MAT1030 – Diskret Matematikk

Forelesning 1: Algoritmer, pseudokoder, kontrollstrukturer

Roger Antonsen

Institutt for informatikk, Universitetet i Oslo

13. januar 2009

(Sist oppdatert: 2009-01-14 16:45)

Velkommen til MAT1030!

Introduksjon

Introduksjon

- Velkommen til MAT1030: Diskret matematikk!

Introduksjon

- Velkommen til MAT1030: Diskret matematikk!
- Plan for i dag:

Introduksjon

- Velkommen til MAT1030: Diskret matematikk!
- Plan for i dag:
  - Mest praktiske opplysninger

Introduksjon

- Velkommen til MAT1030: Diskret matematikk!
- Plan for i dag:
  - Mest praktiske opplysninger
  - En oversikt over kurset

Introduksjon

- Velkommen til MAT1030: Diskret matematikk!
- Plan for i dag:
  - Mest praktiske opplysninger
  - En oversikt over kurset
  - Litt fra kapittel 1 i læreboken

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

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.

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.

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.

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.
  - “Spørsmål eller kommentarer?”

Undervisning

Undervisning

- **Foreleser**: Roger Antonsen (`rantonse@ifi.uio.no`)

Undervisning

- **Foreleser**: Roger Antonsen (`rantonse@ifi.uio.no`)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus

Undervisning

- **Foreleser**: Roger Antonsen (`rantonse@ifi.uio.no`)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus

Undervisning

- **Foreleser**: Roger Antonsen (`rantonse@ifi.uio.no`)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (`georgba@math.uio.no`)

Undervisning

- **Foreleser**: Roger Antonsen (rantonse@ifi.uio.no)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (georgba@math.uio.no)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus

Undervisning

- **Foreleser**: Roger Antonsen (rantonse@ifi.uio.no)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (georgba@math.uio.no)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- **Åpne grupper**: Mathias Barra (georgba@math.uio.no)

Undervisning

- **Foreleser**: Roger Antonsen (rantonse@ifi.uio.no)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (georgba@math.uio.no)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- **Åpne grupper**: Mathias Barra (georgba@math.uio.no)
- Tirsdag 10:15–12:00, Seminarrom C317, Vilhelm Bjerknes hus

Undervisning

- **Foreleser**: Roger Antonsen (rantonse@ifi.uio.no)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (georgba@math.uio.no)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- **Åpne grupper**: Mathias Barra (georgba@math.uio.no)
- Tirsdag 10:15–12:00, Seminarrom C317, Vilhelm Bjerknes hus
- Tirsdag 14:15–16:00, Seminarrom C317, Vilhelm Bjerknes hus

Undervisning

- **Foreleser**: Roger Antonsen (`rantonse@ifi.uio.no`)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (`georgba@math.uio.no`)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- **Åpne grupper**: Mathias Barra (`georgba@math.uio.no`)
- Tirsdag 10:15–12:00, Seminarrom C317, Vilhelm Bjerknes hus
- Tirsdag 14:15–16:00, Seminarrom C317, Vilhelm Bjerknes hus
  - Se kursets hjemmeside for mer informasjon.

Undervisning

- **Foreleser**: Roger Antonsen (`rantonse@ifi.uio.no`)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (`georgba@math.uio.no`)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- **Åpne grupper**: Mathias Barra (`georgba@math.uio.no`)
- Tirsdag 10:15–12:00, Seminarrom C317, Vilhelm Bjerknes hus
- Tirsdag 14:15–16:00, Seminarrom C317, Vilhelm Bjerknes hus
  - Se kursets hjemmeside for mer informasjon.
  - Bruk plenumsregningne og gruppetimene!

Undervisning

- **Foreleser**: Roger Antonsen (`rantonse@ifi.uio.no`)
- Tirsdag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- Onsdag 10:15–12:00, Auditorium 1, Vilhelm Bjerknes hus
- **Plenumsregning**: Mathias Barra (`georgba@math.uio.no`)
- Fredag 12:15–14:00, Auditorium 1, Vilhelm Bjerknes hus
- **Åpne grupper**: Mathias Barra (`georgba@math.uio.no`)
- Tirsdag 10:15–12:00, Seminarrom C317, Vilhelm Bjerknes hus
- Tirsdag 14:15–16:00, Seminarrom C317, Vilhelm Bjerknes hus
  - Se kursets hjemmeside for mer informasjon.
  - Bruk plenumsregningne og gruppetimene!
  - Veldig viktig for læringen.

Obligatoriske oppgaver og eksamen

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 27. februar: Innlevering av obligatorisk oppgave 1

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 27. februar: Innlevering av obligatorisk oppgave 1
  - Fredag 24. april: Innlevering av obligatorisk oppgave 2

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 27. februar: Innlevering av obligatorisk oppgave 1
  - Fredag 24. april: Innlevering av obligatorisk oppgave 2
- Oppgavene vil bli lagt ut senest 14 dager før.

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 27. februar: Innlevering av obligatorisk oppgave 1
  - Fredag 24. april: Innlevering av obligatorisk oppgave 2
- Oppgavene vil bli lagt ut senest 14 dager før.
- Bedømmes til bestått/ikke bestått.

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 27. februar: Innlevering av obligatorisk oppgave 1
  - Fredag 24. april: Innlevering av obligatorisk oppgave 2
- Oppgavene vil bli lagt ut senest 14 dager før.
- Bedømmes til bestått/ikke bestått.
- Alle obligene må bestås for å kunne gå opp til eksamen.

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 27. februar: Innlevering av obligatorisk oppgave 1
  - Fredag 24. april: Innlevering av obligatorisk oppgave 2
- Oppgavene vil bli lagt ut senest 14 dager før.
- Bedømmes til bestått/ikke bestått.
- Alle obligene må bestås for å kunne gå opp til eksamen.
- Skriftlig eksamen, 3 timer, uten hjelpemidler.

Obligatoriske oppgaver og eksamen

- Kurset har to obligatoriske oppgaver.
  - Fredag 27. februar: Innlevering av obligatorisk oppgave 1
  - Fredag 24. april: Innlevering av obligatorisk oppgave 2
- Oppgavene vil bli lagt ut senest 14 dager før.
- Bedømmes til bestått/ikke bestått.
- Alle obligene må bestås for å kunne gå opp til eksamen.
- Skriftlig eksamen, 3 timer, uten hjelpemidler.
  - Tirsdag 9. juni kl. 09:00

Pensum og lærebok

Pensum og lærebok

Peter Grossman

Discrete Mathematics for Computing (2. utgave)

Grassroots Series, Palgrave Macmillian (2002)

ISBN: 0-333-98111-1

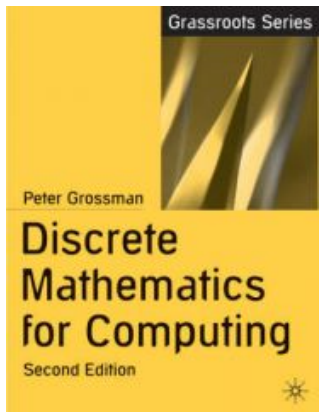
Pensum og lærebok

Peter Grossman

Discrete Mathematics for Computing (2. utgave)

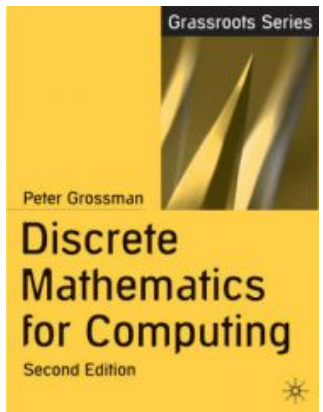
Grassroots Series, Palgrave Macmillian (2002)

ISBN: 0-333-98111-1



Pensum og lærebok

Peter Grossman
Discrete Mathematics for Computing (2. utgave)
Grassroots Series, Palgrave Macmillian (2002)
ISBN: 0-333-98111-1



- 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.

Hva er diskret matematikk?

Hva er diskret matematikk?

- Diskret matematikk er et samlebegrep for matematikk hvor kontinuitet, geometri eller algebra ikke spiller noen stor rolle.

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.

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

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?

Hva er innholdet i MAT1030?

- Algoritmer

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall
- Logikk

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall
- Logikk
  - Relevans for informatikk

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall
- Logikk
  - Relevans for informatikk
  - Innføring i utsagnslogikk

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall
- Logikk
  - Relevans for informatikk
  - Innføring i utsagnslogikk
  - Litt om bevisteknikker

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall
- Logikk
  - Relevans for informatikk
  - Innføring i utsagnslogikk
  - Litt om bevisteknikker
- Mengdelære

Hva er innholdet i MAT1030?

- Algoritmer
- Litt om representasjon av tall
- Logikk
  - Relevans for informatikk
  - Innføring i utsagnslogikk
  - Litt om bevisteknikker
- Mengdelære
  - Grunnleggende begreper

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

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

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

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

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

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

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?

Hva legger vi vekt på?

- Hvilken relevans har dette stoffet for studier i informatikk?

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?

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, og vi vil utvikle en del ekstra undervisningsmateriale underveis.

Kapittel 1: Algoritmer

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:

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.

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.

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.

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

Algoritmer

Hva er det som kjennetegner en algoritme?

Algoritmer

Hva er det som kjennetegner en algoritme?

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

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.

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.

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.

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

Algoritmer

Vi skal fokusere på algoritmer som

Algoritmer

Vi skal fokusere på algoritmer som

- beregner funksjoner

Algoritmer

Vi skal fokusere på algoritmer som

- beregner funksjoner
- avgjør om et objekt eller en datamengde har en gitt egenskap eller ikke

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)

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

Algoritmer

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

Algoritmer

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

1. Kakebakere, flypassasjerer og liknende.

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.

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.

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.

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

Pseudokoder

- En pseudokode er en måte å beskrive en algoritme på.

Pseudokoder

- En pseudokode er en måte å beskrive en algoritme på.
- Pseudokoden beskriver hvordan algoritmen kan følges trinn for trinn.

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.

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.

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

Pseudokoder

Eksempel (Areal av trekant)

Pseudokoder

Eksempel (Areal av trekant)

1. Input h [h er høyden i trekanten.]

Pseudokoder

Eksempel (Areal av trekant)

1. Input h [h er høyden i trekanten.]
2. Input g [g er lengden på grunnlinjen i trekanten.]

Pseudokoder

Eksempel (Areal av trekant)

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

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

Pseudokoder

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

Pseudokoder

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

Eksempel

Pseudokoder

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

Eksempel

1. Input h

Pseudokoder

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

Eksempel

1. Input h
2. Input g

Pseudokoder

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

Eksempel

1. Input h
2. Input g
3. $areal \leftarrow h \cdot g$

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. $areal \leftarrow \frac{areal}{2}$

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 $areal$

Pseudokoder

Pseudokoder

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

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.

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.

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.

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

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.

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

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

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

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

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

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

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

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

Kontrollstrukturer

- En kontrollstruktur brukes for å styre hvordan, og hvorvidt, de enkle instruksjonene i en pseudokode skal utføres.

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.

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

Kontrollstrukturer

Eksempel (Absoluttverdi)

Kontrollstrukturer

Eksempel (Absoluttverdi)

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

Kontrollstrukturer

Eksempel (Absoluttverdi)

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

1. Input x

Kontrollstrukturer

Eksempel (Absoluttverdi)

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

1. Input x
2. **If** $x < 0$ **then**

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. $x \leftarrow -x$

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

Kontrollstrukturer

Eksempel (Avstand)

Kontrollstrukturer

Eksempel (Avstand)

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

Kontrollstrukturer

Eksempel (Avstand)

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

1. Input n [n et heltall]

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]

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**

   else

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

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. $x \leftarrow n - m$

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

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

Kontrollstrukturer

Eksempel (while-løkke)

Kontrollstrukturer

Eksempel (while-løkke)

1. Input n [$n \geqslant 1$, n heltall]

Kontrollstrukturer

Eksempel (while-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$

Kontrollstrukturer

Eksempel (while-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$
3. $i \leftarrow 1$

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**

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. $x \leftarrow x \cdot i$

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. $i \leftarrow i + 1$

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

Kontrollstrukturer

Eksempel (repeat-until-løkke)

Kontrollstrukturer

Eksempel (repeat-until-løkke)

1. Input n [$n \geqslant 1$, n heltall]

Kontrollstrukturer

Eksempel (repeat-until-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$

Kontrollstrukturer

Eksempel (repeat-until-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$
3. $i \leftarrow 1$

Kontrollstrukturer

Eksempel (repeat-until-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$
3. $i \leftarrow 1$
4. **Repeat**

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. $x \leftarrow x \cdot i$

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. $i \leftarrow i + 1$

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** $i = n + 1$

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

Kontrollstrukturer

Eksempel (for-løkke)

Kontrollstrukturer

Eksempel (for-løkke)

1. Input n [$n \geqslant 1$, n heltall]

Kontrollstrukturer

Eksempel (for-løkke)

1. Input n [$n \geqslant 1$, n heltall]
2. $x \leftarrow 1$

Kontrollstrukturer

Eksempel (for-løkke)

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

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. $x \leftarrow x \cdot i$

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

Kontrollstrukturer

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

Kontrollstrukturer

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

Eksempel (Feil måte)

Kontrollstrukturer

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

Eksempel (Feil måte)

1. $y \leftarrow x$

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. $x \leftarrow y$

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)

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. $\mathtt{hjelp} \leftarrow x$

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. $\mathtt{hjelp} \leftarrow x$
2. $x \leftarrow y$

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. $\mathtt{hjelp} \leftarrow x$
2. $x \leftarrow y$
3. $y \leftarrow \mathtt{hjelp}$