INF1000 Høst 2015

Uke 11:
Repetisjon og pensumgjennomgang

Innhold

1. time:
   Guidet tur gjennom læreboken (og pensum): Sentrale konsepter og mekanismer i Java

2. time:
   - Hva trenger vi utover å lese pensum?
   - Eksamenstips

Hva skal evalueres? Fra kurssidene

Etter å ha tatt INF1000

- kan du skrive små til middels store programmer oppdelt i klasser.
- har du grunnleggende ferdigheter i objektorientering i Java med klasser, metoder, objekter og pekere.
- kan du lage programskisser med enkle UML klasse- og ~~objekt-diagrammer.~~
- kan du lage løsninger på mindre, virkelige problemer på én maskin med brukerinteraksjon og lagring av data på disk.
- kan du bruke enkle deler av Javas klassebibliotek.
- kan du finne og rette feil i egne programmer.

Overordnet pensum

- Kapittel 1-8 i boken "Big Java: Late Objects" av Cay S. Horstmann (2012-utgave).
- Personopplysningsloven
- I tillegg er obliger og det som er forelest (dvs lysark fra forelesningene) pensum.

Hvorfor har vi en lærebok?

- Læreboken forteller stort sett mer detaljert enn forelesningene.
- Det er en fordel å få ting fortalt på to ulike måter.
- Læreboken er bedre å slå opp i.
- Random fact og Special topic er artig å lese og utdyper faget (selv om de ikke er pensum).
- Common error, Programming tip og Self check er nyttige (men de er heller ikke pensum)

Kapittel 1: Introduction

Innholder først og fremst bakgrunnsinformasjon for det som kommer senere. Nyttig også i senere emner, og antakelig lettere å forstå nå enn da dere startet.

«Programmering er problemløsing» [BJ: 1.7]

Nyttig lærdom: Det første viktige steget i programmering er å omforme problemet til en algoritme, dvs gi det en form som datamaskinen kan løse. Så kan algoritmen skrives i Java

Kapittel 2: Fundamental data types

1. Variabler og tilordninger

2. Numeriske typer og uttrykk; int og double er viktigst.
   Lite vekt på behandling av matematiske uttrykk i INF1000 – men viktig og nyttig for noen av dere senere.

3. I/O (lesing og skriving) med
   – Scanner / hasNextLine / nextLine /Integer.parseInt
   – System.out.println

⇒ Dette anbefaler vi sterkt at dere bruker I/O-notatet til på eksamen

4. String

Kapittel 3: Decisions

Dette kapittelet tar for seg det som har med valg å gjøre:

- if-setninger
- ulike typer sammenligninger
- boolske variabler, operatorer og uttrykk

Dette må dere kunne bruke i programmering; unntaket er flytskjemaer [BJ: 3.5] som ikke er pensum.

Merk at f eks switch ikke er pensum (special topic), lett å bruke feil og betraktes som "risikabelt".

Kapittel 4: Loops

Dette er uunnværlige redskaper i en programmerers verktøykasse. Omtrent alle programmer inneholder en løkke.

1. while-løkker går så lenge en betingelse gjelder

2. for-løkker går et gitt antall ganger

Dette er ikke pensum:
- do-løkker [BJ: 4.4]
- simulering og tilfeldige tall [BJ: 4.9] (selv om dere har sett et eksempel)

Heller ikke break (special topic). Mye feil bruk i eksamen 2014.

Kapittel 5: Methods I

Nå begynner det å bli mer avansert. Alt i kapittelet er sentralt pensum (unntatt rekursive metoder [BJ: 5.9]):

1. [BJ: 5.1] om metoder som «svarte bokser» for lettere å holde oversikten over programmet.
2. [BJ: 5.2 og 5.6] Hva bruker vi metoder til?
   - unngå å gjenta kode (static -metoder)
   - definere grensesnittet til en klasse (se kapittel 8)
3. [BJ: 5.3] Parametre! Noe av det aller viktigste i hele kurset!
4. [BJ: 5.4–5] Returverdier (for de metoder som ikke er void-metoder)

Kapittel 5: Methods II

(forts)

5. [BJ: 5.7] Problemløsing med stegvis forfining beskriver en god teknikk til å la et program bli til litt etter litt.
6. [BJ: 5.8] En variabels skop er delen av et program en variabel er tilgjengelig. Enkelt sagt: En lokal metodevariabel finnes bare når metoden utføres

Kapittel 6: Arrays and array lists I

Arrayer er en veldig mye brukt datastruktur.

1. [BJ: 6.1] introdusere arrayer.
2. [BJ: 6.2] spesialisert (enhanced) for-løkke brukes for ArrayList og HashMap
3. [BJ: 6.3] gir eksempler på bruk av arrayer.
4. [BJ: 6.4] beskriver hvordan man bruker arrayer i metoder.
5. [BJ: 6.5] er et nyttig avsnitt om hvordan man kan løse problemer ved å bruker arrayer.

Kapittel 6: Arrays and array lists II

6. [BJ: 6.6] todimensjonale arrayer er ikke pensum.
7. [BJ: 6.7] tar for seg klassen ArrayList som man kan bruke når man ikke vet arrayens størrelsen på forhånd.

Kapittel 7: Input/output and exception handling

Dette kapittelet tar opp mye mer om lesing og skriving enn vi trenger i INF1000, så det lille heftet om I/O du finner på nettsiden, er vårt pensum isteden.

Unntakshåndtering er ikke pensum.

Kapittel 8: Objects and classes

Dette er det viktigste kapittelet i INF1000; => *Grunnkurs i objektorientert programmering*.

Hele kapittelet er pensum.

Forelesningene viser vår vektlegging og hva vi forventer at dere kan vise på eksamen (sammen med obliger og prøveeksamen).

- Innkapsling (tilgang til objektenes data gjennom grensesnitt)
- Design og bruk av strukturer med flere objekter av flere klasser

Objektorientert programmering: Fremgangsmåte

1. Finn ut hvilke klasser vi trenger og hvordan de skal henge sammen; her er UML-diagram nyttig. **Husk**: En klasse skal representere noe spesifikt, enten et konsept eller en fysisk gjenstand.
2. Definer grensesnittet (dvs metodene).
3. Finn representasjonen (dvs objektvariablene).
4. Programmer grensesnittmetodene

Annet forelest pensumstoff

UML klassediagrammer [BJ: app K]

UML klassediagrammer brukes under planleggingen av programmeringen og viser klassene (med representasjon og grensesnitt) og hvordan de forholder seg til andre klasser. Tidlig i design-fasen eller for overblikk kan det være nyttig å prøve seg frem kun med klassenavn og forhold mellom disse.

Relasjonsangivelsen (navnet og antallet) gir et grunnlag for å definere mye av representasjonen.

HashMap (oppslagstabeller)

- Viktig!
- Behandles i [BJ: 15.4] – men her er forelesningsnotatene enklere å forholde seg til (uke 8 og 9)
- En HashMap er som en ArrayList der indeksen er en String i stedet for et heltall, og der de lagrede elementene ikke har noen rekkefølge.

Hvordan organisere mengder av objekter? Noen ledetråder

- Skal du lagre et (kjent) antall verdier av en primitiv type (int, boolean, char,..)
  - array
- Er elementene naturlig, løpende nummerert?
  - array eller ArrayList
- Skal du lagre et ukjent/ varierende antall objekter?
  - ArrayList eller Hashmap
- Skal du lagre String- eller andre objekter som det er naturlig å identifisere med en tekst og ikke et nummer?
  - HashMap

Digital representasjon

1. Tegnsett [BJ: RF 2.2, app A] (ikke viktig del av pensum)

Det finnes mange tegnsett; de mest brukte er ISO Latin-1 (= ISO 8859-1) og Unicode; den siste kodes gjerne som UTF-8.

2. Tall [BJ: app I]

Dere bør kunne konvertere små tall <= 64 fra desimalt til binært og heksadesimalt og omvendt; ellers ikke viktig i dette kurset

3. Digital koding av lyd og bilder er ikke pensum i INF1000

Bruk forelesningsnotatene til Dag!

IT og samfunn, personvern

Dette er viktig i pensum - se presentasjonen uke 6

- Berører Personopplysningsloven (Poppl) det jeg ønsker å gjøre? (se figur fra forelesning)
- Definisjoner, krav og unntak – se Poppl

Se også lenke til notat om IT og samfunn, alternativt Digitale medier (Gisle M fl). Dette er almenkunnskap som dere vil ha mye glede av utover i studiet – og ellers.

INF1000 Eksamensforberedelser og -tips

Høst 2015
Siri Moe Jensen

Undervisning mot eksamen

Uke	tirsdag	onsdag	torsdag	fredag	mandag
11: 3.-9.11	forelesning	Grupper: Lab		Grupper: Seminar	
12: 10.-16.11	prøve-eksamen	Grupper: Lab		Grupper: Seminar (siste gruppetimer!)	
13: 17.-23.11		Repetisjons-kurs	Repetisjons-kurs	Repetisjons-kurs	Repetisjons-kurs
14: 24.11-30.11	Repetisjons-kurs	Repetisjons-kurs	Repetisjons-kurs	Repetisjons-kurs	
4. desember				Eksamen (14:30)	

Følg med på kurssidene og les eller videresend UiO-mail!

Prøveeksamen 10. november

- Oppmøte i Smalltalk kl 10:00
- Utdeling av prøveeksamen med svar på praktiske spørsmål (på semestersiden fra kl 8:00 for de som har forelesninger)

< 10:15 – 14:15 Løse prøveeksamen >

- Enkel servering ca 14 i/ ved Simula
- Gjennomgang prøveeksamen i Simula 14:15-16

Før eksamen I

- Trening i større programmer med komplekse strukturer, overblikk/ helhet:
  - Oblig 7 (kan den forbedres?)
  - Eksamensoppgaver (legges ut denne uken og neste)
- Enkeltdeler av pensum
  - Forelesninger, slå opp/ utdyp fra lærebok
  - Løs oppgaver fra Trix og evt lærebok, uløste obligoppg?
  - Repetisjonskurs
- Vektlegging og fremgangsmåter i kurset er noe endret, spesielt fra 2014 (obs gamle eks.oppgaver)

Før eksamen II

- Gjennomfør prøveksamen!
  - Planlegg hjelpemidler
  - Sett av 4 timer
  - Ikke bruk maskin
  - lag en realistisk eksamenssituasjon!
  - revurder evt hjelpemidler til eksamen
- Les eksamensreglementet, sjekk grundig hvor/ når du skal møte, beregn god tid
- Følg med på kurssidene og les eller videresend UiO-mail

På eksamen

- Besvarelsen leses av sensorer = mennesker. Skal ikke kompileres eller kjøres av maskiner
- Foreleser(e) vil gå rundt etter ca 0.5-1.5 time, forbered spørsmål om uklarheter
- Les oppgaven grundig! Hva ber den (ikke) om?
- Poengsum sier noe om hvor mye du (maks) tjener på å løse oppgaven – vurder hvor mye tid det er verdt å bruke på oppgaver der du står fast
- Alle skriftlige hjelpemidler tillatt (dvs alt av papir, ikke noe elektronikk)

1. De aller fleste får liten tid

- Ikke kladd (alt) – skriv så du kan levere direkte. Men tegn/ skisser gjerne ved siden av.
- Husk gjennomslag – ikke rettelakk eller viskelær!!
- Sensor tåler overstrykninger og er velvillig (men det må være lesbart, også på gjennomslag)
- Bruk gjerne forkortelser, for eksempel **System.out.println -> s.o.p** eller **public static void -> p.s.v** (lag gjerne en liste på første ark om du bruker flere)

2. Sensor vil deg vel!

- Hensikten er å se hva du har lært av læringsmålene – ikke å pirke på språk, semikolon, eller innrykk (men leselighet er svært nyttig for deg og oss)
- Vi leter etter hva du kan – men du må vise oss det (og det må svare på det oppgaven spør om)
- Har du ikke tid til å programmere i detalj er kort beskrivelse bedre enn ingenting – men den må vise noen relevante tanker om hvordan du ville gått frem.
- Vi trenger ikke kommentarer om det ikke er noe spesielt du vil ha frem