Velkommen til TDT4102
Prosedyre- og objektorientert programmering, våren 2014!

Trond Aalberg
Trond.Aalberg@idi.ntnu.no
Kontor 213, IT-bygget

Dagens forelesning

- Om faget
- Praktisk informasjon
- Læringsmål for i dag
  - Bli kjent med faget
  - Litt om C++, programmering, og kompilering
  - Komme i gang med egen programmering
  - Bruk av verktøy

Programmering og programmeringsspråk

- I dette faget skal du lære å programmere
  - forteller datamaskinen hva du vil den skal gjøre for deg
  - lage programmer som består av instruksjoner
- Programmeringsspråket du skal lære er C++
  - et standardisert og mye benyttet programmeringsspråk
  - kan brukes på mange plattformer
  - representativt for en familie av høynivå programmeringsspråk
  - (mye felles med f.eks. Java og C#) – imperative språk

Læringsutbytte

- Kunnskap
  - Syntaks og konstruksjoner i programmeringsspråket og hvordan dette brukes til å spesifisere hvordan programmet skal oppføre seg
  - Kjennskap til anvendelsesområder, begrensinger og underliggende teori
- Ferdigheter
  - Beherske programmering som metode og ferdighet
  - Kan løse enkle og mer komplekse programmeringsoppgaver
  - Kan skrive kode som er oversiktlig, ryddig og feilfri
  - Kan videreutvikle deg som programmerer og raskt lære seg andre programmeringsspråk
- I tillegg skal du ha kompetanse i
  - Bruk av programmeringsverktøy
  - Å finne løsninger på problemer på en systematisk og iterativ måte.

NTNU

Faget bygger på

- Emnet Informasjonsteknologi, grunnkurs (TDT4105 eller TDT4110), eller tilsvarende.
- Vi forutsetter at studentene har noe kunnskap om en datamaskins oppbygging
- Og har noe erfaring i programmering
  - tilsvarende det du lærte i TDT4105/TDT4110
  - men vi starter med det helt grunnleggende og gir et "komplett" innføringskurs i programmering
- Det er stor variasjon i den enkeltes utgangspunkt!
  - og undervisningen er tilpasset dette

5

NTNU

Prosedural- og objektorientert programmering

- I første omgang fokuserer vi på prosedural programmering
  - løse problemer ved å tenke sekvensielt og bruke funksjoner
  - en relativt enkel løsningsmetodikk som gir deg mulighet til lære grunnleggende koding og basisen i programmering
- Seinere i semesteret skal vi se på objektorientert programmering
  - organisere koden ved hjelp av klasser
  - løse problemer ved å tenke samhandling mellom objekter
  - det samme språket, men med noen tillegg
- Forskjellen ligger i hvordan større programmer bygges opp av mindre deler

6

NTNU

Faglig opplegg

- Forelesinger og øvinger er grunnstammen i faget
- Forelesingene benyttes til å gi introduksjon til teorien
- Øvingene brukes for at dere skal lære selv og innarbeide ferdigheter og forståelse

7

NTNU

Programmering krever både kunnskap og ferdigheter!

- Kunnskap og forståelse opparbeides over tid.
- Ferdigheter i programmering bygges med erfaring og innsikt.
- Den som har brei og solid erfaring gjør det bestandig bra på eksamen!

8

NTNU

Studieteknikk

- Kombiner teori og utprøving i praksis
- Ikke forvent at du skal kunne lage og forstå komplisert kode uten videre.
- Bygg opp koden din **del for del** og undersøk og **test ut** hvordan den oppfører seg
- Vær **nysgjerrig** og **utprøvende**!
- Bruk forelesingene, boka og forelesingsnotatene til å få svar.

NTNU

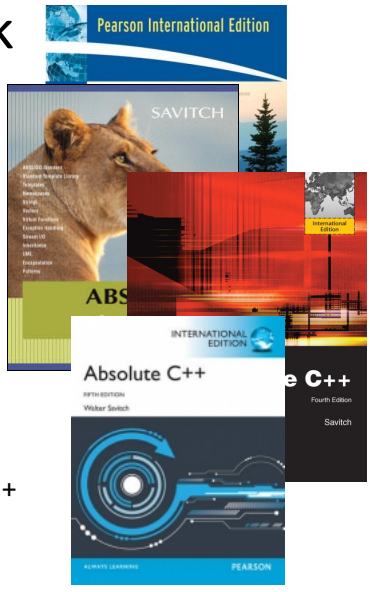
Lærebok

Walter Savitch, Absolute C++,
5th ed., ISBN: 0-273-76932-4

Hele boken er pensum!
Forøvrig er alt forelesnings- og øvingsmateriale (tekster, lysark og løsningsforslag) pensum.

Merk at det finnes mange andre bøker som dekker samme stoff og som dere gjerne må benytte i tillegg.

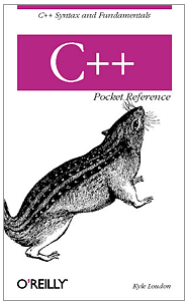
Det finnes også mye nyttig stoff om C++ på web!



NTNU

Tillegg/alternativ litteratur

- Hvis du ønsker en liten bok som kun beskriver C++ syntaksen er denne et alternativ:



- Kyle Loudon, C++ Pocket Reference, O'Reilly 2003, ISBN 0-596-00496-6

NTNU

Tillatt hjelpemiddel på eksamen

- Hjelpemiddel kode: C
- Som betyr: "Spesifiserte trykte og håndskrevne hjelpemidler tillatt. Bestemt, enkel kalkulator tillatt."
- Vi tillater at du har med Absolute C++ eller C++ Pocket Reference
  - Men ikke begge!

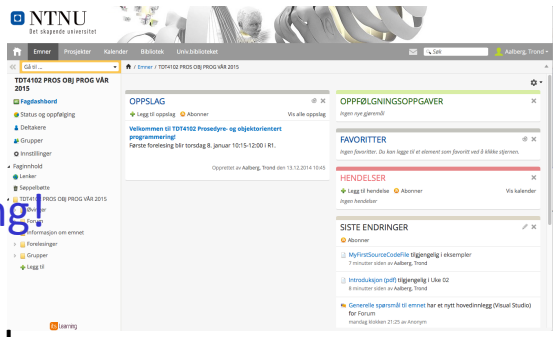
Egne notater i boka?

- Streng kontroll på eksamen!
- Boken du har med på eksamen skal være uten egne notater/kommentarer
- Greit å bruke ”utguling” eller plastlapper uten tekst
- *Ikke lurt å kjøpe ferdig kommentert bruktbok…*
- *Vanskelig å selge bok med notater*

13

Fagsider

- Vi bruker It’s Learning



- Her finner dere
  - nødvendig informasjon om faget
  - spørsmål og svar (diskusjonslister)
  - øvingsoppgaver
  - forelesingsnotater

14

Fagstab

- Faglærer/foreleser
  - Trond Aalberg
- 1 vitass
  - Einar Johan Sømåen
- 5 undasser
- 37 studasser

Vi oppfordrer til bruk av diskusjonsforumet på It's Learning for alle faglige og praktiske spørsmål

Det du lurer på er veldig ofte også relevant for andre!

Du kan poste anonymt hvis du vil

Spørsmål eller kommentarer av mer "privat" karakter:
Faglærer: Trond.Aalberg@ntnu.no
eller bruk epostlista: tdt4102-fagans@idi.ntnu.no

Praktiske/private spørsmål om øvingene kan sendes til:
Epostlista: tdt4102-undass@idi.ntnu.no

Timeplan

- Forelesinger:
  - Tirsdager i R1, 10:15-12:00 (hele semesteret)
  - Torsdager i R1, 10:15-12:00 (t.o.m. uke 6 eller 7)
- Øvingsforelesinger:
  - Fredager i R1, 10:15-12:00
- Øvingstimer på sal:
  - Avhenger av hvilken gruppe du er med i

17

Øvingsopplegg (i)

- **Alle** blir manuelt fordelt på grupper
  - Basert på studieprogram
  - Får tildelt studass som finner tidspunkter
  - Følg med på It's Learning
  - Studass ansvarlig for å finne egnede tidspunkter på sal
- Saler
  - 4. etasje i P15 bygget, 411 Rill og 414 Gorg
- Undervisningsassistentene
  - Finner du på en av salene i "kontortiden"
  - Tar i mot alle typer forespørsler
  - Bruk dette tilbudet aktivt!

18

NTNU

Øvingsopplegg (ii)

- 10 øvinger i programmering
  med veiledning og godkjenning på sal
- Må ha godkjent 8 av disse (minst 70% av hver)
- I tillegg:
  - Denne uka er det "komme i gang"-øving (teller ikke)
  - Neste uka legger vi ut en "0,5 bonusøving" som vil telle som en av de 8 obl. hvis den blir godkjent i løpet av uka
- Deretter 1 ny øving i uka (omtrent)
  - Øving legges ut den uka temaet foreleses
  - Innlevering vanligvis 2 uker etter

19

NTNU

Verktøy til programmering

- Ingen formelle krav til bruk av verktøy
  - men vi legger opp til bruk av IDE (Integrated Development Environment) som er programvare som kombinerer editor, kompilator, hjelpeverktøy med mer.
- Windows
  - Visual Studio Express 2013 (**for Windows Desktop**)
- Mac
  - XCode (Apple sitt utviklingsverktøy)
- Linux
  - Fritt valg…

Se ITL for mer info om verktøy, Veiledning i installasjon og bruk i øvingsforelesingen på fredag

20

NTNU

Referansegruppe

Vi trenger frivillige til en referansegruppe,
1 (eller fler) fra hvert studieprogram

Liste blir sendt rundt til den er fylt ut!

21

NTNU

Foreløpig forelesingsplan

- Forelesingsplanen vil bli lagt ut på web (men kan endre underveis)
- Vi dekker stort sett hele pensumboka
  - men den som kommer på forelesingene vil få med seg hva som er viktigst

22

Innholdet i emnet (i)

- Uke 2
  - Introduksjon til faget
- Uke 3
  - Grunnleggende programmering
  - Variabler, kontrollflyt, funksjoner, C++ syntaksen
  - Tabeller, **pekere**
- Uke 4
  - Praktisk programmering med funksjoner
  - Egendefinerte datatyper (struct og klasser)

Deretter kommer:

- Klasser, konstruktører, innkapsling
- Overlagring av operatorer
- C-String variabler og C++ String-klassen
- Dynamisk allokerte variabler, minnehåndtering
- Streams og fil input/output
- Rekursjon og intro til arv
- Arv og polymorfi, virtuelle funksjoner
- Påske
- Templates og STL
- Lenkede datastrukturer (lister og trær)
- Påske
- Unntakshåndtering
- Oppsummering

Eksamen

- Eksamen 3/6
- Spørretime ca. en uke før
- Vi legger ut gamle eksamensoppgaver
  - Men venter med løsningsforslag…

25

Spørsmål?

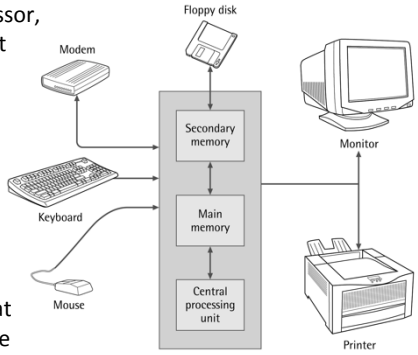
26

Introduksjon til programmering

- Maskinvare og programvare
- Maskinkode, assemblerkode, høynivå programmeringsspråk
- C++
- Fra kildekode til kjørbar fil

Maskinvare og programvare

- Datamaskiner består av
  - fysiske komponenter som prosessor, minne, lagringsenheter, inn og ut enheter (tastatur, mus, skjerm), kommunikasjonsenheter (nettverkskort oa)
- Men en datamaskin er også i stor grad programvare
  - usynlige instruksjoner som gjør at maskinvaren er i stand til å utføre spesifikke oppgaver



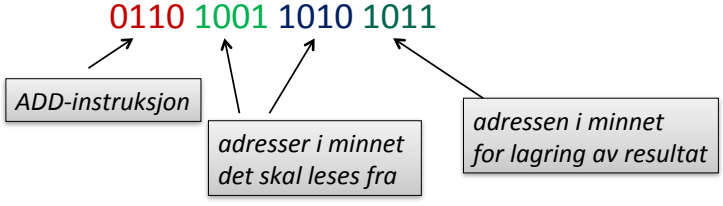
Programvare

- Programvare er et sett instruksjoner som forteller datamaskinen hvordan den skal oppføre seg
- Siden en datamaskin ikke forstår naturlig språk, er vi avhengig av å bruke et språk som maskinen forstår

29

Maskinkode

- Maskinkode er et sett av instruksjoner som er innebygd i en prosessor
  - Forskjellige typer prosessorer har forskjellige sett av instruksjoner
  - http://no.wikipedia.org/wiki/Instruksjon_(datamaskin)
- F. eks. kan vi ha en instruksjon som
  - legger sammen verdiene lagret på to forskjellige minnelokasjoner
  - lagrer resultatet i en tredje minnelokasjon
- På binær form kan dette være:



30

Assembler kode

- Assemblerkode brukes for å gjøre det enklere å skrive og lese maskinkode

  ADD X Y Z

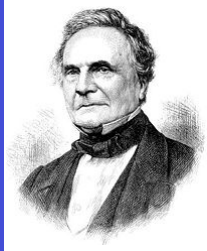
- Assembler-språk er kun en mnemonisk utgave av maskinspråket
  - kan oversettes direkte til maskinkode
  - er derfor også plattformavhengig
- Maskinkodeinstruksjonene er i tillegg ganske så primitive operasjoner
  - tidkrevende å skrive komplekse programmer, vanskelig å formulere ønsket funksjonalitet direkte vha. maskinkode

Høynivå programmeringsspråk

- Er utviklet for å kunne:
  - uttrykke ønsket funksjonalitet med et språk som er lett å lese/skrive
  - effektivt programmere og håndtere store og komplekse programmer
  - slippe å forholde seg til maskinvaren på laveste nivå, bruke mer komplekse/abstrakte måter å beskrive en løsning på
  - kunne lage kode som ikke er bundet til spesifikke sett av instruksjoner (plattformavhengig)

```
Høyde = 5;
Bredde = 8;
Areal = Høyde * Bredde
```

Programmeringsspråk



- Over hundre forskjellige høynivå språk
- Laget for spesifikke formål og mange er fortsatt i daglig bruk
- Felles for alle er at kode skrevet i disse språkene må oversettes til maskinkode
- Kjente historiske navn for de som er interessert:
  - Charles Babbage (1791-1871)
  - Augusta Ada King, Countess of Lovelace (1815-1852)
  - Og ikke minst Alan Turing

COBOL
FORTRAN
BASIC
Pascal
Ada
Visual Basic
Delphi
C
C++
Java
C#
…

33

Programmeringsspråket C++

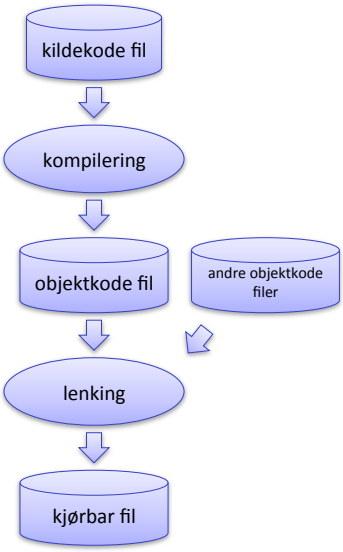
- **C++** er en videreutvikling av programmeringsspråket **C**
- **C++** ble utviklet av Bjarne Stroustrup (AT&T Bell Labs) på 1980-tallet
  - For å løse en del begrensinger ved **C**
  - Støtter objektorientert programmering
  - **C** er fortsatt et subsett av **C++**
- **C** ble på sin side utviklet av Dennis Ritchie (AT&T Bell Labs) på 1970-tallet
  - Brukt for å skrive og vedlikeholde UNIX
  - Finnes fortsatt svært mye kommersielle programvare skrevet i **C**
- **C** er igjen basert på programmeringsspråket **B**
  - Brukt til å utvikle UNIX som alternativ til assemblerkode
  - **B** var en videreutvikling av **BCPL**
- *Hvorfor "++" i C++?*

men det var aldri noe programmeringsspråk kalt **A**

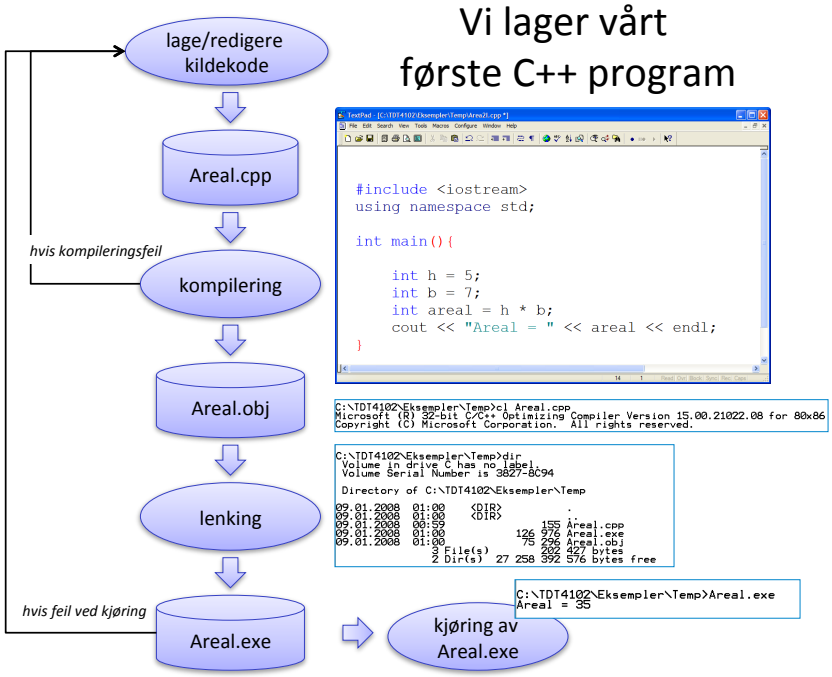
34

Fra kildekode til kjørbar fil

- Koden vi skriver i et programmeringsspråk kaller vi for kildekode (source code)
- Å oversette fra kildekode til maskinkode kalles kompilering
  - gjøres ved hjelp av et program som kalles for kompilator
  - resultatet kalles ofte objektkode
- Maskinkoden som vi selv lager må kombineres med/lenkes til annen eksisterende kode
  - dette kalles for lenking (linking)
- Sluttresultatet er en kjørbar fil



Vi lager vårt første C++ program

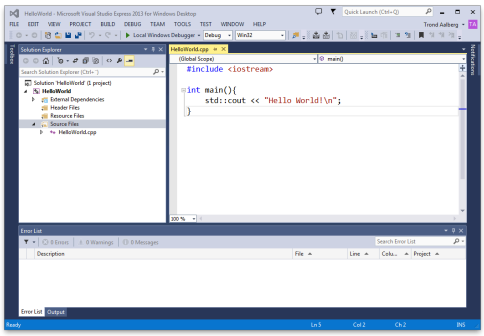


Programmeringsverktøy

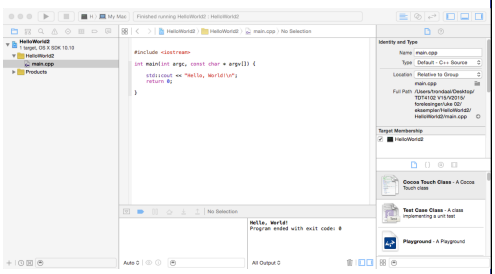
- Effektiv programmering krever mer enn det en vanlig teksteditor + kompilator kan gi
- Trenger hjelp til å
  - skrive og lese kode effektivt
  - følge språkets regler og unngå de verste feilene før kompilering
  - kompilere, kjøre, undersøke programmet vårt
  - teste og finne feil
- Vi bruker derfor utviklingsverktøy ofte kalt ”IDE”
  - ”integrated development environment“: integrert utviklingsverktøy
  - program som støtter prosessen å konstruere programvare

IDE

- Microsoft Visual Studio er et eksempel på et slikt program for Windows



- Xcode er tilsvarende verktøy for Mac



NTNU

Installering på egne maskiner

- Visual Studio Express finnes i en gratis utgave:
  - http://www.visualstudio.com/en-us/products/visual-studio-express-vs.aspx
  - Velg utgaven "**for Windows Desktop**"
- Apple Xcode er også gratis og tilgjenglig i App Store

NTNU

Andre versjoner?

- De fleste tidligere versjoner av Visual Studio og Xcode vil fungere greit
- GUI kan være litt forskjellig, men stort sett samme funksjonalitet
- Siste versjoner vil ha best støtte for C++ v11 (siste versjon av språket), men det har liten betydning i dette faget

Alternativ IDE

- Finnes også andre IDE som du kan bruke hvis du for eksempel ønsker open source verktøy, eller IDE som kjører på flere plattformer:
  - Code::Blocks (kjører på de fleste plattformer)
  - Eclipse kan også brukes for C++
  - Qt Creator
  - Emacs, gcc, gdb, make
    (for linuxkommandolinjeentusiaster)

Oppsummering

- Dagens tema har vært:
  - Introduksjon til faget
  - Programmeringsspråk og datamaskiner
  - Introduksjon til C++
- På neste forelesing starter vi med de mer fundamentale elementene i C++ språket
- Kjøp boka (når den kommer)
  - Eller bruk foilene frem til boka kommer på hyllene i Akademika