



CHALMERS

Objektorienterad programmering

Iteration: `while`-satsen, `for`-satsen och `do`-satsen

Dr. Alex Gerdes | Dr. Carlo A. Furia

Hösttermin 2016

Chalmers University of Technology

- Variabler, namngivning, konvention
- Datatyper
- Gränssnitt av en metod
- Scanner och printf
- if-satsen

Begränsad storlek

Datotypen `double`

- De enkla datatyperna, som används för att lagra tal (t.e. `int` och `double`), har en *begränsad storlek* och representerar således endast en delmängd av de verkliga talen
- Detta innebär att ett resultat eller ett mellanresultat från en beräkning av ett uttryck kan bli ett värde som inte kan lagras
- Resulterar uttrycket i ett för stort värde uppstår *overflow* och om uttrycket resulterar i ett för litet värde uppstår *underflow*
- Reella tal lagras med ett bestämt antal signifikanta siffror, vilket innebär att det vid beräkningar uppstår *trunkeringsfel*



- **Exempel:** antag att vi lagrar decimala tal med 4 decimaler.
Uttrycket

$$1.0/3.0 + 1.0/3.0 + 1.0/3.0$$

kommer då att evalueras på följande sätt:

$$\begin{aligned} 1.0/3.0 + 1.0/3.0 + 1.0/3.0 &= \\ 0.3333 + 0.3333 + 0.3333 &= \\ 0.9999 & \end{aligned}$$

- resultatet blir inte 1.0, som man kanske trott!



När man handhar reella tal skall man aldrig jämföra på exakthet, utan istället jämföra på tillräcklig noggrannhet.

- I en triangel kan man beteckna sidorna a , b och c
- Om man känner till längden av sidorna a och b samt vinkeln β mellan dessa sidor, kan man räkna ut längden av sidan c med hjälp av formeln:

$$c = \sqrt{a^2 + b^2 - 2ab \cdot \cos \beta}$$

- Skriv ett program som läser in längderna på två sidor i en triangel och vinkeln mellan dessa sidor (uttryckt i grader)
- Programmet skall avgöra om triangeln är *liksidig* (alla sidor lika långa), *likbent* (två sidor lika långa) eller *oliksidig* (inga sidor är lika långa)
- Programmet skall ge lämpliga utskrifter

- **Indata:** längderna a och b på två sidor i triangeln, samt den mellanliggande vinkeln v_g i grader
- **Utdata:** Utskrift av huruvida triangeln är liksidig, likbent eller oliksidig
- **Design:**
 - I klassen `Math` i Java anges parametrarna till de trigonometriska funktionerna i radianer. Därför är en omvandling av vinkeln från grader till radianer nödvändig att göra.
 - Denna omvandling kan antingen göras genom att använda metoden `toRadians` i klassen `Math`, eller genom att använda formeln

$$v_r = v_g \cdot \pi / 180$$

där v_r är vinkeln uttryckt i radianer och v_g är vinkeln uttryckt i grader

- För att två sidor skall betraktas som lika antas att sidornas längder skiljer sig med mindre än ϵ längdenheter

- **Algoritm:**

1. Läs längderna på sidorna a och b , samt gradtalet v_g av den mellanliggande vinkeln
2. Beräkna v_r som den mellanliggande vinkeln uttryckt i radianer
3. Beräkna längden c av den okända sidan i triangeln med hjälp av formeln

$$c = \sqrt{a^2 + b^2 - 2ab \cos(v_r)}$$

4. Om $|a - b| \leq \epsilon$ och $|a - c| \leq \epsilon$ och $|b - c| \leq \epsilon$ så skriv ut att triangeln är liksidig
annars om $|a - c| \leq \epsilon$ eller $|a - b| \leq \epsilon$ eller $|b - c| \leq \epsilon$ så skriv ut att triangeln är likbent
annars skriv ut att triangeln är oliksidig

Varför måste vi göra jämförelsen $|b - c| \leq \epsilon$?

- **Datarepresentation:**

- Längderna a , b och c samt vinklarna v_g och v_r är av datatypen `double`
- Konstanten `PI`, för att avbilda π finns tillgänglig i klassen `Math`
- Klassen `Math` har en klassmetod `toRadians` för att omvandla från grader till radianer
- Konstanten `EPS = 0.001`, för att avbilda ϵ

Implementation

```
/* Programmet läser in längderna av två sidor i en triangel samt mellanliggande
 * vinkel, och skriver ut huruvidatriangeln är liksidig, likbent eller oliksidig. */

import javax.swing.*;
import java.util.*;

public class Triangle {
    public static void main(String[] arg) {
        final double EPS = 0.001;
        String input = JOptionPane.showInputDialog("Ange längden av två sidor samt " +
            " mellanliggande vinkel: ");
        Scanner sc = new Scanner(input);
        double a = sc.nextDouble();
        double b = sc.nextDouble();
        double vg = sc.nextDouble();
        double vr = Math.toRadians(vg);
        double c = Math.sqrt(a*a + b*b - 2.0*a*b*Math.cos(vr));

        if (Math.abs(a-b) <= EPS && Math.abs(a-c) <= EPS && Math.abs(b-c) <= EPS)
            JOptionPane.showMessageDialog(null, "LIKSIDIG");
        else if (Math.abs(a-b) <= EPS || Math.abs(a-c) <= EPS || Math.abs(b-c) <= EPS)
            JOptionPane.showMessageDialog(null, "LIK BENT");
        else
            JOptionPane.showMessageDialog(null, "OLIKSIDIG");
    }
}
```

Iteration, while-satsen

- **Sekvens:**

- tilldelningssatsen
- selektionssatser
- iterationssatser
- return-satsen
- anrop av void-metod
- programblock
- exception

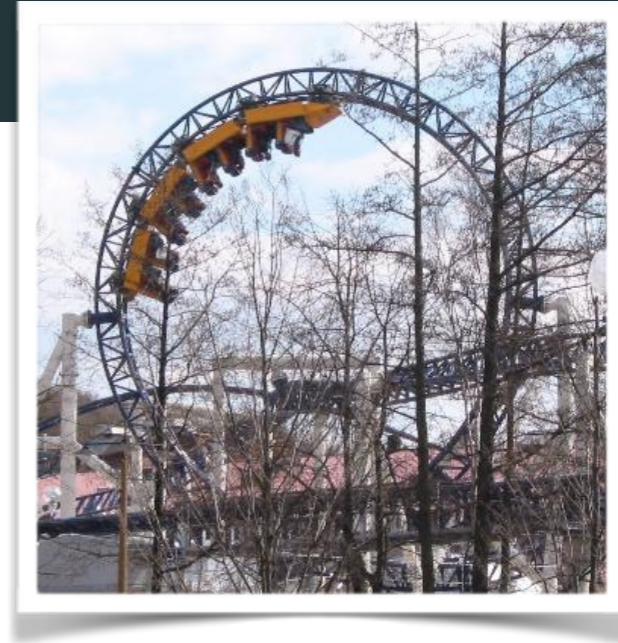
- **Selektion:**

- if-satsen
- switch-satsen

- **Iteration:**

- while-satsen
- do-satsen
- for-satsen

Olika typer av iteration



- Olika typer:
 - A. tills ett visst villkor blivit uppfyllt (*villkorsloop*)
 - B. ett på förhand bestämt antal gånger (*räkneloop*)
- Exempel:
 - A. “Rör om tills smöret har smält” (*villkorsloop*)
 - B. “Gå fem steg framåt” (*räkneloop*)

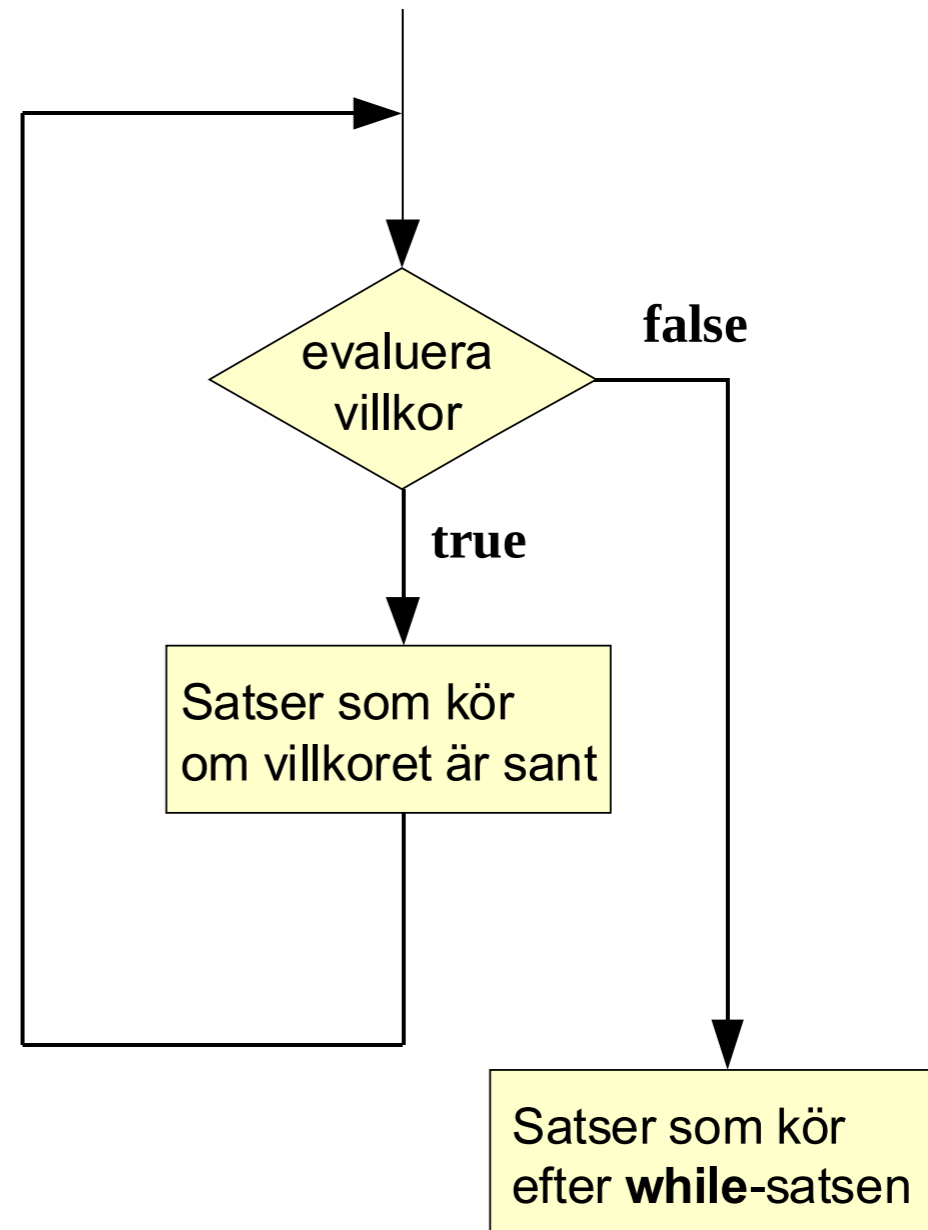
Iteration: `while`-satsen

- Upprepning av en sats:

```
while (villkor)  
  sats;
```

- Upprepning av ett programblock:

```
while (villkor) {  
  sats1;  
  ...  
  satsN;  
}
```



Vad gör programmet?

```
import java.util.Scanner;

public class Question {
    public static void main(String[] args) {
        Scanner sc    = new Scanner(System.in);
        boolean done = false;

        while (!done) {
            System.out.print("Which language do we learn: ");
            String answer = sc.next();
            if (answer.equals("Java"))
                done = true;
            else
                System.out.println("Nope, try again.");
        }
        System.out.println("Well done!");
    }
}
```

- Antalet bakterier y_n i en bakterieodling efter t tidsenheter ges av formeln

$$y_n = y_s e^{1.386t}$$

- där y_s är antalet bakterier vid $t = 0$
- Skriv ett program som beräknar hur många tidsenheter det tar innan en bakterieodling som innehåller en bakterie innehåller minst 1 miljard bakterier

- **Indata:** ingen
- **Utdata:** hur lång tid det tar innan bakterieodlingen innehåller minst 1 miljard bakterier
- **Algoritm:**
 1. Sätt tillväxttiden `tid` till 0
 2. Sätt `yStart`, antalet bakterier vid tiden 0, till 1
 3. Sätt totala antalet bakterier i odlingen `yTotal` till `yStart`
 4. Upprepa så länge som `yTotal` är mindre än 1.000.000.000
 - 4.1. Öka `tid` med 1
 - 4.2. Beräkna `yTotal` mha formeln
$$yTotal = yStart * e^{1.386 * tid}$$
 5. Skriv ut värdet av `tid`
- **Datarepresentation:**
 - `tid` är av datatypen `int`
 - `yStart` och `yTotal` är reella tal, dvs av datatypen `double`

Implementation

```
import javax.swing.*;

public class Bacteria {
    public static void main(String[] args) {
        int time = 0;
        double yStart = 1;
        double yTotal = yStart;

        while (yTotal < 1.0e9) {
            time = time + 1;
            yTotal = yStart * Math.exp(1.386 * time);
        }

        JOptionPane.showMessageDialog(null,
            "Det tar " + time + " tidsenheter innan " +
            "odlingen innehåller 1 miljard bakterier.");
    }
}
```



for-satsen

Iteration: for-satsen

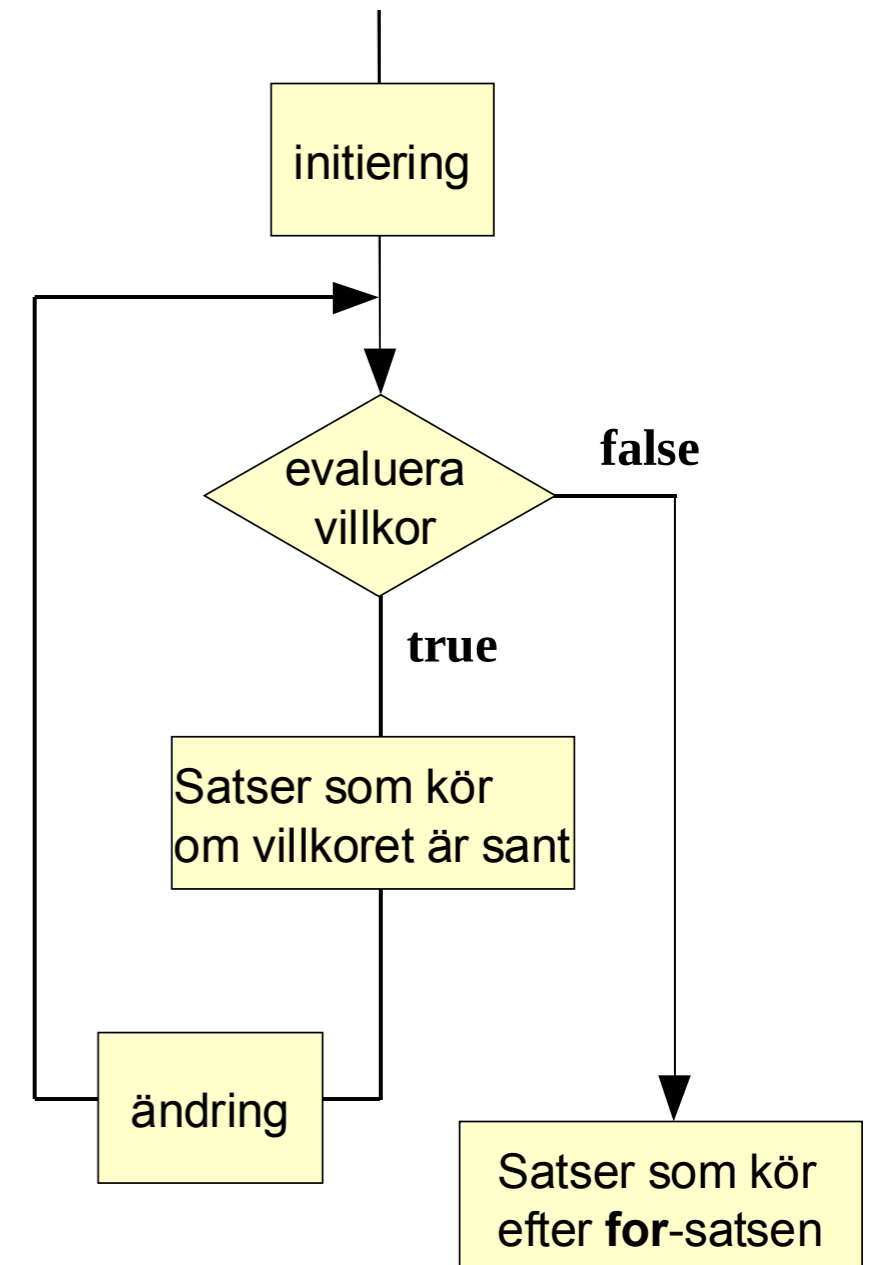
- Upprepning av en sats:

```
for (initiering; villkor; ändring)  
  sats;
```

- Upprepning av ett programblock:

```
for (initiering; villkor; ändring) {  
  sats1;  
  ...  
  satsN;  
}
```

*for-satsen är en villkorsloop,
men använd for-satsen
endast vid räkneloopar!*



```
for (int i = 1; i <= 5; i = i + 1)
    System.out.println(i);
```

Ger utskriften:

1
2
3
4
5

```
for (char c = 'd'; c >= 'a'; c = (char) (c - 1))
    System.out.println(c);
```

Ger utskriften:

d
c
b
a

```
for (double x = 0.5; x < 0.8; x = x + 0.1)
    System.out.println(x);
```

Ger utskriften:

0.5
0.6
0.7
0.7999999999...

Troligen inte förväntat resultat!
Använd endast uppräkningsbara
datatyper för att styra en for-sats!

Problemem exempel

- Att omvandla en temperatur angiven i grader Fahrenheit till grader Celsius sker med följande formel:

$$C = \frac{5(F - 32)}{9}$$

där C anger grader Celsius och F grader Fahrenheit

- Skriv ett program som skriver ut en omvandlingstabell från grader Fahrenheit till grader Celsius i intervallet från -10.0 till +10.0 grader Fahrenheit, där varje grad anges

Farenheit	Celsius
-10.0	-23.3
-9.0	-22.8
...	...
9.0	-12.8
10.0	-12.2

- **Indata:** ingen
- **Utdata:** en konverteringstabell från grader Fahrenheit till grader Celcius som anger varje grad i intervallet -10.0 till +10.0
- **Algoritm:**
 1. Skriv ut rubriken
 2. För $f = -10$ till 10 stega 1
 - 2.1. $c = (f - 32) * 5.0 / 9.0$
 - 2.2. Skriv ut f och c
- **Datarepresentation:** f är en `int` och c en `double`

Implementation

```
public class FahrenheitToCelsius {
    public static void main(String[] args) {
        System.out.printf("%15s%12s", "Fahrenheit", "Celsius\n");

        for (int fahrenheit = -10; fahrenheit <= 10; fahrenheit = fahrenheit + 1) {
            double celsius = (fahrenheit - 32) * 5.0 / 9.0;
            System.out.printf("%12.1f%13.1f\n", (double)fahrenheit, celsius);
        }
    }
}
```

Problemexempel

- Skriv ett program som läser in två heltal, som representerar antal rader respektive antal kolumner, och skriver ut nedanstående mönster i kommandofönstret:

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

rader

kolumner

- **Indata:** antal rader och kolumner
- **Utdata:** en utskrift med r gånger k rektangel av stjärnor
- **Algoritm:**
 1. Läs antal rader r och antal kolumner k
 2. För varje rad
 - 2.1. För varje kolumn
 - 2.1.1. Skriv ut tecknet ' * '
 - 2.2. Skriv ut radslut
- **Datarepresentation:** r och k är en `int`

Live coding

Implementation

```
import javax.swing.*;
import java.util.*;

public class WriteStars {
    public static void main(String[] args) {
        String input =
            JOptionPane.showInputDialog("Ange antal rader och antal kolumner: ");
        Scanner sc = new Scanner(input);

        int nrOfRows    = sc.nextInt();
        int nrOfColumns = sc.nextInt();

        for (int row = 0; row < nrOfRows; row = row + 1) {
            for (int col = 0; col < nrOfColumns; col = col + 1) {
                System.out.print("*");
            }
            System.out.println();
        }
    }
}
```

do-satsen

Iteration: do-satsen

- I en `while`-satsen beräknas testuttrycket inför varje varv i loopen
- I en `do`-satsen beräknas testuttrycket efter varje varv i loopen
- En `do`-sats genomlöps minst en gång, medan en `while`-sats kan genomlöpas noll gånger
- Upprepning av en sats:

do

```
    sats;
```

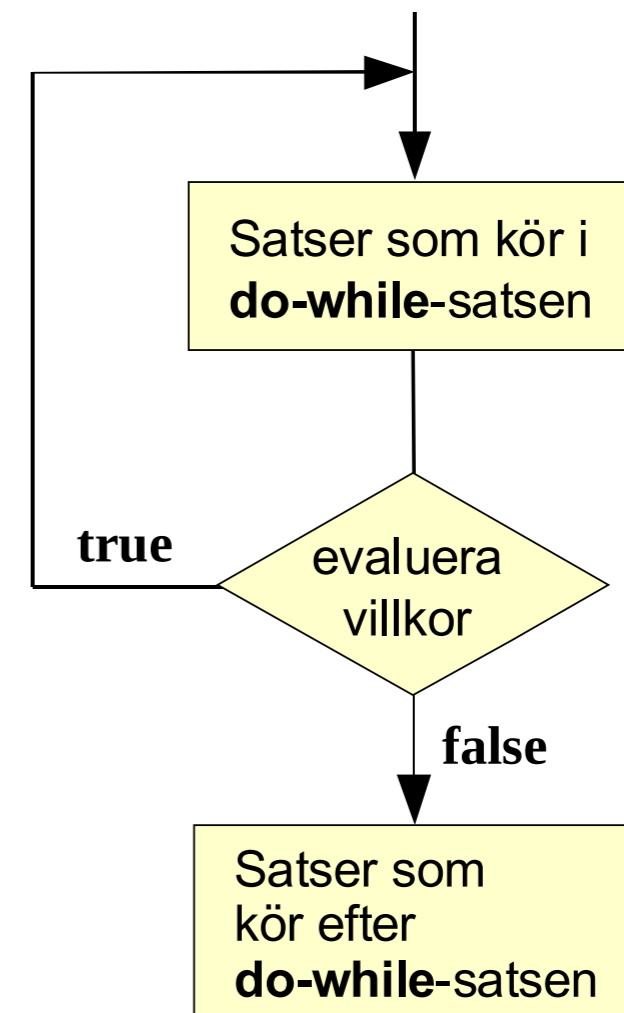
```
while (villkor);
```

- Upprepning av ett programblock:

```
do {
```

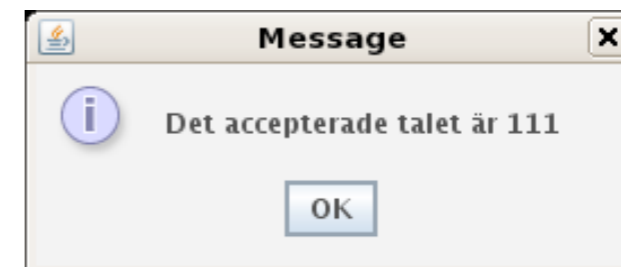
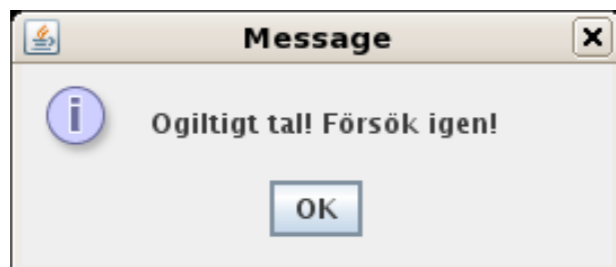
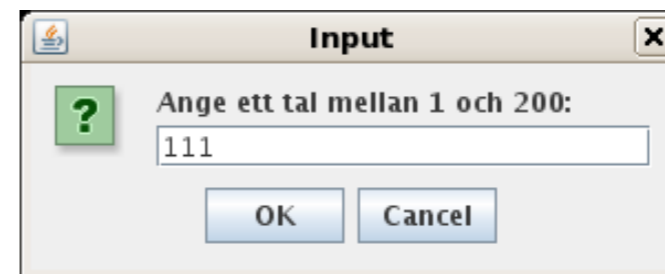
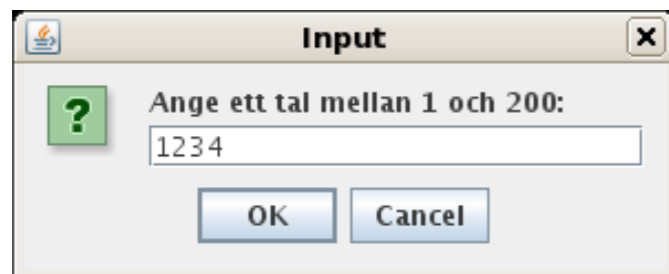
```
    satser
```

```
} while (villkor);
```



Exempel: Inläsningskontroll

```
import javax.swing.*;
public class InspectInput {
    public static void main(String[] args) {
        int number;
        do {
            String input = JOptionPane.showInputDialog("Ange ett tal mellan 1 och 200:");
            number = Integer.parseInt(input);
            if (number < 1 || number > 200)
                JOptionPane.showMessageDialog(null, "Ogiltigt tal! Försök igen!");
        } while (number < 1 || number > 200);
        JOptionPane.showMessageDialog(null, "Det accepterade talet är " + number);
    }
}
```



Variablers räckvidd (scope)

- En variabels *räckvidd* är det kodavsnitt inom vilket variabeln går att använda
- Grundprincip: En variabel är synlig endast inom det programblock där variabeln deklarerats

```
public static void main (String[] arg) {  
    int number;  
  
    do {  
        String input = JOptionPane.showInputDialog("Ange ett tal mellan 1 och 200:");  
        number = Integer.parseInt(input);  
        if (number < 1 || number > 200)  
            JOptionPane.showMessageDialog(null, "Ogiltigt tal! Försök igen!");  
    } while (number < 1 || number > 200);  
  
    JOptionPane.showMessageDialog(null, "Det accepterade talet är " + number);  
}
```

input
okänd
här!

number och
input
kända här!

En variabels räckvidd skall begränsas så mycket som möjligt.

for-, while- eller do-satsen?

- `while`-satsen är mest *generella* iterationssatsen, eftersom `for`- och `do`-satsen enkelt kan simuleras med en `while`-sats
- använd `for`-satsen vid *räkneloopar*
- använd `do`-satsen om loopen skall gå *minst ett varv*
- använd `while`-satsen i alla övriga fall
- är du osäker på vilken iterationssats som skall väljas, välj `while`-satsen

```
for (initiering; villkor; ändring)
  sats;
  ↓
initiering;
while (villkor) {
  sats;
  ändring;
}
```

```
do
  sats;
while (villkor);
  ↓
sats;
while (villkor)
  sats;
```

Upprepad programkörning

Evighetsloop och satsen break

- Konstruktionen

```
while (true) {  
  satser  
}
```

innebär en evighetsloop

- En loop kan när som helst lämnas med hjälp av satsen break:

```
while (villkor) {  
  ...  
  if (villkor för att sluta)  
    break;  
  ...  
}
```

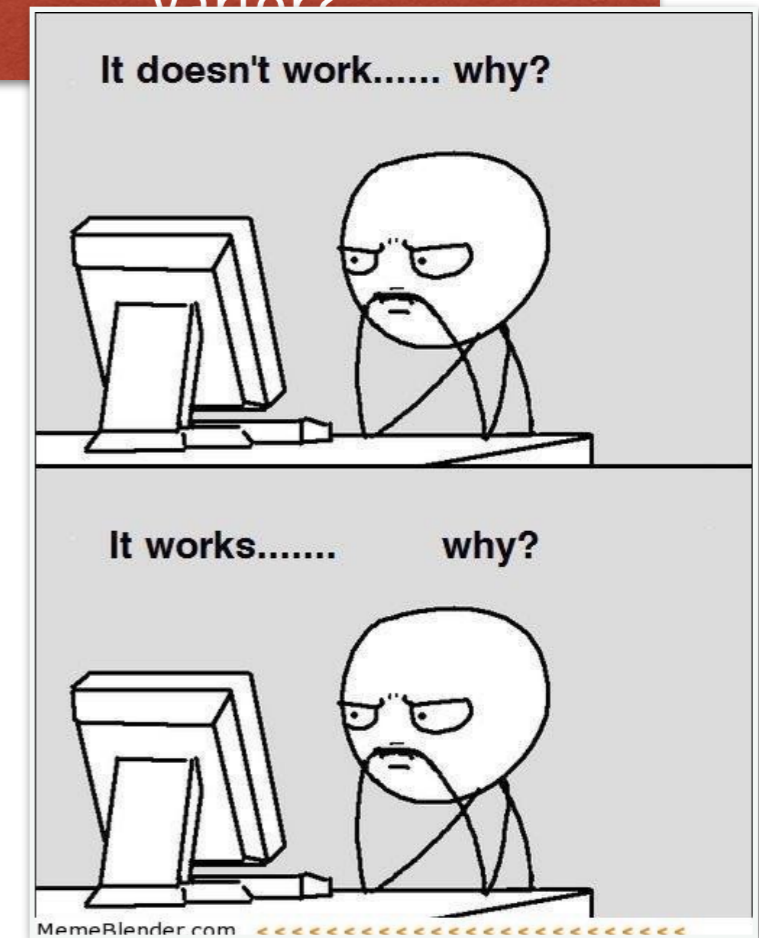
- Observera att vid nästlade loopar lämnas den loop i vilken break-satsen står!

```
while (villkor) {  
  ...  
  while (villkor) {  
    ...  
    if (villkor för att sluta)  
      break;  
    ...  
  }  
  ...  
}
```

En evighetsloop uppkommer vanligtvis på grund av ett misstag hos programmeraren, men kan ibland vara avsiktlig.

Att lämna en loop via en break-sats, skall användas mycket restriktivt!

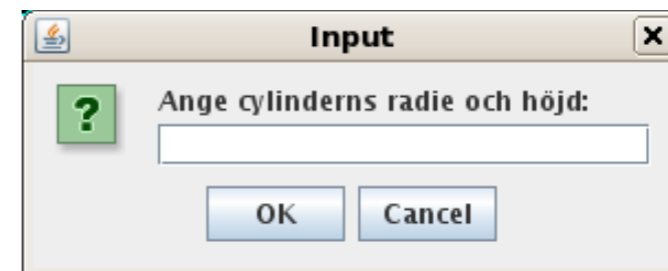
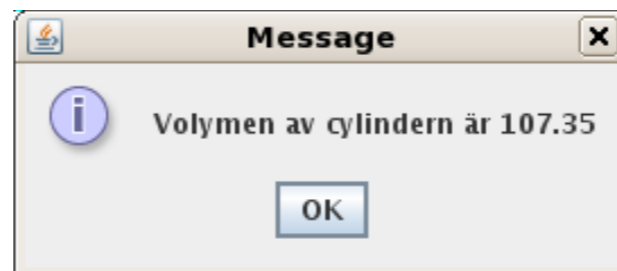
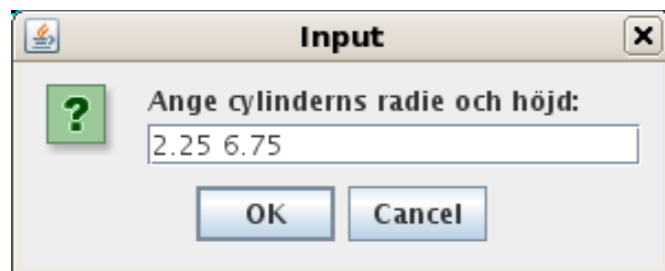
Varför?



Upprepad programkörning

```
import javax.swing.*;
import java.util.*;
public class RepeatedExecution {
    public static void main(String[] args) {
        boolean done = false;
        while (!done) {
            String input = JOptionPane.showInputDialog("Ange cylinderns radie och höjd:");
            if (input == null)
                done = true;
            else {
                Scanner sc = new Scanner(input);
                double radius = sc.nextDouble();
                double height = sc.nextDouble();
                double volume = Math.PI * Math.pow(radius, 2) * height;
                String output = String.format("%s %.2f", "Volymen av cylindern är", volume);
                JOptionPane.showMessageDialog(null, output);
            }
        }
    }
}
```

Cancel-knappen
returnerar null



- Skriv ett program som läser en indataserie bestående av N positiva heltal samt beräknar och skriver ut medelvärdet av dessa tal

- **Analys:**

- **Indata:** antalet tal i dataserien samt själva dataserien
- **Utdata:** medelvärdet av talen i dataserien
- **Speciella åtgärder:** om inga tal ingår i dataserien kan inte medelvärdet beräknas

- **Design:**

- **Utkast till algoritm:**

1. Läs in antalet heltal som ingår i dataserien till variabeln `antal`
2. Läs talen och beräkna talens sammanlagda summa i variabeln `summa`
3. Beräkna medelvärdet `medel` mha formeln $medel = summa / antal$
4. Skriv ut medelvärdet, dvs värdet av variabeln `medel`

Räkneloop!

- **Algoritm:**

1. Läs in antalet heltal som ingår i dataserien till variabeln `antal`
2. Sätt `summa` till 0
3. Upprepa `antal` gånger
 - 3.1. Läs ett tal till variabeln `ta1`
 - 3.2. Addera `summa` och `ta1` och spara resultatet i `summa`
4. Om `antal > 0` så
 - 4.1. Beräkna medelvärdet `medel` mha formeln: $medel = summa / antal$
 - 4.2. Skriv ut medelvärdet `medel`annars
 - 4.3. Skriv ut att inga värden ingick i dataserien

- **Datarepresentation:**

- `antal`, `summa` och `ta1` är heltal av typen `int`
- `medel` är ett reellt tal av typen `double`

Implementation med en for-sats

```
import javax.swing.*;

public class AddValues {
    public static void main(String[] args) {
        String input = JOptionPane.showInputDialog("Ange antalet tal i serien:");
        int number = Integer.parseInt(input);
        int sum = 0;

        for (int i = 0; i < number; i = i + 1) {
            input = JOptionPane.showInputDialog("Ange tal nr " + i + ": ");
            int value = Integer.parseInt(input);
            sum = sum + value;
        }

        if (number > 0) {
            double mean = (double) sum / number;
            JOptionPane.showMessageDialog(null, "Medelvärde av talen är " + mean);
        } else {
            JOptionPane.showMessageDialog(null, "Inga tal ingick i serien!");
        }
    }
}
```

Implementation med en `while`-sats

```
import javax.swing.*;

public class AddValues2 {
    public static void main(String[] args) {
        String input = JOptionPane.showInputDialog("Ange antalet tal i serien:");
        int number = Integer.parseInt(input);
        int sum = 0;

        int i = 0;
        while (i < number) {
            i = i + 1;
            input = JOptionPane.showInputDialog("Ange tal nr " + i + ": ");
            int value = Integer.parseInt(input);
            sum = sum + value;
        }

        if (number > 0) {
            double mean = (double) sum / number;
            JOptionPane.showMessageDialog(null, "Medelvärdet av talen är " + mean);
        } else {
            JOptionPane.showMessageDialog(null, "Inga tal ingick i serien!");
        }
    }
}
```

- Skriv ett program som läsa och beräkna medelvärdet av ett *okänt* antal *positiva* heltal. Serien av heltalen avslutas med ett negativt heltal (vilket inte ingår i serien).
- **Analys:**
 - **Indata:** talen i dataserien som skall läsas in, samt ett negativt tal som avbryter inläsningen
 - **Utdata:** medelvärdet av talen som ingår i dataserien
 - **Speciella åtgärder:** om inga tal ingår i dataserien kan inte medelvärdet beräknas

- **Algoritm:**

1. Sätt `summa` till 0 och `antal` till 0
2. Läs ett tal till variabeln `tal`
3. Upprepa så länge som `tal ≥ 0`
 - 3.1. Öka `antal` med 1
 - 3.2. Addera `summa` och `tal` och spara resultatet i `summa`
 - 3.3. Läs ett tal till variabeln `tal`
4. Om `antal > 0` så
 - 4.1. Beräkna medelvärdet `medel` mha formeln: $medel = summa / antal$
 - 4.2. Skriv ut medelvärdet `medel`annars
 - 4.3. Skriv ut att inga värden ingick i dataserien

- **Datarepresentation:**

- `antal`, `summa` och `tal` är heltal av typen `int`
- `medel` är ett reellt tal av typen `double`

Implementation

```
import javax.swing.*;

public class AddValues3 {
    public static void main(String[] args) {
        int number = 0;
        int sum = 0;
        String input = JOptionPane.showInputDialog("Ange tal nr " + (number + 1) + ":");
        int value = Integer.parseInt(input);

        while (value > 0) {
            number = number + 1;
            sum = sum + value;
            input = JOptionPane.showInputDialog("Ange tal nr " + (number + 1) + ": ");
            value = Integer.parseInt(input);
        }

        if (number > 0) {
            double mean = (double) sum / number;
            JOptionPane.showMessageDialog(null, "Medelvärdet av talen är " + mean);
        } else {
            JOptionPane.showMessageDialog(null, "Inga tal ingick i serien!");
        }
    }
}
```

duplicerad
kod



- Skriv ett program som läsa och beräkna medelvärdet av ett okänt antal heltal
- **Analys:**
 - **Diskussion:** nu kan *alla* heltal ingå i dataserien och hur skall vi då kunna markera slutet på serien? Ett sätt är att utnyttja *Cancel*-knappen i dialogrutan. När man trycker på *Cancel* returneras värdet `null`.
 - **Indata:** talen i dataserien som skall läsas in; inläsningen avbryts genom att trycka på *Cancel*-knappen i dialogrutan
 - **Utdata:** medelvärdet av talen som ingår i dataserien
 - **Speciella åtgärder:** om inga tal ingår i dataserien kan inte medelvärdet beräknas

- **Algoritm:**

1. Sätt `summa` till 0 och `antal` till 0
2. Upprepa
 - 2.1. Gör en inläsning från dialogfönstret till variabeln `indata`
 - 2.2. Om `indata == null` gå till punkt 3
 - 2.3. Öka `antal` med 1
 - 2.4. Konvertera `indata` till heltalet `tal`
 - 2.5. Addera `summa` och `tal` och spara resultatet i `summa`
3. Om `antal > 0` så
 - 3.1. Beräkna medelvärdet `medel` med hjälp av formeln: $medel = summa / antal$
 - 3.2. Skriv ut medelvärdet `medel`annars
 - 3.3. Skriv ut att inga värden ingick i dataserien

- **Datarepresentation:**

- `antal`, `summa` och `tal` är heltal av typen `int`
- `medel` är ett reellt tal av typen `double`

Implementation

```
import javax.swing.*;

public class AddValues4 {
    public static void main(String[] args) {
        int number    = 0;
        int sum        = 0;
        boolean done   = false;

        while (!done) {
            String input = JOptionPane.showInputDialog("Ange nästa tal i serien\n" +
                                                       "Avsluta med Cancel");

            if (input != null) {
                int value = Integer.parseInt(input);
                number = number + 1;
                sum = sum + value;
            } else {
                done = true;
            }
        }

        if (number > 0) {
            double mean = (double) sum / number;
            JOptionPane.showMessageDialog(null, "Medelvärdet av talen är " + mean);
        } else {
            JOptionPane.showMessageDialog(null, "Inga tal ingick i serien!");
        }
    }
}
```