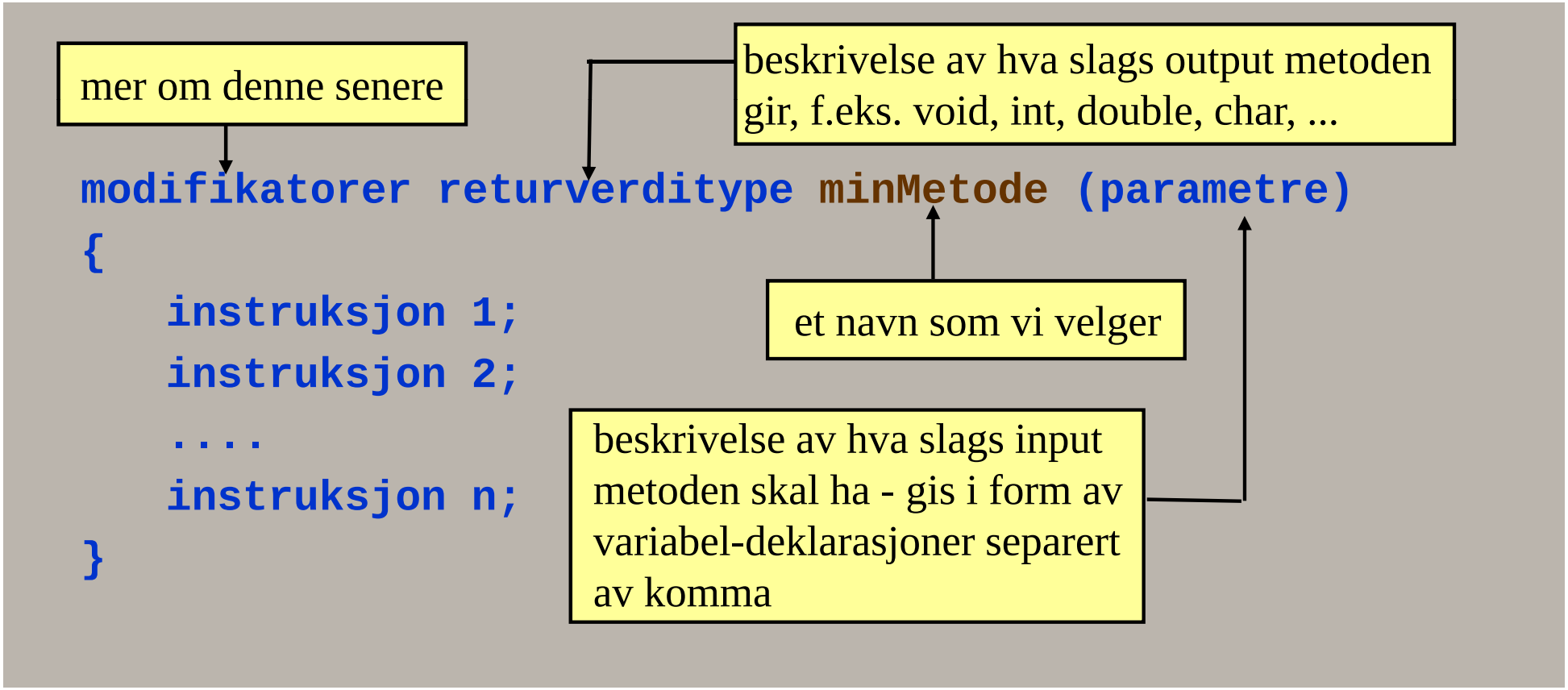
```
eier[i][j] = = null
```

Hvis et felt er solgt:

```
eier[i][j] ! = null
```

Metodedeklarasjoner

- En metode er en navngitt blokk med instruksjoner som vi kan få utført hvor som helst i et program ved å angi metodens navn.
- Vi definerer (deklarerer) metoder etter følgende mønster:



Metodekall

- En metode *kan* kreve input og den *kan* returnere en verdi, men ingen av delene er nødvendig. I enkleste tilfelle er det ingen input og ingen output.
- Når vi benytter en metode sier vi at vi kaller på metoden.
- Kall på metode uten input (=parametre) - eksempel:

```
minMetode();
```

- Kall på metode med input (=parametre) - eksempel:

```
minMetode(34.2, 53, 6);
```

- Kall på metode som returnerer en verdi - eksempel:

```
int alder = minMetode(25.3, 52);
```

Plassering i programmet

Vi plasserer metodene i en egen klasse (eller flere klasser):

Filen MittProgram.java :

```
class MittProgram {
   public static void main(String[] args) {
      Hjelpeklasse hj = new Hjelpeklasse();
      hj.minMetode();
   }
}

class Hjelpeklasse {
   void minMetode() {
     ... innholdet i metoden ...
   }
}
```

Metode uten parametre/returverdi

Følgende metode skriver ut fire linjer på skjermen:

```
void skrivStjerner () {
   String s = "****";
   System.out.println(s);
   System.out.println(s+s);
   System.out.println(s+s+s);
   System.out.println(s+s+s+s);
}
```

Forklaring:

- void forteller at metoden ikke gir noe output.
- skrivStjerner er det navnet vi har valgt å gi metoden

Eksempel på bruk

```
class Stjerner {
   public static void main (String[] args) {
      Hjelpeklasse hj = new Hjelpeklasse();
      hj.skrivStjerner();
   }
}

class Hjelpeklasse {
   void skrivStjerner() {
      String s = "****";
      System.out.println(s);
      System.out.println(s+s);
      System.out.println(s+s+s);
      System.out.println(s+s+s+s);
   }
}
```

Kompilering og kjøring

```
> javac Stjerner.java
> java Stjerner
****

********

**********

****************
```

Metode med returverdi

- Følgende metode leser et positivt desimaltall fra terminal og returnerer det:

```
double lesPositivtTall() {
   In tastatur = new In();
   double x;

   do {
      System.out.println("Gi et positivt tall: ");
      x = tastatur.inDouble();
   } while (x <= 0);

   return x;
}
```

- Forklaring: return x betyr at metoden slutter å eksekvere og at verdien til variabelen x returneres til kallstedet.

Eksempel på bruk

```
import easyIO.*;

class LesPositivtTall {
   public static void main (String[] args) {
      Hjelpeklasse hj = new Hjelpeklasse();
      double tall = hj.lesPositivtTall();
      System.out.println("Tallet var " + tall);
   }
}

class Hjelpeklasse {
   double lesPositivtTall () {
      In tastatur = new In();
      double x;
      do {
         System.out.print("Gi et positivt tall: ");
         x = tastatur.inDouble();
      } while (x <= 0);
      return x;
   }
}
```

Eksempel: metode med input og output

Følgende metode finner summen av elementene i en double-array:

```
double finnSum (double[] x) {
   double sum = 0.0;
   for (int i=0; i<x.length; i++) {
      sum += x[i];
   }
   return sum;
}
```

Forklaring:

- Når metoden kalles vil input-parameteren x bli tilordnet en verdi (= den verdien som benyttes i kallet på metoden).
- Metoden summerer elementene i arrayen, og returnerer så denne summen tilbake til kallstedet.

Eksempel på bruk

```
class Lengde {
   public static void main (String[] args) {
      double[] lengder  = {2.0, 5.1, 7.5, 2.0, 3.8};
      Kalkulator k = new Kalkulator();
      double total = k.finnSum(lengder);
      System.out.println("Samlet lengde: " + total);
   }
}

class Kalkulator {
   double finnSum (double[] x) {
      double sum = 0.0;
      for (int i=0; i<x.length; i++) {
         sum += x[i];
      }
      return sum;
   }
}
```

Parametre og argumenter

```
class MittProgram {
   public static void main (String[] args) {
      Kalkulator k = new Kalkulator();
      double pris = 100.0;
      double beløp = k.trekkFraRabatt(pris);
      System.out.println("Utsalgspris: " + beløp);
   }
}

class Kalkulator {
   double trekkFraRabatt (double x) {
      return x * 0.8;
   }
}
```

Argument

Parameter

Merk: i noen bøker kalles argumenter *aktuelle parametre*, mens parametre kalles *formelle parametre*.

Overføring av verdi til parametere

Anta at følgende eksekveres:

```
double beløp =
 k.trekkFraRabatt(pris);
```

Metoden som kalles:

```
double trekkFraRabatt (double x) {
      return x * 0.8;
}
```

Eksekveringsrekkefølgen (blått angir ting som skjer "bak kulissene"):

```
double beløp = k.trekkFraRabatt(pris);
```

argumentet pris overføres

```
x = pris;
return x * 0.8;
```

metodekallet returnerer med beregnet verdi

```
double beløp = 80.0;
```

Parametre og argumenter

- **Parametre:**

  Deklareres mellom parentesene i toppen (= den første linja) av metode-deklarasjonen. De er "vanlige variable" som bare eksisterer inne i metoden og så lenge denne eksekverer.

- **Argumenter:**

  Verdier som oppgis mellom parentesene når vi kaller på en metode.

- **Antall argumenter:**

  MÅ samsvare med antall parametre i metoden, ellers oppstår feil under kompilering av programmet.

- **Argumentenes datatyper:**

  MÅ samsvare med datatypen til tilsvarende parameter.

  (eller kunne tilordnes til parameteren)