Kursus 02199: Programmering
afsnit 3.6-3.9

Anne Haxthausen

IMM, DTU

1. Løkke-sætninger (**while**, **do**, **for**) (afsnit 3.6 – 3.8)
2. Oversigt over Java sætninger (lært indtil nu)
3. Lidt om programudviklingsfaser (afsnit 3.9)

Kontrol af programudførelsen

- Med mindre andet angives, udføres et program *lineært*:
  - Java udfører sætningerne i `main` metoden en efter en
- Udførelsen kan styres v.h.a. to specielle sætningstyper:
  - betingede sætninger (valg mellem en eller flere sætninger): **if-else** og **switch**.
  - løkker (gentagelse af sætninger): **while**, **do** og **for**.
- styringen foregår v.h.a. *logiske (Boolske) udtryk*:
  - ved betingelser angiver udtrykket hvilken gren, der skal udføres.
  - ved løkker angives *slutbetingelsen*.

Løkker

Løkke-sætninger bruges til at gentage sætninger.

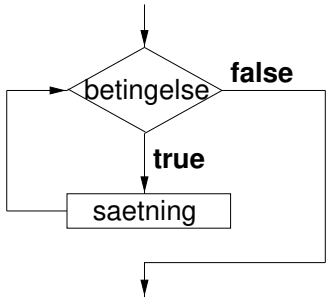
løkke	god til gentagelser på formen
while	så længe ... gør ...
do	gør ... så længe ...
for	gør ... et bestemt antal gange

While-løkker: generelt

Generelt format:

while (*betingelse*)
  sætning

Virkning:



While-løkker: eksempel

```
final int LIMIT = 7;
int count = 2;

while (count <= LIMIT)
{
  System.out.println(count);
  count = count + 1;
}
```

giver følgende output:

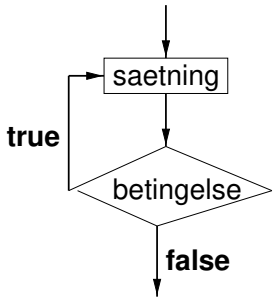
```
2
3
4
5
6
7
```


Do-løkker: generelt

Generelt format:

do *sætning*
while (*betingelse*) ;

Virkning:



Do-løkker: eksempel

do-løkker kan f.eks. bruges til at gentage indlæsning, indtil brugeren har indtastet et korrekt input.

Eksempel:

```
//indlaesning af et 'y','Y', 'n' eller 'N'

  char c;

  do
  {
     System.out.print("Enter (y/n)? ");
     c = Character.toUpperCase(Keyboard.readChar());
  }
  while (! (c=='Y' || c=='N'));
```

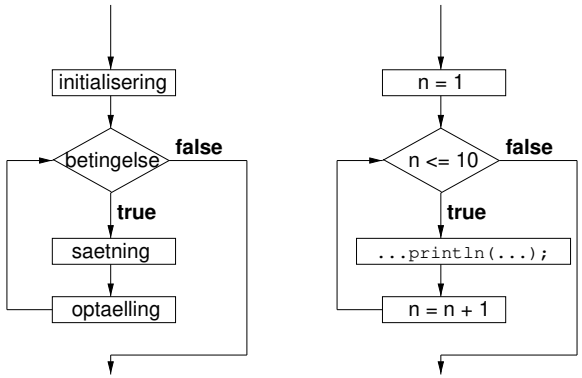

For-løkke: eksempel

```
for (int n=1; n<=10; n = n + 1)
  System.out.println(n + " squared is " + (n * n));
```

giver følgende output:

```
1 squared is 1
2 squared is 4
3 squared is 9
4 squared is 16
5 squared is 25
6 squared is 36
7 squared is 49
8 squared is 64
9 squared is 81
10 squared is 100
```

For-løkker: generelt + eksempel



For-løkker: generelt

Generelt format:

for (*initialisering* **;** *betingelse* **;** *optælling*)
sætning

En *initialisering* er normalt en initialiseret variabelerklæring (f.eks. `int n=1`).
En *betingelse* er et udtryk af type **boolean** (f.eks. `n<=10`); hvis den udelades betyder det **true**
En *optælling* er normalt en tildeling (f.eks. `n=n+1`) eller liste af tildelinger.

Ækvivalent med:

initialisering;
while (*betingelse*)
{
sætning;
optælling ;
}

Indlejrede løkker

En løkke i kroppen på en anden løkke kaldes *indlejret*. **Eksempel:**

```
final int MAX_ROWS = 7;
for (int row = 1; row <= MAX_ROWS; row++)
{
   for (int star = 1; star <= row; star++)
      System.out.print ("*");
   System.out.println();
}
```

giver følgende output:

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


Advarsel

Hvad går galt i følgende stump program?

```
int i = 1;

while (i < 7) { i = i - 1; }
```

Oversigt over Java sætninger

Sætnings-art	Eksempel
tildeling	`x = x + 1;`
if	`if (x != 0) y = 1/y;`
if-else	`if (x != 0) y = 1/x; else y = 0;`

switch	`switch (x) {`
	`  case 0: y = 0; break;`
	`  case 1: y = 117; break;`
	`  default: y = 1/x; break`
	`}`

metodekald	`System.out.println("Hello");`
blok	`{ x = 1; y = 1/x; System.out.print(y); }`
for-løkke	`for (int i=1; i<100; i=i+1) sum = sum + i;`
while-løkke	`while (sum < 100) sum = sum + 1;`
do-while-løkke	`do { sum = sum + 1;} while (sum < 100);`

Sætninger kan kun skrives inde i en metode (f.eks. i `main`), ikke ude i en klasse.

Lidt om programudvikling

Udvikling af et program består af følgende fire faser:

Analyse: fastlæggelse af kravene til programmet – HVAD skal programmet lave?

Design: HVORDAN skal programmet opfylde kravene?

Implementering: skrivning af programmet, idet designet omformes til det valgte programmeringssprog.

Afprøvning: kørsel af programmet med mange forskellige inputs og kontrol af output.

I kan lære meget mere om dette i kursus *02260 Programmeludvikling*.

Programmeludvikling: eksempel (opgave 8)

Krav fra kunden

Lav et program, der (1) indlæser et password og (2) checker, at længden af password'et er større end 4 og mindre end 9.

Programmeludvikling: eksempel, fortsat

Analyse

- Skal programmet være interaktivt (dialog på skærmen)?
- Hvad skal programmet svare, hvis password er lovligt?
- Hvad skal programmet svare, hvis password er ulovligt?
  Skal det f.eks. blot svare "Password er ulovligt", eller skal det angive på hvilken måde det er ulovligt?

Efter at have talt med kunden, finder vi ud af, at svarene er:

- Ja.
- "Passwordets længde er korrekt".
- "Passwordets længde er ikke korrekt"

Programmeludvikling: eksempel, fortsat
Design 1

algoritmen kan angives ved pseudo-kode:

indlæs et password;

check password og udskriv resultat;

Programmeludvikling: eksempel, fortsat
Mere detaljeret design

indlæs et password;

hvis (password er ok) så udskriv "Passwordets længde er korrekt"
ellers udskriv "Passwordets længde er ikke korrekt"

Programmeludvikling: eksempel, fortsat
Implementering

```
import cs1.Keyboard;

public class Password {

public static void main(String[] args) {
    //indlaes et password
    System.out.print("Skriv et password: ");
    String password = Keyboard.readString();

    //Check password og udskriv resultatet
    int plength=password.length();
    if (plength>4 && plength<9)
      System.out.println("Passwordets laengde er korrekt");
    else
      System.out.println("Passwordets laengde er ikke korrekt");
}}
```


Programmeludvikling: eksempel, fortsat
Afprøvning (test)

1. Lav en *test suite*, dvs. en række eksempler på inddata samt forventet uddata.
2. Kør programmet med de valgte indddata og check, at uddata bliver som forventet. Hvis ikke, så gå tilbage og ret design og implementering.

To former for afprøvning:

	Intern afprøvning	Ekstern afprøvning
Udgangspunkt	programteksten	problemstillingen
Fundne fejltyper	logikfejl	oversete tilfælde
	fejlinitialiserede variable	oversete krav

Intern afprøvning: Valg- og gentagelsessætninger

Sætningstype	Tilfælde der skal afprøves
`if`	Betingelse falsk og sand
`switch`	Hver gren skal have været udført
`for`	Nul, ét, og flere gennemløb
`while`	Nul, ét, og flere gennemløb
`do-while`	Ét og flere gennemløb

Programmeludvikling: eksempel, fortsat
Afprøvning: eksempel på test suite

Test 1

```
Skriv et password: pass22
Passwordets laengde er korrekt
```

Test 2

```
Skriv et password: hej
Passwordets laengde er ikke korrekt
```

Test 3

```
Skriv et password: hejmeddig
Passwordets laengde er ikke korrekt
```


Husk til næste tirsdag

- Læs afsnit 4.1-4.4.
- Forbered opgaverne.
- Afhent cd, hvis du har bestilt en.