MAT1030 – Diskret Matematikk

Forelesning 1: Algoritmer, pseudokoder, kontrollstrukturer

Dag Normann

Matematisk Institutt, Universitetet i Oslo

19. januar 2010

(Sist oppdatert: 2010-01-19 14:12)

Velkommen til MAT1030!

Introduksjon

- Velkommen til MAT1030: Diskret matematikk!
- Plan for i dag:
  - Mest praktiske opplysninger
  - En oversikt over kurset
  - Litt fra kapittel 1 i læreboken
- Før vi begynner
  - Det er lov å stille spørsmål underveis.
  - Alle forelesningene vil bli lagt ut på kursets hjemmeside.
  - Vi begynner kvart over, og ikke senere.
  - "Spørsmål eller kommentarer?"

Undervisning

- **Foreleser**: Dag Normann (`dnormann@math.uio.no`)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Knut Berg (`knube@student.matnat.uio.no`)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Knut vil basere seg på regning på tavlen.
- **Åpne grupper/orakler**:
- Mandag - fredag
- 10.15 - 14.00, Auditorium 3, Vilhelm Bjerknes hus.
  - Se kursets hjemmeside for mer informasjon.
  - Prøv å løs ukeoppgavene selv.
  - Bruk plenumsregningne og de åpne gruppene!

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 3. mars: Innlevering av obligatorisk oppgave 1.
  - Fredag 7. mai: Innlevering av obligatorisk oppgave 2.
- Oppgavene vil bli lagt ut senest 14 dager før.
- Bedømmes til bestått/ikke bestått.
- Begge oppgavesettene må bestås for å kunne gå opp til eksamen.
- Skriftlig eksamen, 3 timer, uten hjelpemidler.
  - Mandag 7. juni kl. 14.30.

Evaluering

- Alle emner i matematikk skal evalueres av studentene.
- Vi skal velge to til fire kontaktpersoner som sammen med foreleser skal bestemme hvordan evalueringen skal foregå.
- Vi kommer tilbake til valg av kontaktpersoner neste uke.

Pensum og lærebok

Peter Grossman
Discrete Mathematics for Computing (3. utgave)
Grassroots Series, Palgrave Macmillian (2009)
ISBN: 13:978-0230-21611-2

Det vil være mulig å bruke utgave 2.

Pensum er kapittel 1–7 og 9 og utdrag fra kapittel 10, 11 og 13, samt alle forelesningsnotater og øvingsoppgaver som legges ut på kurset hjemmeside.

Forelesningsnotatene vil inneholde lærestoff som ikke står i boka, **og det er også pensum.**

Hva er diskret matematikk?

- Diskret matematikk er et samlebegrep for matematikk hvor kontinuitet, geometri eller algebra ikke spiller noen stor rolle.
- Diskret matematikk er matematikken tilpasset en digital verden, mens mye annen matematikk er tilpasset en analog verden.
- I MAT1030 skal vi ta for oss temaer som er relevante for en grunnleggende forståelse av bruk av, og virkemåte til, datamaskiner.

Emnebeskrivelsen

Tallsystemer, utsagnslogikk med sannhetsverditabeller, litt om kvantorer og utforming av bevis, elementær mengde- , relasjons- og funksjonslære, induktivt definerte strukturer med generelle rekursive konstruksjoner og induksjonsbevis, litt kombinatorikk, grafer og trær og til sist litt om kompleksitet av algoritmer, heri bruk av O-notasjonen. I emnet legges det vekt på utformingen av algoritmer i tilknytning til stoffet.

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall
- Logikk
  - Relevans for informatikk
  - Innføring i utsagnslogikk
  - Litt om bevisteknikker
- Mengdelære
  - Grunnleggende begreper
  - Relasjoner
  - Funksjoner
- Induksjon og rekursjon
- Kombinatorikk
- Grafteori med anvendelser
- Trær
- Kompleksitet av algoritmer

Hva legger vi vekt på?

- Hvilken relevans har dette stoffet for studier i informatikk?
- Hvordan kan vi finne algoritmer for å løse de problemene som presenterer seg?
- Vi kommer til å arbeide med algoritmer i tilknytning til logikk i større grad enn det boka gjør.
- Undervisningsmaterialet for dette kommer som en del av kompendiet basert på forelesningene.
- Det finnes lenker til de tilsvarende kompendiene fra 2008 og 2009 på semestersidene for MAT1030.
- Pensum vil være basert på årets kompendium.

Kapittel 1: Algoritmer

Algoritmer

En algoritme er en oppskrift som forteller oss hvordan vi skritt for skritt skal kunne oppnå et resultat eller løse et problem. Eksempler på algoritmer kan være:

- Kakeoppskrifter.
- Automatisk innsjekking på fly.
- Beskrivelsen av hvordan man utfører divisjon mellom flersifrede tall.
- Oppskrift på hvordan man løser opp parenteser og trekker sammen flerleddede uttrykk i algebra.

Algoritmer

Hva er det som kjennetegner en algoritme?

Det skal ikke kreves intelligens eller forståelse for å følge den.

- Du skal ikke kunne kjemi for å bake en kake.
- Du skal kunne sjekke inn på fly selv om du har teknologifobi.
- Det er ikke nødvendig å forstå hva man gjør når man utfører en divisjon, regnetrening er det som trengs.
- Mange lærer seg hvordan de kan løse oppgaver i skolealgebra, uten å ha peiling på hva de egentlig driver med.

Algoritmer

Vi skal fokusere på algoritmer som

- beregner funksjoner
- avgjør om et objekt eller en datamengde har en gitt egenskap eller ikke
- organiserer gitte data på en ønsket måte (eksempelvis ordner dataene)
- utfører andre oppgaver i tilknytning til matematikk eller informatikk som vi ønsker å kunne få utført.

Algoritmer

Hvem ønsker vi å kommunisere algoritmen til, og hvordan skal det gjøres?

1. Kakebakere, flypassasjerer og liknende.
2. Skolebarn/ungdom og studenter som skal lære matematikk.
3. Teknisk kyndige medmennesker som skal “forstå” algoritmen.
4. Datamaskiner som skal utføre algoritmen for oss.

I MAT1030 er gruppe 3 den mest aktuelle.

Pseudokoder

- En pseudokode er en måte å beskrive en algoritme på.
- Pseudokoden beskriver hvordan algoritmen kan følges trinn for trinn.
- En pseudokode formuleres i et språk som er mer teknisk enn naturlige språk og mindre teknisk enn programmeringsspråk.
- Vi skal bruke pseudokoder på samme måte som i læreboka.
- Man må se eksempler, og øve, for å bli flink til å skrive pseudokoder.

Pseudokoder

Eksempel (Areal av trekant)

1. Input h [h er høyden i trekanten.]
2. Input g [g er lengden på grunnlinjen i trekanten.]
3. $areal \leftarrow \frac{h \cdot g}{2}$
4. Output $areal$

Pseudokoder

Vi kunne ha skrevet en annen pseudokode for å beregne det samme:

Eksempel

1. Input h
2. Input g
3. $areal \leftarrow h \cdot g$
4. $areal \leftarrow \frac{areal}{2}$
5. Output $areal$

- Et viktig aspekt ved pseudokoder er bruk av variabler
- Variabler er noe som kan gis forskjellige verdier.

Pseudokoder

Så langt består en pseudokode av en nummerert liste instruksjoner hvor hver instruksjon har et av følgende tre formater:

- Gi en input-verdi til en variabel.
- Gi en variabel en ny verdi, som en funksjon av de eksisterende verdiene på variablene.
- Gi verdien til en av variablene som output, det vil si resultatet av algoritmen.

Vi kan bruke hva vi vil som variable, eksempelvis er h, g og $areal$ variablene i pseudokodene vi har sett på.

Pseudokoder

Hvis vi skal beregne verdien av en formel for areal, volum, hastighet etter en viss tids fritt fall og liknende, kan vi bruke pseudokoder slik vi har sett dem til nå. Det finnes imidlertid algoritmer, og tilhørende kontrollstrukturer, for å beregne

- $|x|$ fra x
- $n!$ fra n
- ledd nummer n i Fibonacci-følgen

$$1, 1, 2, 3, 5, 8, 13, 21, \ldots$$

og for å undersøke om

- parentesene i et algebraisk uttrykk er satt på lovlig måte
- et naturlig tall er et primtall.

Kontrollstrukturer

- En kontrollstruktur brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.
- Vi skal innføre de kontrollstrukturene det er aktuelt å bruke i dette emnet via eksempler på bruk. Disse eksemplene skal supplere eksemplene fra læreboka.
- Vi skal benytte oss av de samme kontrollstrukturene som boka.

Kontrollstrukturer

Eksempel (Absoluttverdi)

Vi skal gi en pseudokode for å beregne absoluttverdien til et tall x:

1. Input x
2. **If** $x < 0$ **then**
   2.1. $x \leftarrow -x$
3. Output x

Kontrollstrukturer

Eksempel (Avstand)

Vi skal gi en pseudokode for å beregne avstanden mellom to heltall.

1. Input n [n et heltall]
2. Input m [m et heltall]
3. **If** $n < m$ **then**
   3.1. $x \leftarrow m - n$
   else
   3.2. $x \leftarrow n - m$
4. Output x

Kontrollstrukturer

Vi skal se tre eksempler på hvordan vi kan skrive pseudokoder for algoritmer som beregner

$$n! = 1 \cdot 2 \cdot \ldots \cdot n.$$

I det ene eksemplet bruker vi en while-løkke, i det andre en repeat-until-løkke og i det siste en for-løkke.

Kontrollstrukturer

Eksempel (while-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$
3. $i \leftarrow 1$
4. **While** $i \leqslant n$ **do**
   4.1. $x \leftarrow x \cdot i$
   4.2. $i \leftarrow i + 1$
5. Output x

Kontrollstrukturer

Eksempel (repeat-until-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$
3. $i \leftarrow 1$
4. **Repeat**
   4.1. $x \leftarrow x \cdot i$
   4.2. $i \leftarrow i + 1$
5. **until** $i = n + 1$
6. Output x

Kontrollstrukturer

Eksempel (for-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$
3. **For** $i = 1$ **to** n **do**
   3.1. $x \leftarrow x \cdot i$
4. Output x

Kontrollstrukturer

Bytte av verdi på variablene, hvordan vi ikke skal gjøre det og hvordan vi skal gjøre det.

Eksempel (Feil måte)

1. $y \leftarrow x$
2. $x \leftarrow y$

Eksempel (Riktig måte)

1. $hjelp \leftarrow x$
2. $x \leftarrow y$
3. $y \leftarrow hjelp$