

Grundlæggende Programmering

ved

IT-højskolen

Kurset, lærerne, forelæsningerne

- Lærer: Peter Sestoft (og Niels Hallenberg)
- Kursusforløb, se forelæsningsplanen
- Kursets hjemmeside er <http://www.it-c.dk/courses/GP/E2000/>
- Hjemmesiden indeholder udleverede materialer, sidste nyt, ...
- Forelæsningsplanen indeholder forelæsningsplancherne
- Forelæsningsplancherne udkommer (senest) dagen før forelæsningen
- **I skal holde øje med hjemmesiden hele semestret igennem**
- Stil endelig spørgsmål under forelæsningerne

Grundlæggende Programmering, efterår 2000 Forelæsning 1, onsdag den 30. august 2000

- Om kurset
- Om øvelserne
- Om nødvendige anskaffelser
- Om datamaskiner
- Om programmeringssprog
- Oversættelse og afvikling af et Java program
- Eksempler på Java programmer
- Programkommentarer

I får brug for

- Lærebog: Lewis & Loftus: Java Software Solutions, anden udgave (første udgave duer også)
- Lærebogen kan fås i Polyteknisk Boghandel og en række andre større boghandler. Pris ca. 500 kroner.
- Noter (udleveres trykt i løbet af semestret):
 - *Searching and sorting with Java*, 44 sider
 - *Systematic software test*, 14 sider
 - *Java precisely*, ca 60 sider (frivillig supplerende reference).
- Konto på IT-højskolens maskiner
- Ringbind til udleverede løbesedler, noter, forelæsningsresumeeer, mv.
- Disketter til programmer
- CD-ROM med nødvendige (gratis) programmer udlånes til installation på hjemme-pcer

Øvelser

- Man lærer kun for alvor ved at prøve selv.
- Læsevejledning: brug ikke for lang tid på bogens store eksempler. **Løs hellere opgaver** og fokuser på forelæsningerne.
- Øvelser: 2 timer pr. uge pr. studerende, fra onsdag den 6. september
- Øvelserne foregår på IT-højskolen, Glentevej 67
- Øvelsesidtpunkter: onsdag 13–15, 15–17 og 17–19 (samt torsdag hvis der er behov?)
- Instruktører: Thomas Jørgensen, Troels Nordfalk, Kasper Bøgebjerg Pedersen, Niels Hallenberg
- Skriv jer på øvelsesilmeldingen (rundsendes)
- Ugentlige opgaver til skriftlig aflevering. Eksempel:
 - Opgaver stilles onsdag 30/8 på løbesedlen
 - I forbereder opgaverne derhjemme
 - I kan spørge instruktørerne ved øvelserne onsdag 6/9
 - Besvarelserne skal afleveres fredag 8/9
 - Vi prøver at give de rettede opgaver tilbage onsdag 13/9
- Ud af semestret 11 sæt obligatoriske opgaver skal mindst 8 godkendes

Hvad er en datamaskine?

- materiel: chips, harddiske, floppydiske, tastatur, skærm, printer, cpu, ...
- programmel:
 - operativs/stem: Windows 98, Windows NT, Unix, PalmOS, Linux, ...
 - applikationsprogrammer: Microsoft Word, Star Office, Netscape, JavEdit, HelloWorld, ...

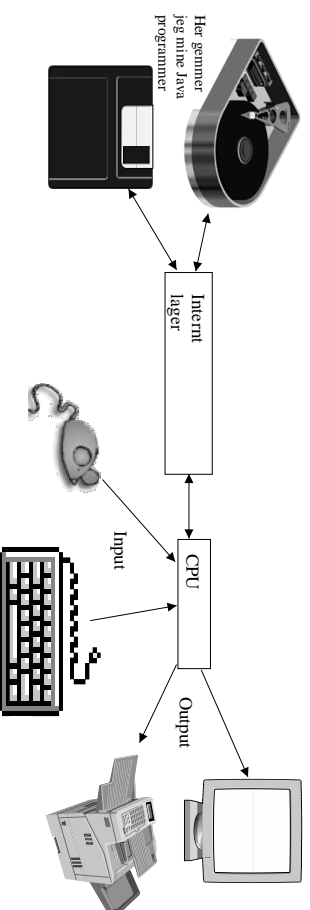
Eksempler på datamaskiner

- PC
- Høreapparat
- Lommeregner
- Palm Pilot
- Mobiltelefon
- iButton, Smartcards

Om kurset, motivation

- Kurset: programmering i sproget Java
- Kursusmål: I skal lære at skrive interessante programmer i Java
- Forudsætninger: I skal kunne bruge Windows, tekstbehandling og WWW
- Hvorfor lære programmering?
 - Fordi man kun på den måde virkelig forstår muligheder og begrænsninger i databehandling
 - Fordi det er nødvendigt i alle videregående kurser
 - Fordi man bedre kan forestille sig nye typer af løsninger og produkter
 - Fordi det er sejt
 - *Software developers are a hot commodity today, and they increasingly drive platform decisions within companies...* (IBM talsperson i New York Times, July 24, 2000)
 - *Know Java? It Could Help Your Salary: Salaries for IT professionals, especially those with Java training, continue to increase...* gennemsnitligt \$78750 i nordøst-USA (ACM TechNews, August 18, 2000).
- Hvorfor lære Java?
 - Java er velegnet til generel programmering (ligesom C, Pascal, C++, Basic, Standard ML, ...)
 - Java er velegnet til programmering af vinduesorienterede brugergrænseflader.
 - Java er velegnet til programmer der skal køre ude hos brugeren: Internethandel, hjemmebank, interaktiv grafik, chat, distribuerede spil, ...

Forenklet datamat



Hvad er et program?

Et program er interne instrukser til en datamaskine.

Interne instrukser kan medføre eksterne, synlige resultater.

F.eks. beregning og visning af en saldo, tegning af en graf: ...

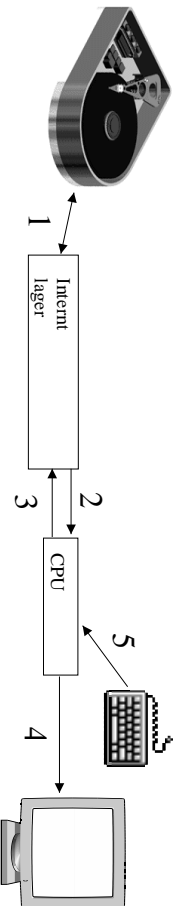
Afviklingen af et program er 100% mekanisk.

Programmer er generelle: kan løse uendelig mange problemer af en bestemt slags.

Programmer er stupide: kan ikke improvisere eller 'forstå'.

Datamaskiner er hurtige: derfor kan programmer alligevel virke smarte.

Hvordan udføres et program på en datamaskine?



1. Program indlæses fra harddisk til det interne lager.
2. Den første/ræste instruktion i programmet hentes ind i CPU'en. CPU'en udfører instruktionen.
3. CPU'en gemmer måske resultatet af en beregning i det interne lager.
4. CPU'en udskriver måske resultatet af en beregning på skærmen.
5. CPU'en modtager måske inddata fra en bruger.

Hvad er et programmeringssprog?

Et programmeringssprog er en notation for instrukser til datamaten, dvs. for programmer.

Man skal være meget omhyggelig med programmeringssprog.

Eksempel: Korrekt:

```
add (knap) ;
```

Helt forkert:

```
Add (knap) .
```

Et simpelt Java program

```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Det virker sandelig!");  
    }  
}
```

Bemærk:

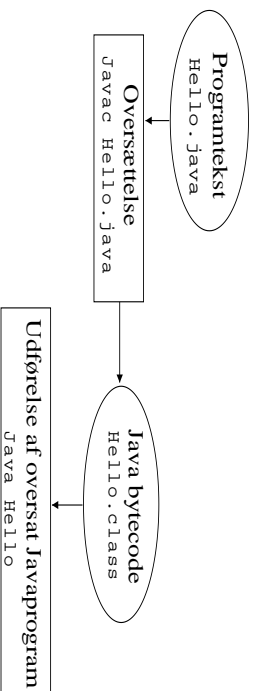
Programmets (klassens) navn er Hello; filens navn skal være Hello.java.

Programteksten i metoden main udføres når programmet startes.

Metodekaldet System.out.println udskriver til skærmen.

Betydningen af public static void og String[] args vil blive forklaret senere.

Redigering, oversættelse og udførelse af et Java program



1. Du skriver et Java program, f.eks. Hello.java.
2. Du oversætter Java programmet med `javac Hello.java` eller `javac Hello.java`.
3. Oversætteren `javac` (eller `javac`) genererer automatisk et program som kan afvikles på datamaten: `Hello.class`.
4. For at udføre programmet `Hello.class` benytter vi fortolkeren `java`.

- ### Redigering, oversættelse og udførelse af et Java program
- Rediger et Java program og gem det i filen `Hello.java`
 - Start en MS-DOS boks (kaldet Command Prompt i Windows NT).
 - Oversæt programmet til Java bytecode, ved brug af `javac` (eller `javac`) oversætteren:
`javac Hello.java`
- Derved dannes en fil `Hello.class` med den oversatte udgave af programmet.
- Udfør det oversatte program, ved brug af Java fortolkeren:
`java Hello`

Dette udfører programmet, som udskriver på skærmen.

Redigering af Java programmer med Javacedit

Javacedit er en simpel, gratis Java editor.

Den startes ved at man skriver `javacedit` i en DOS boks.

Rediger programmer (*.java filer), og oversæt dem ved klik på kaffekop-ikonet.

Programmer skal stadig udføres i MS-DOS boksen ved brug af Java fortolkeren.

Javacedit skal konfigureres før brug: se noten om **opsætning af PC** på kurssets hjemmeside.

Eksempel: Skat i år 2000

Bidrag og indkomstskat		Topstat	
Bundfradrag	267.600 kr.		15,0 pct.*
Bundfradrag	164.300 kr.**	Mellemskat	6,0 pct.
		Bundskat	7,0 pct.***
Personfradrag	33.400 kr.	Kommune-, amts- og kirkeskat	32,8 pct.****
Fradrag i indkomsten for arbejds-tilmarkedets- og særligt pensionsbidrag			
Arbejdsmarkeds- og særligt pensionsbidrag			1,0 pct.
Indkomst			6,0 pct.

15,0 pct. for evt. reduktion for skat skattelet. (13,9 pct. i en gennemsnitskommune i 2000).
** Mellemkattegrænsen forhøjes med 8.000 kr. snigti 2001 og 2002 ud over den normale regulering.
*** Bundskatteprocenten sænkes yderligere til 6,25 pct. i 2001 og til 5,5 pct. fra og med 2002.
**** Skattestaten i en gennemsnitskommune i 2000. Skattestaten varierer fra 27,6 pct. i den billigste til 35,85 pct. i landet direkte kommune.

Kilde: FORSTRÅ Skatten..., Skatteministeriet, november 1999

Eksempel: Beregning af AMBI og særligt pensionsbidrag

```
public class Skat1 {  
    public static void main(String[] args) {  
        int indkomst = 120000;  
        double ambi, pension;  
  
        ambi = indkomst * 8.0 / 100.0;  
        pension = indkomst * 1.0 / 100.0;  
  
        System.out.println("Arbejdsmarkedsbidrag: ");  
        System.out.println(ambi);  
        System.out.println("Særlig pensionsopsparing: ");  
        System.out.println(pension);  
    }  
}
```

Uddata fra programmet

```
Arbejdsmarkedsbidrag: 9600.0  
Særlig pensionsopsparing: 1200.0
```

Hvilken type har følgende værdier?

Værdi	Type
"GP"	
58	
true	
-23	
"afd"	
false	
42.0	
"42.0"	
"true"	
42 + 23	
23.0 - 12.0	

Grundbegreber: værdi og type

En værdi kan f.eks. være

- et heltal: 120000
- et kommat: 8.0
- en tegnstring: "Bundskat: "
- en sandhedsværdi: false eller true

En type er en familie af værdier:

int er typen af heltal: 3, 4, 10, 117, -2, 0, 120000, ...

double er typen af kommat: 1.0, 42.456, -32.3, 32E9, ...

String er typen af tegnstringe: "Bundskat: ", "Ole-Johan", "...

boolean er typen af sandhedsværdier: true, false

Grundbegreber: Variable, erklæringer og tildeling

En *variabel* kan indeholde en værdi af en given type.

En variabel svarer til et sted i datamaterns lager.

Du navngiver selv dine variable.

En *erklæring* indfører type og navn(e) for en eller flere variable:

```
double ambi, pension;
```

Erklæringen afsætter også plads af til variablerne i datamaterns lager.

En variabelerklæring kan indeholde en *initialisering*:

```
int indkomst = 120000;
```

En *tildelings-ordre* ændrer en variabels værdi ved at skrive i datamaterns lager:

```
ambi = indkomst * 8.0 / 100.0;
```

Hvilke variable, erklæringer, initialiseringer og tildelinger findes i programmet Skat1?

Datamatens lager ved afvikling af Skat-t1

```
int indkomst = 120000;  
  
ambi = indkomst * 8.0 / 100.0;  
pension = indkomst * 1.0 / 100.0;
```

Lager

```
Indkomst: 120000  
Ambi: 9600.0  
Pension: 1200.0
```

Konvertering mellem heltal og kommatial

Trunkeringen (int) ambi omdanner kommatiallet ambi til et heltal ved at smide decimalerne væk. Der rundes altid mod 0 – decimalerne smides væk.

Heltallet indkomst konverteres til et flydende-kommatial ved (double) indkomst eller ved at "regne" med et kommatial, f.eks. indkomst + 0.0.

```
public class Trunkering {  
    public static void main(String[] args) {  
        double t1 = 2.9234123412341234;  
  
        int t2 = (int)t1;  
        double t3 = (double)t2;  
        double t4 = t2 + 0.0;  
  
        System.out.println("t1 = "); System.out.println(t1);  
        System.out.println("t2 = "); System.out.println(t2);  
        System.out.println("t3 = "); System.out.println(t3);  
        System.out.println("t4 = "); System.out.println(t4);  
    }  
}
```

Grundbegreber: heltal og flydende-kommatial. Regner en datamat altid rigtigt?

I datamater er beregninger med flydende-kommatial kun omtrentlige, men beregninger med heltal er eksakte.

```
public class Komma {  
    public static void main(String[] args) {  
        double t = 2.1234123412341234;  
  
        System.out.println("    t = ");  
        System.out.println(t);  
        System.out.println("t + t = ");  
        System.out.println(t + t);  
    }  
}
```

Uddata fra programmet

```
t = 2.1234123412341233  
t + t = 4.2468246824682465
```

Derfor skelnes mellem flydende-kommatial (double) og heltal (int) i programmer.

Grundbegreber: Udtryk og ordre

Tilstanden består af datamatens lagerindhold, uddata (på skærmen), osv.

Et udtryk beregner en værdi:
indkomst * 8.0 / 100

En ordre ændrer tilstanden. For eksempel kan den ændre en variabels værdi (tildelings-ordre):
pension = indkomst * 1.0 / 100.0;
eller den kan skrive uddata på skærmen:
System.out.println("Særlig pensionsoptimering: ");

Flere erklæringer og ordre kan samles i en blok:

```
{ int indkomst = 120000; ... System.out.println(pension); }
```

En blok kan bruges hvor helst en ordre kan bruges.

Eksempel: Beregning af bundskat

Ambi	8.0 %	Særligt pension	1.0 %	Bundskat	7.0 %	Personfradrag	33.400 kr.
------	-------	-----------------	-------	----------	-------	---------------	------------

```
public class Skat2 {
    public static void main(String[] args) {
        int indkomst = 120000;
        double ambi, pension, bundskat;

        ambi = indkomst * 8.0 / 100.0;
        pension = indkomst * 1.0 / 100.0;

        indkomst = indkomst - (int)(ambi+pension);
        bundskat = (indkomst - 33400) * 7.0 / 100.0;

        System.out.print("Arbejdsmedlemsbidrag: "); System.out.println(ambi);
        System.out.print("Særligt pensionsopsparing: "); System.out.println(pension);
        System.out.print("Bundskat: "); System.out.println(bundskat);
    }
}
```

Hvordan ser lageret ud? Giver programmet også et rimeligt resultat for indkomst lig 10.000 kr?

```
Programmer skal kunne vælge
Man skal kun betale bundskat hvis indkomst minus ambi og særligt pensionsbidrag er mindst 33400 kroner.
For at kunne vælge mellem de to tilfælde skal vi bruge betingede ordre (if).

public class Skat3 {
    public static void main(String[] args) {
        int indkomst = 120000;
        double ambi, pension, bundskat;

        ambi = indkomst * 8.0 / 100.0;
        pension = indkomst * 1.0 / 100.0;
        int indkomst_etter_ap = indkomst - (int)(ambi+pension);

        if (indkomst_etter_ap > 33400)
            bundskat = (indkomst_etter_ap - 33400) * 7.0 / 100.0;
        else
            bundskat = 0.0;

        System.out.print("Bundskat: "); System.out.println(bundskat);
    }
}
```

Betingede ordre: if-else ordre

Generelt format:

```
if (udtryk)
    ordre1
else
    ordre2
```

Vikning:

- (1) beregn værdien af *udtryk*
- (2) hvis true så udlør *ordre1*, ellers udlør *ordre2*

Dvs. at if-else ordren vælger mellem to ordre, afhængig af *udtryk*.

```
Eisegrenen kan udelades når den ikke behøves

public class Skat4 {
    public static void main(String[] args) {
        int indkomst = 300000;
        double ambi, pension, bundskat = 0.0, mellemsskat = 0.0, topskat = 0.0;

        ambi = indkomst * 8.0 / 100.0;
        pension = indkomst * 1.0 / 100.0;

        int indkomst_etter_ap = indkomst - (int)(ambi+pension);
        if (indkomst_etter_ap > 33400)
            bundskat = (indkomst_etter_ap - 33400) * 7.0 / 100.0;
        if (indkomst_etter_ap > 164300)
            mellemsskat = (indkomst_etter_ap - 164300) * 6.0 / 100.0;
        if (indkomst_etter_ap > 267600)
            topskat = (indkomst_etter_ap - 267600) * 15.0 / 100.0;
        double skat = ambi + pension + bundskat + mellemsskat + topskat;

        System.out.print("Samlet skat: "); System.out.println(skat);
    }
}
```

En variabel kan opdateres mere end en gang

I stedet for at beregne skatterne hver for sig, lægger vi bare sammen undervejs.

Hvis f.eks. bundskatten er nul, lader vi bare være med at lægge den til.

```
public class Skat5 {
    public static void main(String[] args) {
        int indkomst = 300000;
        double skat;

        skat = indkomst * 8.0 / 100.0;
        skat = skat + indkomst * 1.0 / 100.0;

        int indkomst_etter_ap = indkomst - (int)(skat);
        if (indkomst_etter_ap > 33400)
            skat = skat + (indkomst_etter_ap - 33400) * 7.0 / 100.0;
        if (indkomst_etter_ap > 164300)
            skat = skat + (indkomst_etter_ap - 164300) * 6.0 / 100.0;
        if (indkomst_etter_ap > 267600)
            skat = skat + (indkomst_etter_ap - 267600) * 15.0 / 100.0;

        System.out.println("Samlet skat: "); System.out.println(skat);
    }
}
```

Hvordan ser lageret ud når programmet afvikles?

else-problemet (elseProblem.java)

Hvad giver dette hvis x er 4:

```
if (x > 5)
    if (y > 5)
        System.out.println("gren A");
    else
        System.out.println("gren B");
```

Regel: En else-gren knytter sig til den seneste uafsluttede if.

Man kan afslutte en if ved at lukke den inde i en blok:

```
if (x > 5) {
    if (y > 5)
        System.out.println("gren A");
    } else
        System.out.println("gren B");
```

Brug gerne ekstra { ... } til at gøre meningen med en ordre klar.

Ligesom man bruger ekstra (...) til at gøre meningen med et regneudtryk klar.

En blok består af flere ordrer (Skat6.java)

En række ordrer kan grupperes til én ordre med { ... }:

En blok kan forekomme hvor som helst en ordre kan forekomme.

```
if (indkomst_etter_ap > 267600) {
    skat = skat + (indkomst_etter_ap - 267600) * 15.0 / 100.0;
    System.out.println("Tak fordi du betaler topskat");
}
```

Hvad ville der ske her?

```
if (indkomst_etter_ap > 267600)
    skat = skat + (indkomst_etter_ap - 267600) * 15.0 / 100.0;
System.out.println("Tak fordi du betaler topskat");
```

Det er en hyppig fejl at glemme { ... } omkring de ordrer, der skal samles i en blok.

Programkommentarer

Erklæringer og ordrer er instrukser til datamaskinen.

En *programkommentar* har ingen virkning på datamaskinen.

Hvorfor skrive programkommentarer?

Af hensyn til programmører der skal læse programmet.

En-liniers kommentarer

```
// Solving the quadratic equation 2x^2 + 3x - 2 = 0
public class Quadratic1 {
    ...
}
```

Fler-liniers kommentarer

```
/* Solving the quadratic equation 2x^2 + 3x - 2 = 0
   sestof@dina.kvl.dk 1997-09-04 */
public class Quadratic1 {
    ...
}
```