INF1000: Grunnkurs i objektorientert programmering

Uke 0, høst 2015

Litt administrativt

- Studieadministrasjonen registrerer oppmøte i pausen – utenfor auditoriet.
- Mikronfonlyd og skjerm tas opp og legges ut
- Se semestersiden m/ lenker for informasjon!
  http://www.uio.no/studier/emner/matnat/ifi/INF1000/h15/

Informatikk-studiet og INF1000

- Ingen krav til forkunnskaper, men høye krav til jevnt arbeid!

- Tidligere erfaring?
  - Obs hull og alternative mentale modeller
  - Følg undervisning og løs obliger, ikke «mist toget»
  - MYE BAKGRUNN?
    => Henvend Dag Langmyhr på INF2100-forelesning onsdag
- Mål for emnet
  - Solid grunnlag for videre studier
  - Vekt på generelle begreper og grunnleggende mekanismer
  - Trening i programmering ved hjelp av disse i Java
  - Ikke et hurtigkurs i praktisk programmering!

Etter denne forelesningen skal du

- Ha noe kunnskap om begrepene informatikk, datasystem og programmering
- Ha noe kunnskap om objektorientert programmering og programmeringsspråket Java.
- Kunne skrive enkle Java-programmer med
  - kommentarer
  - utskrift-setninger til skjerm
  - deklarasjon og tilordning til heltallsvariable
- Kjenne til studieopplegg og læremidler for emnet

Informatikk

Informatikk er læren om hvordan datasystemer konstrueres og brukes*

et datasystem består av en eller flere *datamaskiner* som kjører *programvare* og kan være knyttet til et eller flere *nettverk* for overføring av data.

Informasjonsteknologi:
Informatikk handler om teknologi, men også mye mer!

* Dekan ved MatNat; Morten Dæhlen

Er informatikk viktig?

Justisminister Anders Anundsen oppnevner et utvalg som skal se på den digitale sårbarheten i Norge. FOTO: Audun Braastad / NTB scanpix

Regjeringen oppnevner digitalt sårbarhetsutvalg

Et utvalg som skal kartlegge samfunnets digitale sårbarhet oppnevnes i statsråd fredag. Rapporten skal være klar i september neste år.

NTB

Del Tweet E-post Lagre artikkelen i leselisten

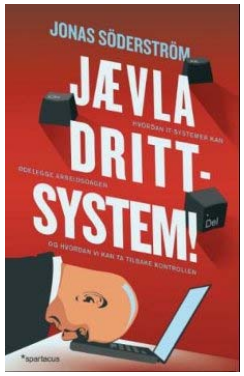
- Det er snart 15 år siden forrige sårbarhetsutvalg la fram sin rapport. Si… da har det skjedd enormt mye som påvirker den digitale sårbarheten vå… å si alt i det moderne Norge er digitalisert. Å nedsette et nytt utvalg nå … veldig viktig, slik at de får kartlagt hvor stor sårbarheten er og foreslått konkrete tiltak som kan redusere denne, sier justis- og beredskapsmin…

Olav Lysne, Simula Research/ UiO

.. krevende?

Fra Stortingsmelding 10(2012-2013):
God kvalitet – trygge tjenester
Kvalitet og pasientsikkerhet i helse- og omsorgstjenesten:

«.. Eksempler på områder der Norge skårer dårlig er informasjonsflyt, kommunikasjon og koordinering mellom ulike deler av helsetjenesten…»



.. fullt av muligheter?

Løser dataalderens tøffeste utfordring

Optique

Søker fremtiden for Microsoft

Musikk + IT = kreativ boom

Verdens største på helsedata

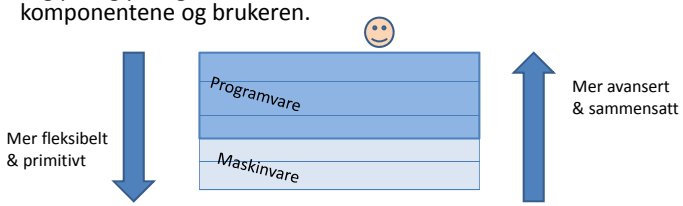
Fra Blindern til Silicon Valley

Aktuelle saker innen forskning

tilfeldig utdrag fra Ifi's forside Forskning

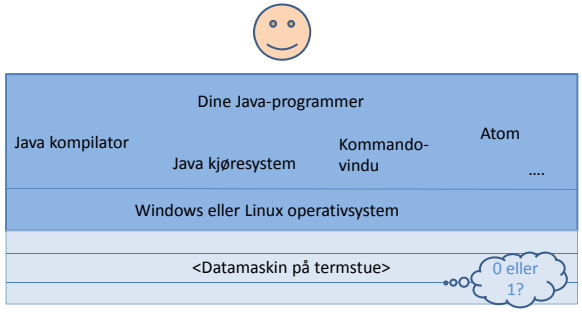
Sentrale konsepter: Lagdeling og grensesnitt

Lag på lag på lag mellom de minste elektroniske komponentene og brukeren.

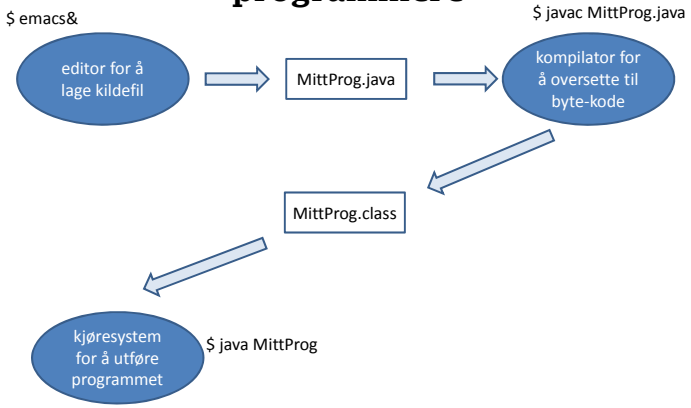


Grensesnitt definerer hvordan programmer kan bruke andre programmer og tilby mer avanserte funksjoner.
Brukergrensesnitt definerer hvordan mennesker kan bruke et program.

«Våre» lag



Programmer vi bruker for å programmere



Hva er programmering?

Å lære en datamaskin å gjøre «noe» på en bestemt måte.
- løse et problem
- overta en repetitiv/ arbeidskrevende oppgave
- utføre noe som ikke lar seg gjøre manuelt

Programmering omfatter alltid de samme elementene:

1. Hva skal gjøres? — Spesifikasjon
2. Hvordan skal maskinen gå frem? — Design
3. Hvilke data trenger den å holde rede på? — Design
4. Hvordan beskrive dette i et språk maskinen forstår? — Java
5. Hvordan være sikker på at maskinen (alltid) gjør det jeg har tenkt (se 1)? — Testing

Dette er *ikke* en sekvensiell prosess – vi lærer underveis og må gå tilbake og i flere runder.

Hva handler det om?

- formulere problemer/ arbeidsoppgaver
- tenke kreativt omkring løsninger
- og formulere løsninger klart og nøyaktig i en form som kan utføres av en datamaskin

Programmering er problemløsning –
og innebærer å leve mye av tiden med "problemer".

Eller: .. med "å skape noe nytt!"

Objektorientert programmering – hva og hvorfor ?

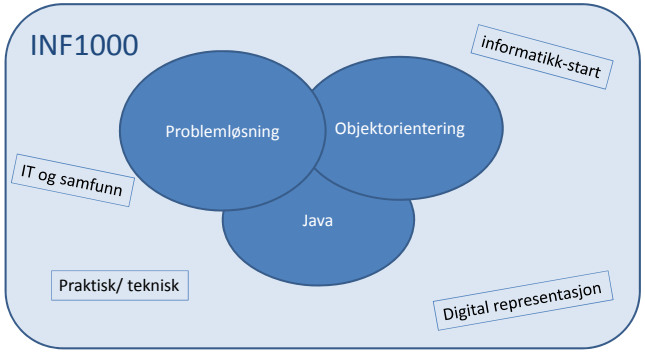
- *Objektorientering* er et tankesett eller *paradigme* som egner seg til å modellere (lage representasjoner av) komplekse problemstillinger
- Spesielt nyttig når deler av et program skal kunne utvikles og vedlikeholdes uavhengig av hverandre, men likevel fungere godt sammen
- OOP støtter programmereren i å *modellere* virkeligheten ut fra de behovene programmet skal dekke

Objektorientert programmering - hvordan?

et lite frempek

- *Modellerer* virkeligheten: Velger ut sentrale begreper/ «ting» og operasjoner knyttet til disse
- Representeres under kjøring av *objekter*, som lagrer informasjon og kan utføre handlinger
- Hvilke handlinger et objekt kan utføre og hvordan, beskrives i *klassen* objektet tilhører. Java-programmer består av en eller flere klasser.
- I INF1000 starter vi med hvordan data representeres og bearbeides i Java, i klasser som vi ikke lager objekter av
- Senere (fra uke 5) skal vi bruke dette i objektorienterte programmer («Late Objects»)

Grunnkurs i objektorientert programmering



Programmeringsspråket Java

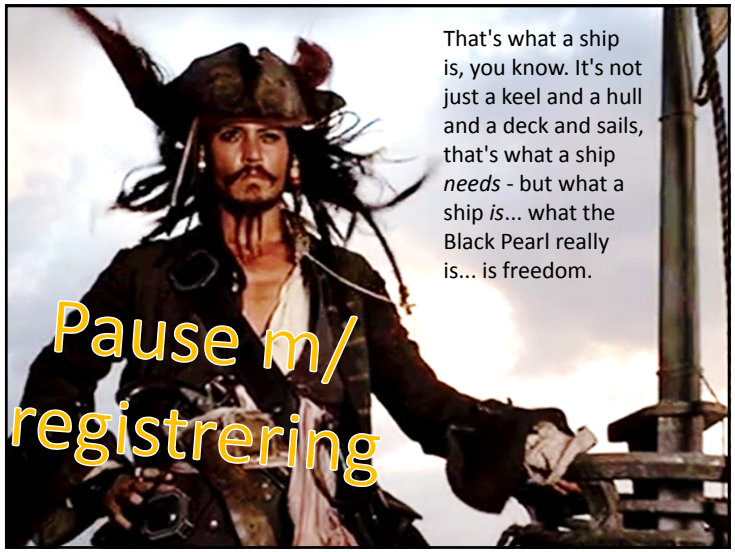
- Mye brukt
- Presist/ sikkert – lar deg ikke gjøre «farlige» feil som utvikler
- Samme programkode kan kjøre på flere maskiner
- Støtter sentrale konsepter for programmerere (OO)
  - Kraftfullt, men kan kjennes omstendelig
  - Undervisning: Nyttig, men krevende i starten
- NB: Java er ikke JavaScript på tross av navnelikhet!

Java – hva er det?

- Programvare og spesifikasjoner for utvikling og kjøring av programmer i programmeringsspråket Java
- Finnes for ulike typer maskinvare og operativsystemer

Java – hva trenger vi?

- *Kjøresystemet* Java Runtime Environment (JRE) støtter kjøring av ferdige programmer
- Java Development Kit (JDK) for *programutvikling* inneholder kompilator i tillegg til JRE
- De (vi!) som utvikler Java programmer trenger JDK
- På Ifi bruker vi Java SE 7 (versjon 7 for typiske desktop og server-maskiner)
- Laptop-hjelpen (se nettsider) kan hjelpe med installasjon



Et (tomt) første program i Java

```
class Uke00 {
  public static void main (String [] args) {

  }
}
```

Et første program – med kommentarer

```
class Uke00 {
  public static void main (String [] args) {

    // Dette programmet gjoer ingenting!

    /* Kommentarer kan vi ogsaa skrive
         på denne maaten, over flere linjer */
  }
}
```

Skrive ut en linje

```
class Uke01 {
  public static void main (String [] args) {

    // Skriver ut en tekst til brukerens skjerm:

    System.out.println ("Velkommen til INF1000!");

  }
}
```

Skrive ut: Med og uten linjeskift

```
class Uke02 {
  public static void main (String [] args) {

    // Skriver ut Velkommen til INF1000!:

    System.out.print ("Velkommen ");
    System.out.print ("til ");
    System.out.println ("INF1000!");

  }
}
```

Tekst-konkatenering

```
class Uke03 {
  public static void main (String [] args) {

    // Konkatenerer (skjoeter) to tekster
    // og skriver ut til skjerm:

    System.out.println ("Velkommen til" + "INF1000!");
  }
}
```

Variabler – for å lagre verdier

```
class Uke04 {
  public static void main (String [] args) {

    // Deklarere en variabel for lagring av et heltall
    int alder;

    // Lagrer et tall i variabelen
    alder = 19;

    // Skriver ut tallet med en forklaring
    System.out.println ("alder har verdien " + alder);
  }
}
```

Endring av variabler

```
class Uke05 {
  public static void main (String [] args) {

    int alder;
    alder = 19;

    // Endrer verdien i alder
    alder = 25;
    System.out.println ("alder har verdien " + alder);
  }
}
```

Feil – uunngåelig & lærerikt

- Fordi datamaskiner er maskiner er de lite tolerante og lite forståelsesfulle (om ikke de er programmert til å virke slik)
- Når vi programmerer vil vi heller ikke at maskinen skal begynne å «gjette» hva vi mener (mer akseptabelt når vi leter etter noe i en søkemotor!)
- Dvs strenge krav til nøyaktighet for at alt funker
- Kompileringsfeil (typisk skriveleifer)
- Kjøretidsfeil (noe går feil underveis)
  - Logiske feil (programmet gjør noe annet enn ventet)
- Tips foreløpig: UNNGÅ NORSKE TEGN HELT

Kodestil - kodekonvensjoner

Tilleggsregler av hensyn til programmerer og leser, som ikke kreves av Java

- Økt lesbarhet og oversikt
- Ryddigere kode, også når flere samarbeider

⇒ Redusert fare for feil, høyere kvalitet

Se Coding guidelines i Big Java, Appendix L

INF1000 semesterplan

Semesterside /Timeplan

INF1000	2015	Mål for uka	Obl frister
Uke 0	34	Introduksjon	
Uke 1	35	Programmering er problemløsning!	
Uke 2	36	Kontrollflyt, feilsøking	1
Uke 3	37	Hvordan løse problemer med programmering? På Sundvollen	2 (søndag 6.9)
Uke 4	38	Hvordan løse problemer som involverer data?	3
Uke 5	39	Hva er objektorientert programmering?	4
Uke 6	40	Hvilken rolle spiller IT (utviklere) i samfunnet?	5
Uke 7	41	Mekanismer og teknikker for utvikling av mer komplekse, objektorienterte programmer.	
Uke 8	42		
Uke 9	43		6
Uke 10	44	Hvordan representeres tall, tekst, bilder med mer i en datamaskin?	
Uke 11	45	Et større program – eksempel.	7
Uke 12	46	Prøveeksamen	
Uke 13	47	Repetisjonskurs med gruppelærere	
Uke 14	48		
Uke 15	49	Eksamen	

«Normal» undervisningsuke

	Tirsdag	Onsdag	Torsdag	Fredag		Mandag
08:15					H	
09:15					E	
10:15	INF1000 ekstra-gruppe				L	
11:15		INF1000-gruppe. 2 t lab		INF1000-gruppe. 2 t seminar		
12:15						
13:15	OBL-FRIST					
14:15	INF 1000 fore-lesning				H	
15:15					E	
16:15					L	
17:15					G	

Forventet arbeidsinnsats: 13-14 timer/ uke/ emne

PROGRAMMERING
- Oblig. oppgaver
- Øvingsoppgaver

Hvordan jobbe med emnet

- kikk på lærebok før forelesning
- forelesning
- flervalgstest
- lærebok for økt forståelse, praktiske tips og detaljer
- før og på lab: Løs ukeoppgaver (Trix) og oblig
- delta aktivt i seminartimene

Lærebok

- Cay S. Horstmann; Big Java Late Objetcs
- Elektroniske versjoner finnes, ikke sjekket ut
  NB: Kun trykte/ skrevne hjelpemidler på eksamen!
- Big Java dekker også INF1010-pensum
- Java for Everyone: samme stoff, men kun INF1000

«Uke 0» (gult)

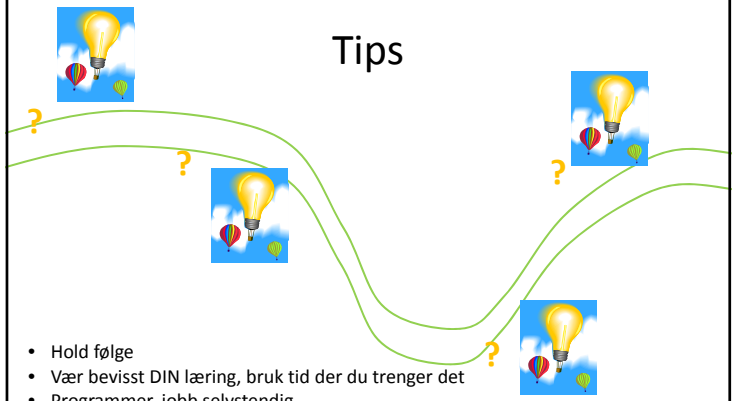
Kalender-uke 34

Mandag 17.8	Tirsdag 18.8	Onsdag 19.8	Torsdag 20.8	Fredag 21.8	Helg
	INF1000 Fore-lesning 2 t	Frivillig lab «Forkurs-trening» 2 t		Første gruppe-timer (seminar)	

Kalender-uke 35

Mandag 24.8	Tirsdag 25.8	Onsdag 26.8	Torsdag 27.8	Fredag 28.8	Helg
Første gruppe-timer (seminar)	INF1000 Fore-lesning 2 t	Gruppe-timer (lab)	Gruppe-timer (lab)	Gruppe-timer (seminar)	

Tips



- Hold følge
- Vær bevisst DIN læring, bruk tid der du trenger det
- Programmer, jobb selvstendig
- Kollokver, diskuter
- Bruk semestersidene for struktur
- Snakk med gruppelærer om evt problemer, i tide!
- Husk at dette er moro ☺ - når du jobber på (litt over) ditt nivå!

Mye info?
Dette bør du ha fått med deg i uke 0

- Skrive inn og kjøre ett Java-program
- Les eller videresend mail til studentkonto
- Følg med på semestersiden for
  - Praktisk informasjon og beskjeder
  - Krav til obliger, innlevering
  - Undervisningsplan
    - Pensum og ressurser
    - Flervalgstester for teori
    - Programmeringsoppgaver for praksis, inkl obliger
- Flervalgstest for uke 0

Neste uke

- Programmering som problemløsning
  - variable, forgreninger og innlesing fra bruker

Lykke til med semesteret og programmeringen!