University of Southern Denmark • Syddansk Universitet

DM502

Forelæsning 4

Indhold

- Flere kontrolstrukturer
  - `for`-løkke
  - `switch-case`
- Metoder
- Arrays og sortering af arrays
- String-funktioner

for-løkke

- Ofte har man brug for at udføre det samme kode, for en sekvens af tal/elementer
- Dette kan selvfølgelig gøres med en while-løkke
  - ```
    int i = 1;
    while (i <= 10) {
        System.out.println(i);
        ++i;
    }
    ```
- Alternativt med en `for`-løkke
  - ```
    int i;
    for (i=1; i<=10; ++i) {
        System.out.println(i);
    }
    ```
- Eller lidt mere kompakt
  - ```
    for (int i=1; i<=10; ++i) {
        System.out.println(i);
    }
    ```


    ```

for-løkke

- Vi kan også gennemløbe elementerne i (f.eks.) en ArrayList
  - ```
    ArrayList<Integer> list = new ArrayList<Integer>();
    Collections.addAll(list, 1,5,7,9,5);

    for (int x=0; x<list.size(); ++x) {
        System.out.println(list.get(x));
    }
    ```
- Eller mere kompakt
  - ```
    for (Integer i : list) {
        System.out.println(i);
    }
    ```

switch-case

```
if (i == 0) {
    ...;
} else if (i == 1) {
    ...;
} else if (i == 2) {
    ...;
.
.
.
.
.
} else {
    ...;
}
```

switch-case

```
switch (i) {
case 0:
    ...;
    break;
case 1:
    ...;
    break;
case 2:
.
.
.
.
.
default:
    ...;
    break;
}
```

switch-case

```
String navn;
switch (dag) {
case 1:
    navn = “Mandag”;
    break;
case 2:
    navn = “Mandag”;
    break;
case 3:
    navn = “Mandag”;
    break;
.
.
.
case 7:
    navn = “Mandag”;
    break;
default:
    navn = “Ikke en gyldig dag”;
    break;
```

switch-case

```
int antal;
switch (maaned) {
case 1:
case 3:
case 5:
case 7:
case 8:
case 10:
case 12:
    antal = 31;
    break;
case 4:
case 6:
case 9:
case 11:
    antal = 30;
    break;
case 2:
    if (skudaar) {
        antal = 29;
    } else {
        antal = 28;
    }
}
```

Metoder

- Også kaldet funktioner
- De muliggør genbrug af kode
  - Skal kun skrive koden én gang
  - Men kan bruges mange steder
- En metode er meget lig en matematisk funktion
  - Man propper noget ind i den
  - og får noget ud i den anden ende
  - Fx $f(x) = x^3$
    - Man kalder x et argument til funktionen f

Metoder

- Vi har allerede set metoder brugt flere gange
  - `median = list.get( index - 1 );`
    - `get` er en metode på ArrayLister
    - Den tager en int som argument
  - `tal = tastatur.nextInt();`
    - `nextInt` er en metode på ArrayLister
    - Den tager ingen argumenter
- Ligesom matematiske funktioner
  - Man propper noget ind (evt. ingenting)
    - Fx en int - et index
  - Man får noget ud i den anden ende
    - Fx et element i listen - elementet i listen på det pågældende index

Metoder i Java

- Erklæring
  - Hvilke argumenter
  - Hvad returneres (sendes tilbage)
    - Fx en int i `nextInt()` tilfældet
- Implementation
  - Hvordan skrives selve koden
  - Brug af argumenter
  - Returnering
- Brug af en metode.
  - Hvordan anvendes en metode
  - Det har vi allerede set flere gange

Erklæring af metoder

- `public static RETURTYPE metodenavn( ARGUMENTLISTE ) {`
  `... KODE ...`
  `}`
- `public static`
  - Ignorer vi foreløbigt
- `RETURTYPE`
  - Typen af det der returneres fra funktionen
  - Fx `int` i `nextInt()` tilfældet
  - Kan være `void` hvis der ikke returneres noget
    - Bruges ofte ved sideeffekter
- `metodenavn`
  - Navnet på metoden

Erklæring af metoder

- `ARGUMENTLISTE`
  - En liste, adskilt med komma, af argumenter til metoden
  - Evt. er listen tom (ingen argumenter)
    - Som fx for `nextInt()`
  - Hvert argument i listen angives som
    - `TYPE navn`
    - Fx er argumentet til metoden `get` fra `ArrayList` `int index`
- Bemærk:
  - Flere ting ind (argumenter)
  - Kun én ting ud (retur-værdi)

Eksempler på erklæringer

- `nextInt()` på `Scanner`
  - `public int nextInt()`
  - Kan også ses i API'et
- `println()` fra `System.out`
  - `public void println( String x )`
  - Metoden tager et argument, en String
  - Metoden returner ingenting
- Erklæringen på en metode der opløfter x i y'te
  - Metoden skal tage x og y som argumenter
  - Returnere x^y
  - `public static int power( int x, int y )`

Implementation af metoder

- Argumenter kan bruges ligesom alle andre variable (de har den værdi brugeren af metoden har angivet)
- Returnering sker med `return`
- Variable erklæret i en metode kan ikke "ses" i en anden metode (scope)
  - Man kan altså godt bruge samme variabelnavn i flere metoder

Eksempel

```
public static int power( int x, int y ) {
      int result = 1;
      int i;

      for( i = 1; i <= y; ++i ) {
            result = result * x;
      }

      return result;
}
```

Eksempel

```
public static int power( int x, int y ) {
      int result = 1;
      int i;

      for( i = 1; i <= y; ++i ) {
            result = result * x;
      }

      return result;
}
```

Ækvivalent med:

```
public static int power( int x, int y ) {
      int result = 1;
      int i;

      i = 1;
      while( i <= y ) {
            result = result * x;
            ++i;
      }

      return result;
}
```

Komplet eksempel

```
public class PowerExample {
    public static void main( String[] args ) {
        int result;
        int a, b;

        a = 2;
        b = 4;
        result = power( a, b );
        System.out.println( result );

        result = power( b, 0 );
        System.out.println( result );

        System.out.println( power( 1, 2 ) );
    }

    public static int power( int x, int y ) {
        int result = 1;
        int i;

        for( i = 1; i <= y; ++i ) {
            result = result * x;
        }

        return result;
    }
}
```

Arrays

- En statisk liste i Java
  - Dvs. ingen indsættelser/tilføjelser (længden er fast)
  - Må dog godt ændre i et array
- Erklæring
  - Hvordan laves et array?
- Initialisering
  - Hvordan kommes ting ind i arrayet?
- Brug
  - Hvordan læses og ændres i arrayet?

Arrays

- `TYPE[] navn;`
  - Hvilken type elementer skal der ligge i arrayet
  - Fx `int[] speedDial;`
- `navn = new TYPE[ LÆNGDE ];`
  - Hvor mange pladser skal der være i arrayet
  - Fx `speedDial = new int[20];`
- Eller det hele på en linie
  - `int[] speedDial = new int[20];`
- Bemærk: Ikke noget antal mellem første sæt klammer
- Initialisering
  - int[] speedDial = {12345678, 13579246}
    - Længden givet ved længden af initialiseringslisten

Arrays

- `int[] speedDial = new int[20];`
- Husk at lister i Java er 0-indekseret!
  - `speedDial` har elementer på plads 0 - 19
- Læs det 5. element i listen
  - `speedDial[4];`
- Skriv en værdi på den 5. plads i arrayet
  - `speedDial[4] = 21323454;`
- Bemærk at nogle pladser godt kan være uinitialiseret
- Længden af arrayet kan læses
  - `speedDial.length`

Array - eksempel

```
public class ArrayExample {
    public static void main( String[] args ) {
        int[] speedDial = new int[5];
        int i;

        speedDial[0] = 12345678;
        speedDial[1] = 90123456;

        System.out.println( "Plads 0: " + speedDial[0] );
        System.out.println( "Plads 1: " + speedDial[1] );

        System.out.println( "Længde: " + speedDial.length );

        for( i = 0; i < speedDial.length; ++i ) {
            speedDial[i] = i;
        }

        for( i = 0; i < speedDial.length; ++i ) {
            System.out.println( speedDial[i] );
        }
    }
}
```

Array - sortering

```
import java.util.Arrays;

public class ArraySortExample {
    public static void main( String[] args ) {

        int[] a = {1,78,4,67,4,2,2,5,7};

        System.out.println(Arrays.toString(a));
        Arrays.sort(a);
        System.out.println(Arrays.toString(a));

    }
}
```

String

- "Strenge" på dansk
- En type der kan indeholde tekst
- Vi har allerede set `String` brugt flere gange
- Erklæring og initialisering
  - ```
    String s;
    s = "Hej med dig";
    ```
  - ```
    String s = "Hej med dig";
    ```
- Man kan ikke ændre på indholdet af en `String` (fx ændre et bogstav)
  - En `String` er i virkeligheden et `char` array
    - Men vi kan ikke ændre det der står på den enkelte plads

String - vigtige funktioner

- Eksempel: `String s = “Hej med dig”;`
- `char charAt( int index )`
  - `s.charAt( 2 );`
  - Returnerer `“j”`
- `boolean equals( String str )`
  - `s.equals( “HEj med dig” );`
  - Returnerer `false`
- `boolean equalsIgnoreCase( String str )`
  - `s.equalsIgnoreCase( “HEj med dig” );`
  - Returnerer `true`
- `int length()`
  - `s.length();`
  - Returnerer `11`

String - vigtigt

- Man kan IKKE sammenligne på følgende måde:
  - ```
    String s = “Hej med dig”;
    String t = “Hej med dig”;
    if( s == t ) { ... }
    ```
    ```

String - eksempel

```
public class StringExample {
    public static void main( String[] args ) {
        String s = "Hej med dig";

        System.out.println( s.charAt( 2 ) );
        System.out.println( s.equals( "HEj med dig" ) );
        System.out.println( s.equalsIgnoreCase( "HEj med dig" ) );
        System.out.println( s.length() );
```

String - eksempel

```
public class StringExample {
    public static void main( String[] args ) {
        String s = "Hej med dig";

        System.out.println( s.charAt( 2 ) );
        System.out.println( s.equals( "HEj med dig" ) );
        System.out.println( s.equalsIgnoreCase( "HEj med dig" ) );
        System.out.println( s.length() );
        System.out.println( substring( s, 0, 2 ) );
    }

    public static String substring( String str, int i, int j ) {
        String result = "";

        while( i <= j ) {
            result = result + str.charAt( i );
            i++;
        }

        return result;
    }
}
```

main-metoden

- Helt almindelig metode
  - Pånær det er her programmer "starter"
- `public static void main( String[] args ) {...}`
  - Returnerer altså ingenting
  - Tager et String-array som argument
    - Det er argumenter som er givet på kommandolinjen

main-metoden - eksempel

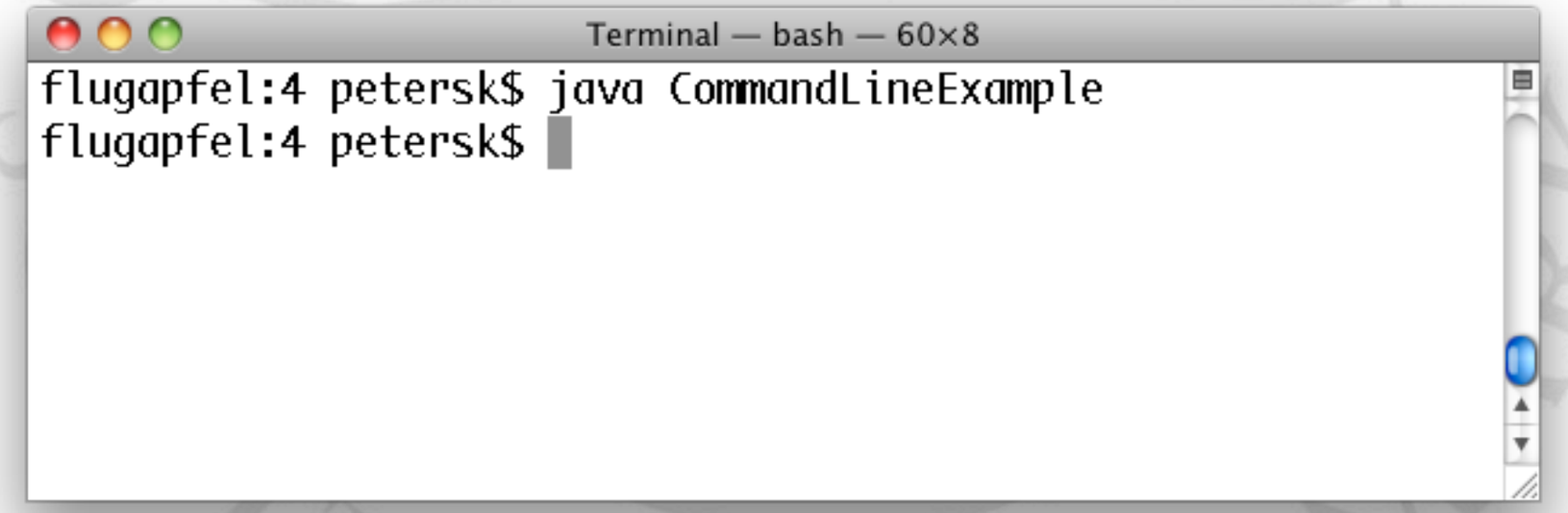
```
public class CommandLineExample {
    public static void main( String[] args ) {
        int i;

        for( i = 0; i < args.length; i++ ) {
            System.out.println( args[i] );
        }
    }
}
```

main-metoden - eksempel

```
public class CommandLineExample {
    public static void main( String[] args ) {
        int i;

        for( i = 0; i < args.length; i++ ) {
            System.out.println( args[i] );
        }
    }
}
```



main-metoden - eksempel

```
public class CommandLineExample {
    public static void main( String[] args ) {
        int i;

        for( i = 0; i < args.length; i++ ) {
            System.out.println( args[i] );
        }
    }
}
```