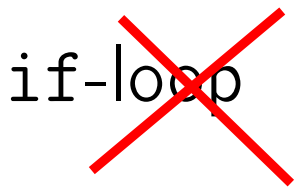
Dagens föreläsning

- Sett i datorsalarna
- Mer om arrayer
- Matriser
- Formatering av utskrifter
- Inläsning med hjälp av `Scanner`-klassen

Hört och sett



Skall vara `if-`*sats*!!

Hört och sett

Uppgiften ”skriv en metod som *returnerar* ...”

```
public void metod(...) {
   ...
   System.out.println("Värdet är: " + ...);
}
```

Skall vara

```
public  typ  metod(...) {
   ...
   return ...;
}
```

Sett vid redovisningar

```
...
d.getValue();
System.out.println(d.getValue());
...
```

Första `d.getValue()` meningslöst — har ingen effekt!

Sett vid redovisningar

Vad händer när nedanstående kod körs?

```
int i = 1;
while (i < 10);
{
    ...
    i = i + 1;
}
```

Skriver nedanstående kod något?

```
int x = 1;
if (x < 0);
{
   System.out.println("x är negativt");
}
```

Sett vid redovisningar

Vad skrivs ut av

```
int i = 1;
i = i++;
System.out.println(i);
```

Vad får `n` för värde i dessa fall?

```
i = 1;
n = i + i++;
```

```
i = 1;
n = i++ + i;
```

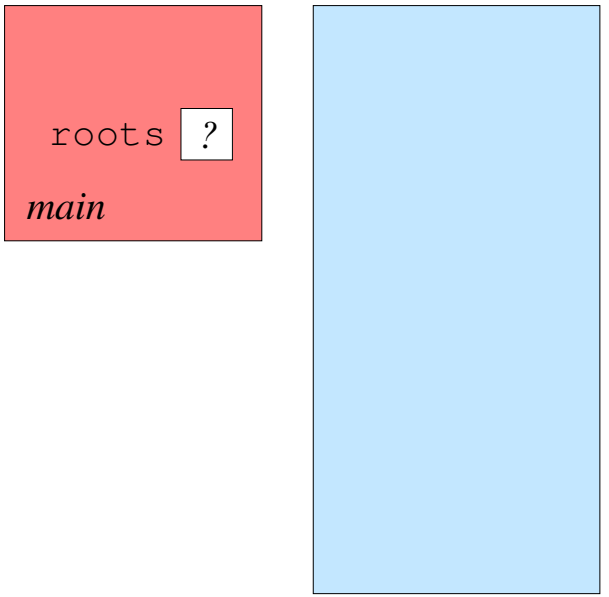
Råd: Använd operatorerna `++` och `--` som självständiga satser

Metoder kan returnera arrayer

```
public static double[] quadEquation(double p, double q) {
    double d = (p*p - 4*q)/4;
    if (d < 0) {
      System.out.println("Complex roots");
      return null;
    } else {
      d = Math.sqrt(d);
      double[] r = new double[2];
      r[0] = -p/2 + d;
      r[1] = -p/2 - d;
      return r;
    }
}
```

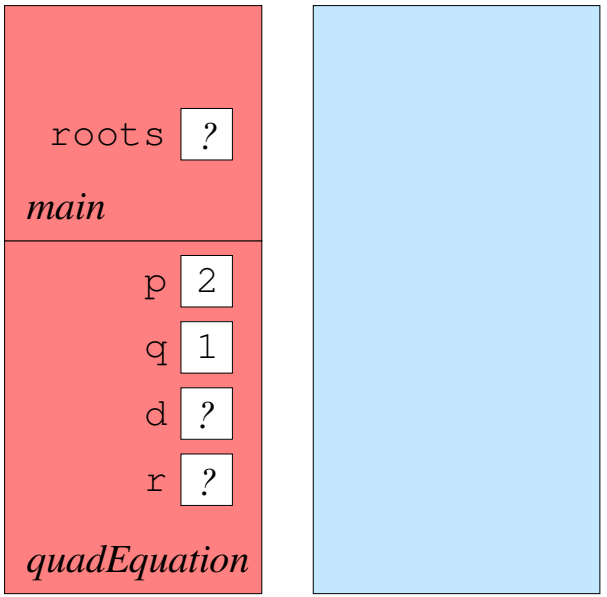
Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots;
→
    roots = quadEquation(2, 1);
    ...
}
```



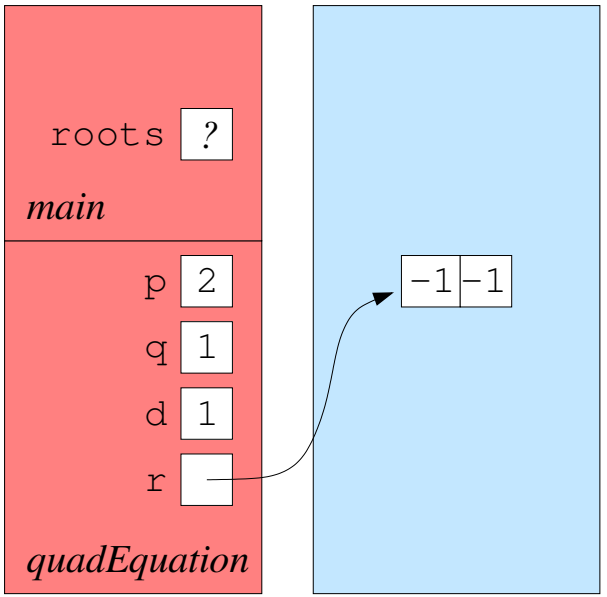
Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots;

    roots = quadEquation(2, 1);
    ...
}


public static double[] quadEquation(double p,
                                    double q) {
→    double d = (p*p - 4*q)/4;
     if (d < 0) {
       System.out.println("Complex roots");
       return null;
     } else {
       d = Math.sqrt(d);
       double[] r = new double[2];
       r[0] = -p/2 + d;
       r[1] = -p/2 - d;
       return r;
     }
}
```



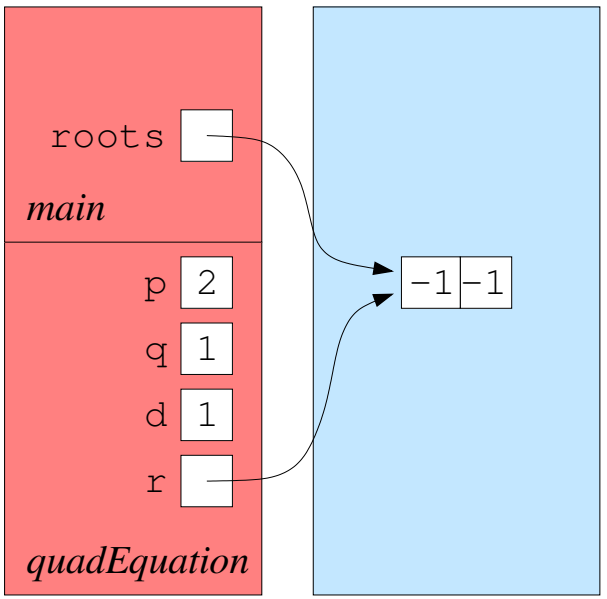
Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots;

    roots = quadEquation(2, 1);
    ...
}


public static double[] quadEquation(double p,
                                    double q) {
      double d = (p*p - 4*q)/4;
      if (d < 0) {
        System.out.println("Complex roots");
        return null;
      } else {
        d = Math.sqrt(d);
        double[] r = new double[2];
        r[0] = -p/2 + d;
        r[1] = -p/2 - d;
→       return result;
      }
}
```



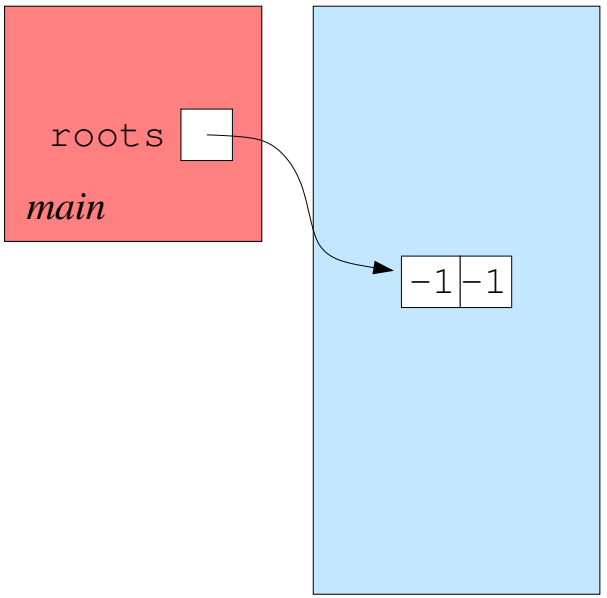
Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots;
    roots = quadEquation(2, 1);
    ...
}


public static double[] quadEquation(double p,
                                    double q) {
      double d = (p*p - 4*q)/4;
      if (d < 0) {
        System.out.println("Complex roots");
        return null;
      } else {
        d = Math.sqrt(d);
        double[] r = new double[2];
        r[0] = -p/2 + d;
        r[1] = -p/2 - d;
        return result;
      }
}
```



Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots;
    roots = quadEquation(2, 1);
    ...
}
```



En anmärkning

De 4 sista raderna i `quadEquation`:

```
double[] r = new double[2];
r[0] = -p/2 + d;
r[1] = -p/2 - d;
return r;
```

kan ersättas med

```
return new double[]{-p/2+d, -p/2-d};
```

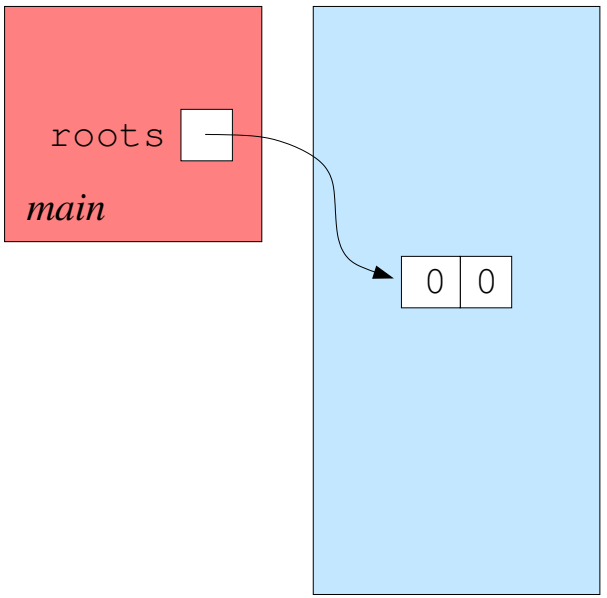
Alternativ design

Låt den anropande koden skapa arrayen skicka den som parameter.

```
public static void quadEquation(double p,
                                double q,
                                double[] r) {
    double d = (p*p - 4*q)/4;
    if (d < 0) {
        ...
    } else {
      d = Math.sqrt(d);
      double[] r = new double[2];
      r[0] = -p/2 + d;
      r[1] = -p/2 - d;
      return r;
    }
}
```

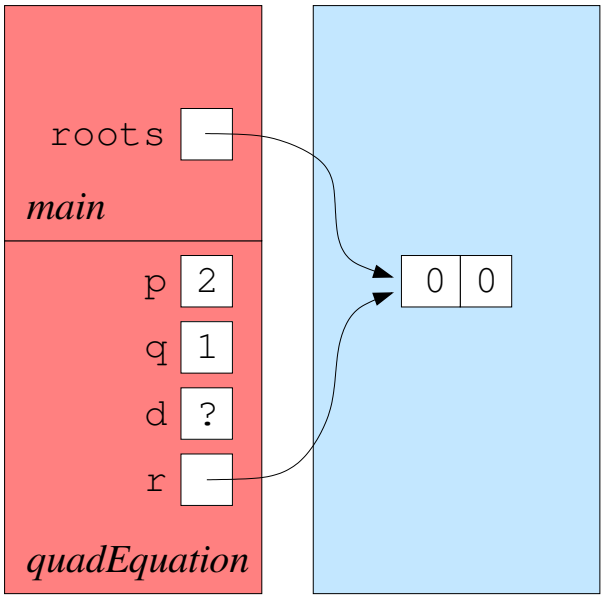
Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots = new double[2];
→
    quadEquation(2, 1, roots);
    ...
}
```



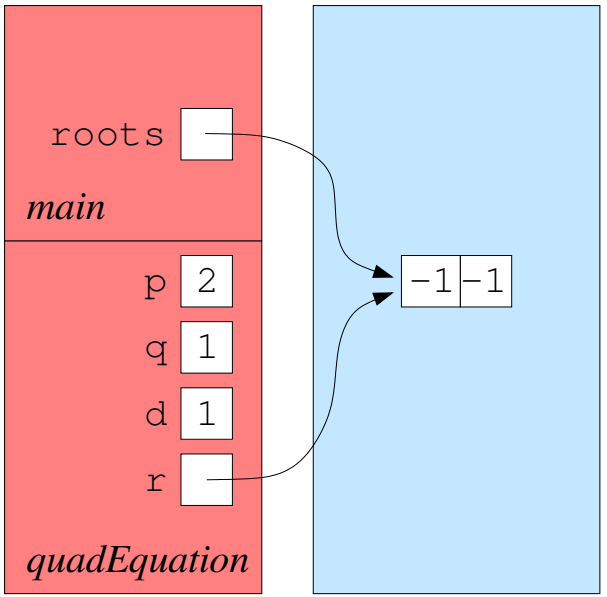
Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots = new double[2];

    quadEquation(2, 1, roots);
    ...
}


public static void quadEquation(double p,
                                 double q,
                                 double[]r) {
→
      double d = (p*p - 4*q)/4;
      if (d < 0) {
        System.out.println("Complex roots");
      } else {
        d = Math.sqrt(d);
        r[0] = -p/2 + d;
        r[1] = -p/2 - d;

      }
}
```



Vad händer vid anrop av quadEquation?

```
public static void main(String[] a) {
    double[] roots = new double[2];

    quadEquation(2, 1, roots);
    ...
}


public static void quadEquation(double p,
                                 double q,
                                 double[]r) {

      double d = (p*p - 4*q)/4;
      if (d < 0) {
        System.out.println("Complex roots");
      } else {
        d = Math.sqrt(d);
        r[0] = -p/2 + d;
        r[1] = -p/2 - d;
→
      }
}
```



Klassen RandomArray i nätlektion 6

```
public class RandomArray {

  private int[] theArray;

  public RandomArray(int size,
                     int limit) {
    theArray = new int[size];
    randomize(limit);
  }

  public RandomArray(int size) {
    this(size, 10);
  }

  public RandomArray() {
    this(25);
  }
```

- Tre konstruktorer.
- Kan anropa varandra.
- Kan anropa metoder.

Klassen RandomArray i nätlektion 6

```
public void randomize(int limit) {
    for (int i = 0; i<theArray.length; i++) {
        theArray[i] = (int)(Math.random()*limit);
    }
}
```

Kan man skriva den med en ”for-each”-sats?

```
public void randomize(int limit) {
    for (int i : theArray) {
        i = (int)(Math.random()*limit);
    }
}
```

? Näää!

Klassen RandomArray i nätlektion 6

```
public class RandomArray {

  private int[] theArray;

  public RandomArray(int size,
                     int limit) {
    theArray = new int[size];
    randomize(limit);
  }

  public RandomArray(int size) {
    this(size, 10);
  }

  public RandomArray() {
    this(25);
  }
```

Vad blir resultatet av nedanstående uttryck?

```
new RandomArray(5,10)

new RandomArray(20)

new RandomArray()
```

Klassen RandomArray i nätlektion 6

```
public String toString() {
   String ret = "";
   for (int i= 0; i< theArray.length; i++) {
      ret = ret + theArray[i];
      if (i < theArray.length-1) {
         ret = ret + ", ";
      }
   }
   return "<" + ret + ">";
}
```

Kan `toString` skrivas med en ”for-each”-loop?

Nej, det går inte att få kommatecken enbart *mellan* elementen.

(Däremot kan man skriva den så här:
`return java.util.Arrays.toString(theArray);`)

Övning: En metod som vänder på arrayen

```
public void reverse1() {
    int n = theArray.length - 1;
    for (int i = 0; i <= n; i++) {
        int x = theArray[i];
        theArray[i] = theArray[n-i];
        theArray[n-i] = x;
    }
}
```

(Bekvämt med ett lokalt kortare namn)

Kommentar?

Demo reverse

Fungerar ej! Byter ju tillbaka.
Lösning: Stoppa loopen när hälften när första halvan behandlats!.

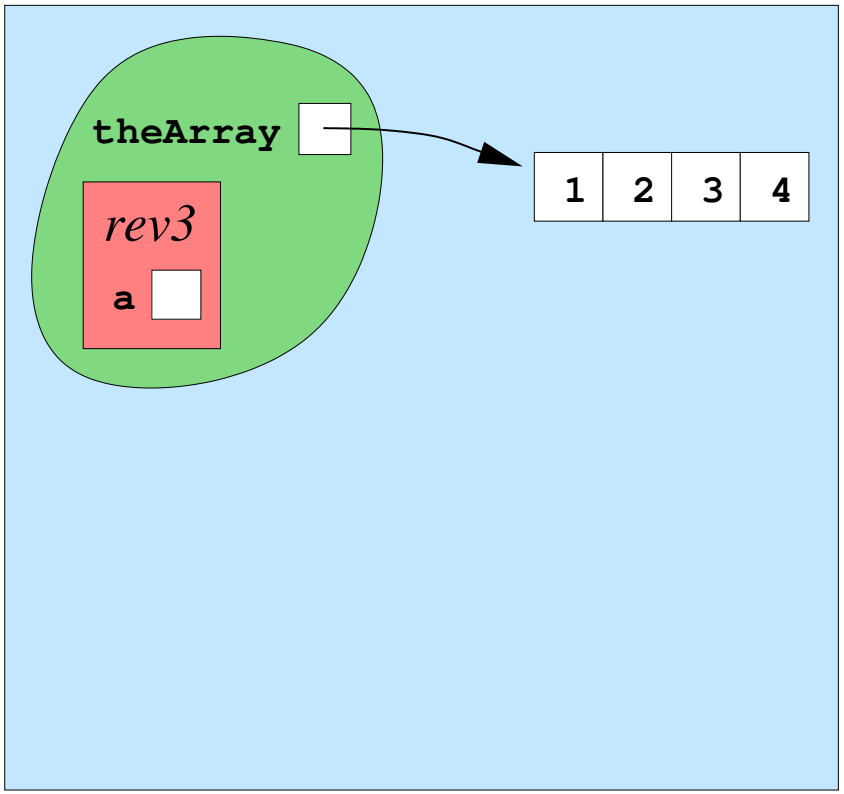
Alternativ reverse

```
public void reverse3() {
    int n = theArray.length;
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
        a[i] = theArray[n-1-i];
    }
    theArray = a;
}
```

Vad händer?

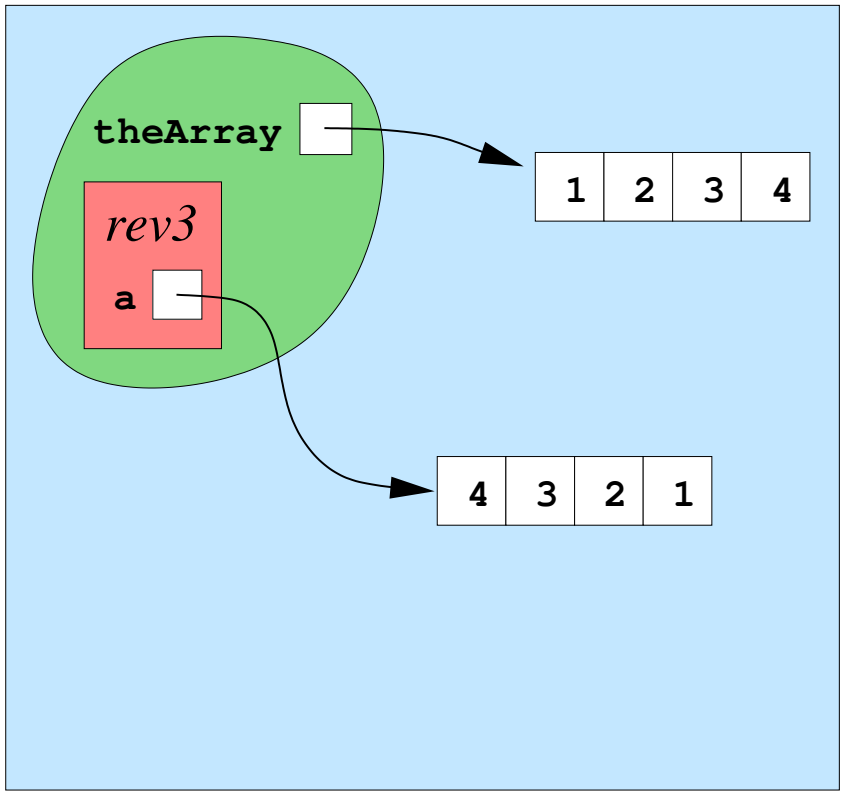
Så här ser det ut när vi just anropat `reverse3`

```
public void reverse3() {
    Här
    int n = theArray.length;
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
       a[i] = theArray[n-1-i];
    }

    theArray = a;

}
```



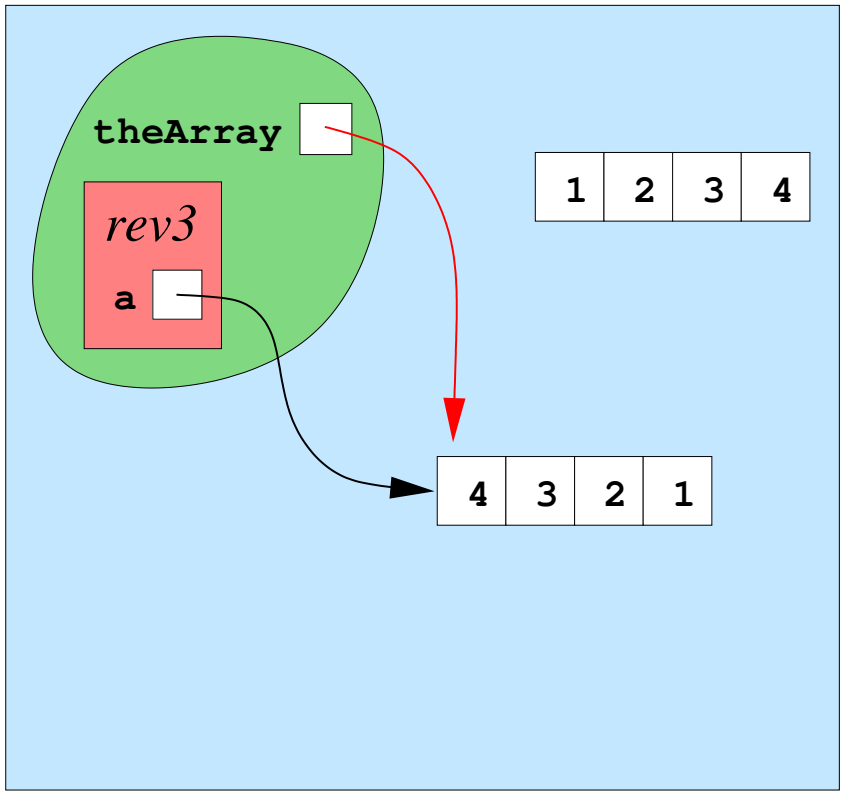
Så här ser det ut när vi är

```
public void reverse3() {

    int n = theArray.length;
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
       a[i] = theArray[n-1-i];
    }
    här
    theArray = a;

}
```



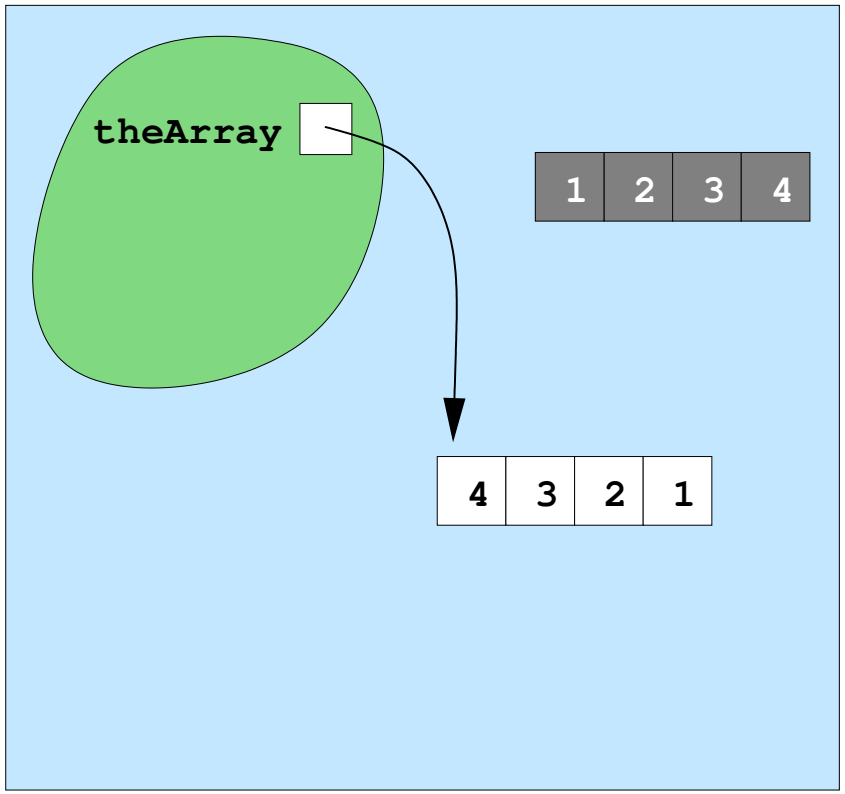
och när vi är

```
public void reverse3() {

    int n = theArray.length;
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
       a[i] = theArray[n-1-i];
    }

    theArray = a;
    här
}
```



och när reverse3 är klar.

```
public void reverse3() {

    int n = theArray.length;
    int[] a = new int[n];
    for (int i = 0; i < n; i++) {
       a[i] = theArray[n-1-i];
    }

    theArray = a;

}
```



Annan design på `reverse`

Låt metoden skapa och returnera ett nytt objekt:

```
public RandomArray reverse4() {
    int n = theArray.length;
    RandomArray result = new RandomArray(n);
    for (int i = 0; i < n; i++) {
       result.theArray[i] = this.theArray[n-i-1];
    }
    return result;
}
```

Skapa ett `RandomArray`-objekt med en given array

Antag att vi vill kunna skapa ett objekt med en given array, t ex:

```
int a[] = {1, 2, 3, 4, 5, 6};
RandomArray ra = new RandomArray(a);
```

Behövs en till konstruktor:

```
public RandomArray(int[] a) {
   theArray = a;
}
```

Hur fungerar den?

Är den bra?

Konstruktor med array-parameter

Skapa ett eget array-objekt och kopiera värden från parametern till den:

```
public RandomArray(int[] a) {
    theArray = new int[a.length];
    for (int i = 0; i < a.length; i++) {
      theArray[i] = a[i];
    }
}
```

Nu har klassen själv kontroll över vad som händer i arrayen.

Det är dock inte säkert att man alltid vill ha det så!

Några frågor

- Antag

  ```
  RandomArray ra = new RandomArray();
  ```

  - Fungerar det att skriva `int x = ra[3]`?
  - Fungerar det att skriva `int x = ra.theArray[3]`?

- Vad är skapas av nedanstående uttryck?

  ```
  new RandomArray(3)
  ```

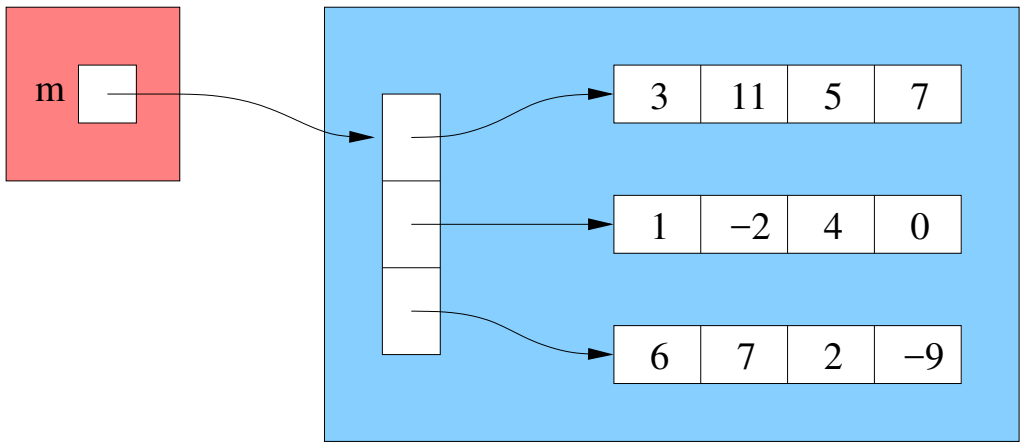
  respektive

  ```
  new RandomArray[3]
  ```

- Är det någon mening att skriva en klass med bara en instansvariabel?

Tvådimensionella arrayer — matriser

En matris är en array
där elementen är arrayer:



Kan skapas med

```
int[][] m = { {3, 11, 5, 7},
              {1, -2, 4, 0},
              {6, 7, 2, -9} };
```

medan

```
int[][] m = new int[3][4];
```

skapar en 3x4 matris med nollor.

Exempel: Statistik på tärningspar

```
public class PairOfDice {
   private Dice[] dice = new Dice[2];

   public PairOfDice(int n1, int n2) {
      dice[0] = new Dice(n1);
      dice[1] = new Dice(n2);
   }

   public void roll() {
      for(Dice d : dice) {
         d.roll();
      }
   }

   public int getValue(int i) {
      return dice[i%2].getValue();
   }
}
```

Implementation med en array som skapas i deklarationen

Tärningarna måste skapas i konstruktorn.

Jämnt värde ger tärning 0, udda värde tärning 1

Statistikklass PoDStatistics

```
public static void main(String[] args){
   PairOfDice pod = new PairOfDice(4, 8);
   int[][] freq = new int[4][8];
   for (int i = 1; i<=10000; i++) {
      pod.roll();
      freq[pod.getValue(0)-1][pod.getValue(1)-1]++;
   }
   for (int i= 0; i<4; i++) {
      for (int j=0; j<8; j++) {
         System.out.print(freq[i][j]/10000 + "  ");
      }
      System.out.println();
   }
   System.out.println();
}
```

Ser ni buggen?

Demo PoDStatistics

Typiska fel

Glömma skapa arrayen:

```
int[] a;
a[i] = 4;
```

NullPointerException

Felaktigt index:

```
int[] a = new int[5];
a[5] = 4;
```

ArrayIndexOutOfBoundsException

Glömma att skapa objekten
i en ”objektarray”:

```
Turtle[] t = new Turle[5];
t[0].move(40);
```

NullPointerException

Observera att felmeddelandet säger *var* felet har inträffat!

Javadoc –ett sätt att producera dokumentationssidor

```
/**
 * Represents a pair of dice
 * @version 2019-02-05
 * @author Tom Smedsaas
 */
public class PairOfDice {
   private Dice d1;
   private Dice d2;

   /**
    * Creates of pair of dice with a specified number of sides
    * @param n1 number of sides of the first dice
    * @param n2 number of sides of the second dice
    */
   public PairOfDice(int n1, int n2) {...}

   /**
    * Roll the dice
    * @return An array with the values
    */
   public int[] roll() {...}
```

Se vidare minilektionen!

Kommunikation med användaren

- Utskrifter i terminalfönster (”konsolen”).
  Gjort med `System.out.print` och `System.out.println`.

  Ska visa på bättre möjligheter att redigera utskrifterna (t ex ange antal decimaler) med `format`-metoden.

- Tolka (”läsa in”) information som användaren skriver i terminalfönstret.
  Ska visa på användning av `Scanner`-klassen.

- *Dialog-rutor* är ett alternativ till att använda konsolen.

- *Grafiska användargränssnitt* (”GUI”).

Utskrift med `println`

Exempel: Koden

```
int x = 10;
double sinx = Math.sin(x);
double expx = Math.exp(x);
System.out.println("x = " + x +
                   ", sin(x) = " + sinx +
                   ", exp(x) = " + expx);
```

resulterar i utskriften

```
x = 10, sin(x) = -0.5440211108893698, exp(x) = 22026.465794806718
```

Utskrift med format

```
int x = 10;
double sinx = Math.sin(x);
double expx = Math.exp(x);
System.out.format("x = %d, sin(x) = %7.4f, cos(x) = %10.4e\n",
                  x, sinx, expx);
```

resulterar i utskriften

```
x = 10, sin(x) = -0.5440, cos(x) = 2.2026e+04
```

i stället för

```
x = 10, sin(x) = -0.5440211108893698, exp(x) = 22026.465794806718
```

format-metoden

`format(`s, p_1, p_2, ..., p_n`)`

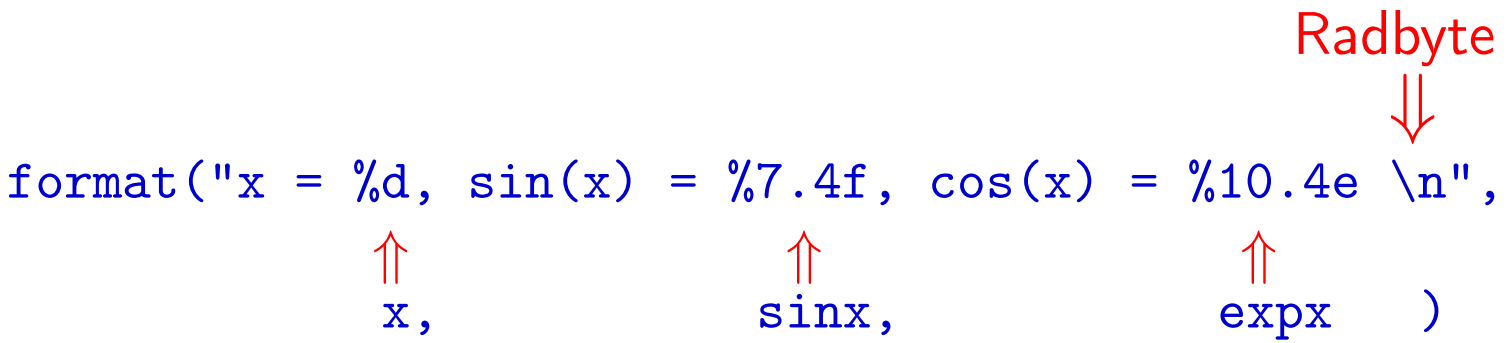
skapar ett `String`-objekt utifrån strängen s.

Strängen s ska innehålla *formatkoder* en för var och en de övriga parametrarna. Varje formatkod byts mot värdet av motsvarande parameter.

Exempel på formatkoder:

`%d`	plats för ett heltal
`%10d`	heltal i ett fält om 10 positioner, högerjusterat
`%-10d`	heltal i ett fält om 10 positioner, vänsterjusterat
`%8.2f`	flyttal i ett fält om 8 positioner, 2 decimaler
`%12.3e`	flyttal i ett fält om 12 positioner, 3 decimaler på exponentform
`%12s`	flyttal i ett fält om 12 positioner, högerjusterat
`%-12s`	flyttal i ett fält om 12 positioner, vänsterrjusterat
`%c`	teckenvärde (`char`)

format-metoden



Metoden `format` finns, förutom i `System.out`, som *klassmetod* i `String`.

Exempel:

```
String name = "Kalle Kula";
int age = 36;
String str = String.format("%s är %d år", name, age);
```

kommer att ge `str` värdet "*Kalle Kula är 36 år*"

Klassen Scanner

Metoden `format` är (ett) sätt att omvandla tal mm till text.

För att gå åt andra hållet dvs omvandla text (”strömmar av tecken”) till tal mm. kan `Scanner`-klassen användas.

Exempel:

```
Scanner scan = new java.util.Scanner(System.in);
System.out.print("Antal sidor: ");
int sides = scan.nextInt();
System.out.print("Antal slag : ");
int rolls = scan.nextInt();
...
```

Demo ScannerDemo1

Normalt *importeras* klassen med satsen `import java.util.Scanner`.

Då behöver man inte ange `java.util` när man refererar klassen.

Metoder i `Scanner`

`int nextInt()`	läs och returnera nästa heltal
`double nextDouble()`	läs och returnera nästa flyttal

Dessa metoder läser förbi eventuella blanktecken och radslut.

Om det inte går att tolka nästa ”grej” som den typ man efterfrågar avbryts programmet.

För att hantera detta kan man testa med metoderna:

`boolean hasNextInt()`	`true` om OK att anropa `nextInt`
`boolean hasNextDouble()`	`true` om OK att anropa `nextDouble`

Demo ScannerDemo2

Metoder i `Scanner`

För att läsa annat än tal:

`String next()`	läs och returnera nästa tok (”grej”)
`String nextLine()`	läs och returnera nästa rad
`boolean hasNext()`	`true` om det finns något icke-blankt att läsa
`boolean hasNextLine()`	`true` om det finns en rad att läsa

Viktigt att veta att

- `nextLine()` läser *resten av raden*.
- `next`, `nextInt` och `nextDouble` byter *inte* rad efter fullbordad läsning. Ställer lätt till trassel vid blandning av rad- och talläsning.

Exempel

Nedanstående kod är tänkt att först läsa ett tal och sedan en hel rad.

```
public class ScannerDemo3 {
   public static void main(String[] s) {
      Scanner scan = new Scanner(System.in);
      System.out.print("Tal : ");
      int n = scan.nextInt();
      System.out.print("Text: ");
      String txt = scan.nextLine();
      System.out.println("Tal : " + n);
      System.out.println("Text: " + txt);
   }
}
```

Demo ScannerDemo3

Måste byta rad med ett `nextLine` innan nästa rad kan läsas.

Exempel: Läs rader och summera talen på varje rad för sig

Så här

```
Scanner scan = new Scanner(System.in);
while (scan.hasNextLine()) {
    int sum = 0;
    while (flera tal på raden) {
        sum += scan.nextInt();
    }
    System.out.println(sum);
}
```

?

Det finns inget enkelt sätt att se om det finns flera tal på raden.

Bättre att läsa rad för rad till en `String` och koppla ett nytt *Scanner*-objekt till detta.

Exempel: Läs rader och summera talen på varje rad för sig

```
public class ScannerDemo4 {

public static void main(String[] args) {
    Scanner scan = new Scanner(System.in);    En scanner för tangentbordet
    while (scan.hasNextLine()) {              Så länge det finns rader:
        String line = scan.nextLine();            Läs en rad
        Scanner lineScan = new Scanner(line);     Koppla scanner till raden
        int sum = 0;
        while (lineScan.hasNextInt()) {           Så länge fler tal på raden:
            sum += lineScan.nextInt();                Läs tal och summera
        }
        lineScan.close();                         Stäng rad-scannern.
        System.out.println(sum);
    }
    scan.close();                             Stäng tangentbords-scannern
}
```