INF1000 : Forelesning 3

Lese fra terminal, formatert utskrift, forgreninger

5. september 2006

Ole Christian Lingjærde
Gruppen for bioinformatikk
Institutt for informatikk
Universitetet i Oslo

Når du løser oppgaver

1. **Bestem programmets oppførsel sett utenfra:**
   - Hva skal være inndata (input) til programmet?
   - Hvordan skal programmet få tak i inndataene?
   - Hva skal være utdata (output) fra programmet?
   - Hvordan skal utdataene presenteres for brukeren?
2. **Avgjør hvordan du skal transformere inndata til utdata:**
   - Hvordan skal inn- og utdata representeres (lagres) i programmet?
   - Reduser transformasjonen inndata -> utdata til en sekvens av trinn hvor hvert trinn gjør en enkel ting med dataene og hvor hvert trinn er enkelt å programmere.
3. **Skriv programkode (og test løsningen).**

Eksempel: Celcius og Fahrenheit

- Problem:

  I Norge angis vanligvis temperaturer i Celcius (C), mens man bl.a. i USA benytter Fahrenheit (F). F.eks. svarer 0 C til 32 F.

  Lag et program som lager en tabell som nedenfor (og med temperaturer i Fahrenheit fylt inn):

Celcius	Fahrenheit
-10.0	.....
0.0	.....
37.0	.....
100.0	.....

Hvilke data beskriver problemet?

- Inndata:
  - De fire Celcius-temperaturene -10, 0, 37 og 100 (desimaltall)
  - Vi tenker oss at temperaturene er gitt når vi skriver programmet. Senere skal vi se hvordan programmet kunne ha lest inndata fra terminal (fra brukeren).
- Utdata:
  - De tilsvarende (konverterte) Fahrenheit-temperaturene (desimaltall)
  - Skal skrives ut på skjermen i en tabell

Transformere inndata til utdata

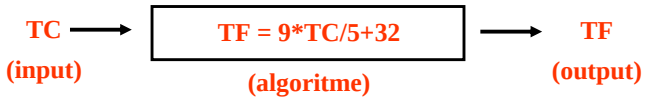
- Vi må kjenne formelen for å regne om fra Celcius til Fahrenheit. La

  TC = Temperatur i Celcius
  TF = Temperatur i Fahrenheit

  Vi finner i et oppslagsverk at omregningsformelen er

  TF = 9 * TC / 5 + 32

  Dermed blir fremgangsmåten slik:



Programskisse

```
class TemperaturKonvertering {
  public static void main (String[] args) {
    <deklarasjoner>

    <Skriv overskrift>

    <sett TC lik -10>
    <regn ut TF>
    <skriv ut>

    <sett TC lik 0>
    <regn ut TF>
    <skriv ut>

    <sett TC lik 37>
    <regn ut TF>
    <skriv ut>

    <sett TC lik 100>
    <regn ut TF>
    <skriv ut>
  }
}
```


Ferdig program

```
class TemperaturKonvertering {
  public static void main (String[] args) {
    double TC, TF;

    System.out.println("Celcius     Fahrenheit");

    TC = -10;
    TF = 9 * TC / 5 + 32;
    System.out.println(TC + "       " + TF);

    TC = 0;
    TF = 9 * TC / 5 + 32;
    System.out.println(TC + "       " + TF);

    TC = 37;
    TF = 9 * TC / 5 + 32;
    System.out.println(TC + "       " + TF);

    TC = 100;
    TF = 9 * TC / 5 + 32;
    System.out.println(TC + "       " + TF);

  }
}
```

Test programmet

Innlesning fra terminal

- Innlesning fra terminal kan gjøres på flere måter i Java. I INF1000 bruker vi pakken easyIO. Du må da skrive i toppen av programmet:

```
import easyIO.*;
```

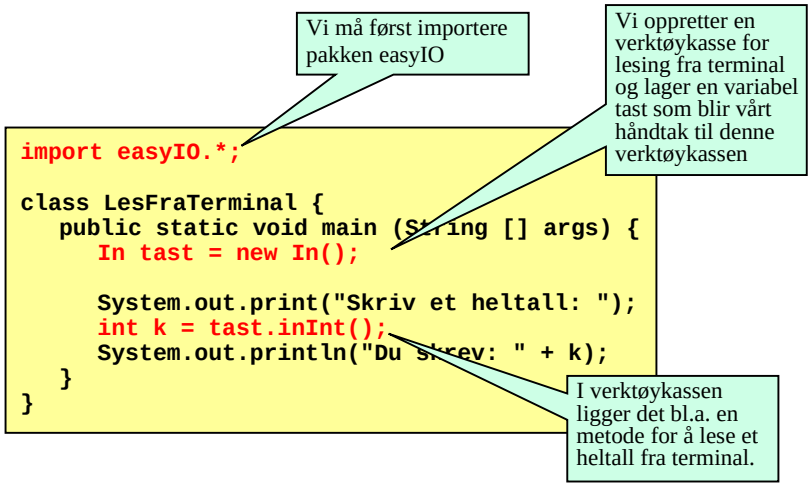
- Inne i klassen skriver vi følgende før vi kan starte innlesning:

```
In tastatur = new In();
```

- Så kan vi lese inn fra terminal (=tastatur), f.eks. et heltall:

```
int radius;
System.out.print("Oppgi radiusen: ");
radius = tastatur.inInt();
```

Eksempel




```
import easyIO.*;

class LesFraTerminal {
   public static void main (String [] args) {
      In tast = new In();

      System.out.print("Skriv et heltall: ");
      int k = tast.inInt();
      System.out.println("Du skrev: " + k);
   }
}
```


Lesemetoder

```
// Opprette forbindelse med tastatur:
In tastatur = new In();

// Lese et heltall:
int k = tastatur.inInt();

// Lese et desimaltall:
double x = tastatur.inDouble();

// Lese et enkelt tegn:
char c = tastatur.inChar();

// Lese et enkelt ord:
String s = tastatur.inWord();

// Lese resten av linjen:
String s = tastatur.inLine();
```


Hvilken lesemetode skal jeg velge?

- Først: importere easyIO og åpne forbindelse til tastaturet
- Lese item for item:
  - For å lese et heltall: inInt() [egentlig: tastatur.inInt()]
  - For å lese et desimaltall: inDouble()
  - For å lese ett ord: inWord()
  - For å lese en hel linje (med minst ett tegn): inLine()
- Lese linje for linje:
  - Bruk readLine()
- Lese tegn for tegn:
  - For å lese neste tegn (også hvite tegn): inChar()

Hvordan lesemetodene virker

- Metodene inInt(), inDouble() og inWord() virker slik:
  - De hopper over eventuelle innledende blanke tegn.
  - De leser så alt fram til neste blanke tegn eller linjeskift. Dersom det som leses ikke er et heltall når inInt() brukes eller et desimaltall når inDouble() brukes, gis det en feilmelding og man får en ny sjanse.
- Metoden inChar() virker slik:
  - Den leser neste tegn, enten det er et blankt tegn eller ikke.
- Metoden inLine() virker slik:
  - Den leser alt fram til slutten av linjen (inkludert blanke tegn), men ignorerer linjer hvor det kun står (igjen) et linjeskift.

Hvordan lesemetodene virker

Terminal-input: `_ _ x y z _ 1 6 1 2 7 5` `_ = blank`

`String s1 = tast.inWord();` `String s2 = tast.inWord();`	`s1: "xyz"` `s2: "161275"`
`String s1 = tast.inWord();` `int x = tast.inInt();`	`s1: "xyz"` `x :161275`
`String s = tast.inLine();`	`s: "  xyz 161275"`

`char c1 = tast.inChar();`
`char c2 = tast.inChar();`
`char c3 = tast.inChar();`
	`c1: ' '`
	`c2: ' '`
	`c3: 'x'`

`int x = tast.inInt();`	`feilmelding`

Eksempel: lese data om en person

- Problem:

  Lag et program som leser fra terminal disse dataene om en person:
  - Navn
  - Yrke
  - Alder

  og som skriver ut dataene på skjermen etterpå.

- Framgangsmåte:
  - Vi bruker inLine() til å lese navn og yrke, og inInt() til å lese alder.

Ferdig program

```
import easyIO.*;

class LesDataOmPerson {
  public static void main (String [] args) {
    String navn, yrke;
    int alder;
    In tast = new In();

    System.out.print("Navn: ");
    navn = tast.inLine();

    System.out.print("Yrke: ");
    yrke = tast.inLine();

    System.out.print("Alder: ");
    alder = tast.inInt();

    System.out.print("Hei " + navn + ", du er " + alder);
    System.out.println(" år gammel og jobber som " + yrke);
  }
}
```

Test programmet

Et eksempel til

```
import easyIO.*;

class LesFraTerminal2 {
  public static void main (String [] args) {
    In tastatur = new In();
    System.out.print("Skriv tre ord: ");
    String s1 = tastatur.inWord();
    String s2 = tastatur.inWord();
    String s3 = tastatur.inWord();
    System.out.println("Du skrev disse ordene:");
    System.out.println("  " + s1);
    System.out.println("  " + s2);
    System.out.println("  " + s3);
  }
}
```

Test programmet

Formatert utskrift til skjerm

- Formatert utskrift vil si at vi angir nøyaktig hvordan utskriften skal se ut og plasseres på skjermen.
- Kan gjøres "manuelt" med System.out.print(...), men det er upraktisk.
- Bedre: bruke en ferdiglaget pakke for slikt. I INF1000 bruker vi pakken easyIO. For å få tilgang til denne pakken må vi som før skrive helt i toppen av programmet vårt (før class):

```
import easyIO.*;
```

- Inne i klassen skriver vi følgende før vi kan starte formatert utskrift:

```
Out skjerm = new Out();
```

- Så kan vi skrive ut det vi ønsker, f.eks.:

```
double pi = 3.1415926;
skjerm.out(pi, 2, 6);   // Skriv ut pi med 2 desimaler,
                           høyrejustert på 6 plasser.
```


Eksempel

Vi må først importere pakken easyIO

```
import easyIO.*;

class FormatertUtskrift {
   public static void main (String [] args) {
      Out skjerm = new Out();
      int BREDDE1 = 15;
      int BREDDE2 = 30;
      skjerm.out("NAVN", BREDDE1);
      skjerm.outln("ADRESSE", BREDDE2);
      skjerm.out("Kristin Olsen", BREDDE1);
      skjerm.outln("Vassfaret 14, 0773 Oslo", BREDDE2);
   }
}
```

Vi oppretter en verktøykasse for skriving til terminal

Test programmet

I verktøykassen ligger det bl.a. verktøy (på java-språk: *metoder*) for å skrive til skjerm med og uten linjeskift til slutt.

Tre måter å skrive ut på

- Uten formatering:

```
skjerm.out("Per Hansen");
skjerm.out(12345);
skjerm.out(3.1415, 2);
```

- Angi utskriftsbredde:

```
skjerm.out("Per Hansen", 15);      // Bredde 15 tegn
skjerm.out(12345, 15);             // Bredde 15 tegn
skjerm.out(3.1415, 2, 15);         // Bredde 15 tegn
```

- Angi utskriftsbredde og justering:

```
skjerm.out("Per Hansen", 15, Out.RIGHT);   // Høyrejuster
skjerm.out(12345, 15, Out.CENTER);         // Senterjuster
skjerm.out(3.1415, 2, 15, Out.LEFT);       // Venstrejuster
```


Programmer med forgreninger

- En svært nyttig programmeringsteknikk er å bruke forgreninger, dvs forskjellige instruksjoner utføres i ulike situasjoner.
- Vi kan få til dette med en if-setning:

```
if (logisk uttrykk)
{
  <instruksjoner>
}
else
{
  <instruksjoner>
}
```

f.eks. x < y eller et annet uttrykk som enten er true eller false

denne blir utført når det logiske uttrykket er sant (true)

denne blir utført når det logiske uttrykket er usant (false)

- Eksempel:

```
if (x > 0) {
  System.out.println("Tallet er positivt");
} else {
  System.out.println("Tallet er ikke positivt");
}
```

Programmer med forgreninger

- Else-delen kan utelates, slik som her:

```
if (pris > 1500) {
  System.out.println(”Det er for dyrt”);
}
```

- Vi kan legge if-setninger inni if-setninger:

```
if (lønn < 400000) {
  if (ferieuker < 8) {
    System.out.println(”Ikke søk på jobben”);
  }
}
```

- Vi kan lage sammensatte if-setninger av typen

```
if (a < 10) { // a er positivt heltall
  System.out.println(”Ett siffer”);
} else if (a < 100) {
  System.out.println(”To siffer”);
} else {
  System.out.println(”Mer enn to siffer”);
}
```


Eksempel på bruk av if-setning

Program som avgjør hvem av to personer som er høyest:

```
import easyIO.*;

class Hoyde {
  public static void main (String[] args) {
    In tastatur = new In();
    double høyde1, høyde2;

    System.out.print("Høyden til Per: ");
    høyde1 = tastatur.inDouble();
    System.out.print("Høyden til Kari: ");
    høyde2 = tastatur.inDouble();

    if (høyde1 > høyde2) {
      System.out.println("Per er høyere enn Kari");
    } else {
      System.out.println("Per er ikke høyere enn Kari");
    }
  }
}
```

Test programmet

Eksempel: Body Mass Index

Oppgave:

Body Mass Index (BMI) er et mål som kan regnes ut fra høyden og vekten til en person. Ifølge verdens helseorganisasjon (WHO)[1] :

BMI	Vektstatus
Under 18.5	Undervekt
18.5 – 24.9	Normal vekt
25.0 – 29.9	Overvekt
30.0 eller høyere	Fedme

Vi skal lage et program som beregner BMI ut fra høyde og vekt og gir melding om hvilken vektstatus (se tabellen) det tilsvarer.

[1]Se http://www.who.int/hpr/NPH/docs/gs_obesity.pdf

Inndata og utdata

- **Inndata:**
  - Personens høyde (i m)
  - Personens vekt (i kg)
  - Leses fra terminal
- **Utdata:**
  - BMI
  - Skrives ut på skjerm, sammen med en av beskjedene
    - *Undervekt* (hvis BMI <= 18.4)
    - *Normal vekt* (hvis 18.5 <= BMI <= 24.9)
    - *Overvekt* (hvis 25.0 <= BMI <= 29.9)
    - *Fedme* (hvis BMI >= 30.0)

Transformere inndata til utdata

- Vi må kjenne formelen for å regne ut BMI. La

  vekt = personens vekt i kg
  hoyde = personens høyde i m

  Da er

  BMI = vekt / (hoyde*hoyde)

Ferdig program

```
import easyIO.*;

class BodyMassIndex {
  public static void main (String[] args) {
    In tast = new In();

    System.out.print("Vekt (i kg): ");
    double vekt = tast.inDouble();

    System.out.print("Høyde (i cm): ");
    double høyde = tast.inDouble()/100;

    double bmi = vekt / (høyde * høyde);
    System.out.println("BMI = " + bmi);
```


Ferdig program (*forts.*)

```
    if (bmi <= 18.4) {
       System.out.println("Vektstatus: undervekt");
    } else if (bmi <= 24.9) {
       System.out.println("Vektstatus: normalvekt");
    } else if (bmi <= 29.9) {
       System.out.println("Vektstatus: overvekt");
    } else {
       System.out.println("Vektstatus: fedme");
    }

  }
}
```


Alternativ til if-else: switch

- En sammensetning av flere if-setninger kan i noen tilfeller erstattes med en switch-setning:

```
switch (uttrykk) {
  case verdi1:
     <instruksjoner>
     break;
  ....
  case verdiN:
     <instruksjoner>
     break;
  default:
     <instruksjoner>
}
```

Et uttrykk som gir en verdi som er av en av typene char eller int (evt. byte eller short)

- Nøkkelordet break avbryter utførelsen av switch-setningen. Når break mangler, fortsetter utførelsen på neste linje (det er sjelden ønskelig).

Eksempel

```
class BrukAvSwitch {
  public static void main (String [] args) {
    char c = 'x';

    switch(c) {
      case 'a':
        System.out.println("Tegnet var en a");
        break;
      case 'b':
        System.out.println("Tegnet var en b");
        break;
      default :
        System.out.println("Tegnet var ikke a eller b");
    }
  }
}
```


Oppgave 1

- Hva blir skrevet ut av dette programmet ? (les programmet nøye)

```
class IfTest {
   public static void main (String [] args) {
     String s = "Petter";

     if (s.equals("Jens"));
     {
       System.out.println("Ordet var " + s);
     }
   }
}
```

Test programmet

Oppgave 2

- Hva blir skrevet ut av dette programmet?

```
class IfTest2 {
   public static void main (String [] args) {
     double x = -0.5;
     double y = 0.5;

     if (Math.ceil(x) == Math.ceil(y)) {
       System.out.println("A");
     }
     if ((int) x == (int) y) {
       System.out.println("B");
     }
     if (x < y) {
       if (x < 0) {
         if (y < 0) {
           System.out.println("C");
         }
       } else {
         System.out.println("D");
       }
     }
   }
}
```

Test programmet

Oppgave 3

- Hva blir skrevet ut av dette programmet?

```
class Divisjon {
   public static void main (String [] args) {
     if (1/2 > 0) {
       System.out.println("A");
     } else {
       System.out.println("B");
     }
   }
}
```

Test programmet