Introduksjon til Linux og C

TTK4125 Datastyring

Læringsmål

Studentene skal få en introduksjon til C, samt hvordan man kan skrive, kompilere og kjøre programmer under bash-shell i Linux.

Komme i gang

Bakgrunn

Denne øvingen utføres på Sanntidssalen. Vi i denne øvingen bruke *GCC* (GNU Compiler Collection) på operativsystemet *Linux*.

Programmeringsspråket *C* er tett bundet til operativsystemet *Unix* og det standardiserte grensesnittet *POSIX* (C-standardbiblioteket er bl.a. en del av POSIX). Når man skal begynne å programmere i C kan det derfor være mange fordeler med å bruke Unix/POSIX, slik at man ikke blir heftet med detaljer i andre operativsystemer (Windows), hvor man gjerne må gå noen omveier for å få tilgang til de standardiserte bibliotekene.

Linux er et Unix-liknende operativsystem, som har en nesten komplett implementasjon av POSIX. Fordelen med Linux er at det har åpen kildekode, som betyr at det er fritt og lett tilgjengelig for alle som ønsker å bruke det.

GCC har også åpen kildekode, og ble i utgangspunktet laget for å være en erstatning for C-kompilatorer som fulgte med kommersielle Unix-operativsystemer, tradisjonelt kalt CC.

Navigering under Linux/bash

Hvis ikke Linux allerede kjører på maskinen, start maskinen på nytt og velg Linux i oppstartsmenyen (som heter GRUB, og er en såkalt *bootloader*). Logg inn med brukernavn/passord: student/student.

Først åpner dere et terminalvindu ved å klikke på ikonet () øverst på skjermen. Deretter lager dere en ny mappe som dere kan lagre filene deres i med kommandoen

```
mkdir mappenavn
```

Et godt forslag til mappenavn er «gruppeXX». Gå inn i mappen med

```
cd mappenavn
```

Man kan liste opp filer i gjeldende katalog med en av disse. Den siste gir mer informasjon om filene (dato, størrelse, rettigheter):

```
ls
ls -l
```

For å gå opp (tilbake) en katalog kan man skrive «`cd ..`», og for å gå tilbake til hjemmekatalogen skriver man bare «`cd`»

Redigere kildefiler

For å redigere kildefilene til øvingen kan dere benytte skriveprogrammet gedit. Gedit finnes i systemmenyen et sted, men det letteste er starte det fra kommandolinjen, med en av følgende kommandoer: (Den første hvis man vil åpne en fil med en gang.)

```
gedit filnavn &
gedit &
```

Husk &-tegnet til slutt. «&» etter en vilkårlig kommando vil kjøre kommandoen i bakgrunnen. Hvis man ikke bruker «&» her, så vil man ikke kunne skrive inn flere kommandoer i terminalvinduet før gedit er avsluttet.[1]

Tips! Gedit er et greit skriveprogram, men for programmering finnes det bedre alternativer. Andre gode valg er Emacs, gvim eller vim.

Se forøvrig:

http://www.linuxconfig.org/Vim_Tutorial
http://www2.lib.uchicago.edu/~keith//tcl-course/emacs-tutorial.html
http://en.wikipedia.org/wiki/Editor_wars

---

[1] Alternativt kan man starte programmet uten &-tegnet og benytte seg av bash-skallets jobbkontrollfunksjoner. Etter at man har startet et program kan man bruke tastekombinasjonen `ctrl-z` for suspendere programmet og gi kontroll tilbake til skallet. Skriver så kommandoen `bg` (forkortelse for background) vil programmet utføres i bakgrunnen. Dette er helt ekvivalent med å bruke &-tegnet.

Likeledes kan man bruke kommandoen `fg` (foreground) for å bringe et program tilbake til forgrunnen.

Kompilere og linke

Når programmet er ferdig, skal det kompileres og linkes til en kjørbar fil. Dette gjøres i terminalvinduet. Kommandoen for kompilering og linking på en gang er:

```
gcc -Wall -o navn-på-kjørbar-fil navn-på-kildefil
```

Eksempel: For oppgave 1a) blir kommandoen

```
gcc -Wall -o print print.c
```

`gcc` er navnet på kommandoen. `-Wall` er en opsjon for kommandoen `gcc` som forteller `gcc` at man ønsker å få skrevet ut alle feil og advarsler som forekommer under kompilering og linking. -o opsjonen tar argumentet *`navn-på-kjørbar-fil`* (`-o outfile`).

Hvis det ikke skjer feil under kompileringen kommer ingen informasjon på skjermen. Man kan liste opp filer i katalogen med `ls`, og da vil man kunne finne igjen det nye programmet. Kjørbare filer har ofte grønne navn når de listes opp.

Rette eventuelle feil i programmet

Om programmet inneholder feil, så vil `gcc` returnere med informasjon om hva som gikk feil og hvor. Denne informasjonen er ofte nyttig for å rette opp feilen.

Hvis man for eksempel glemmer et semikolon, så vil man få opp feilmeldinger som ligner på dette:

```
print.c: In function ‘main’:
print.c:7: error: expected ‘;’ before ‘return’
print.c:8: warning: control reaches end of non-void function
```

Den første linjen sier at feilen finnes i filen print.c under funksjonen `main`. Den andre linjen sier at det trolig mangler et semikolon på linje 7.

Advarselen om at noe er galt på linje 8 er i dette tilfellet feil; den er en konsekvens av forrige feil på linje 7. Forsøk alltid å rette opp feil ovenfra og ned: Én enkel feil i toppen av en fil kan gi mange feilmeldinger nedover i filen.

Kjøre programmet

Det ferdige programmet kjøres med kommandoen

```
./programfil
```

for eksempel

```
./print
```

Komprimere filer og kopiere til/fra hjemmeområdet

Filer lagret lokalt på PC-ene kan bli slettet mellom hver lab, så ta kopi av øvingene deres. Filer kan pakkes ned og opp med kommandoene

```
zip navn-på-zipfil fil[er]
unzip navn-på-zipfil
```

Den letteste måten å ta vare på de pakkede filene på er å sende de via e-post til seg selv. Det er også mulig å kopiere filene rett til hjemmeområdet, med en av følgende kommandoer:

```
scp fil[er] brukernavn@login.stud.ntnu.no:katalog
scp fil[er] brukernavn@login.stud.ntnu.no:
```

Det er ikke nødvendig å spesifisere en katalog – i så fall havner filene i toppkatalogen på hjemmeområdet – men man må ha med kolonet uansett. For å kopiere fra hjemmeområdet til lokal disk kan man bruke kommandoen nedenfor.

```
scp brukernavn@login.stud.ntnu.no:katalog/filnavn .
scp brukernavn@login.stud.ntnu.no:filnavn .
```

Andre nyttige kommandoer

cp	Kopierer filer
mv	Flytter filer
rm	Sletter filer. NB! Filer slettet på denne måten kan ikke gjenopprettes.
rmdir	Fjerner tomme underkataloger
cat	Lister opp innholdet i filer. (Kan også gjøre mye annet)
grep	Søker gjennom filer etter tekst
tar	Pakker sammen og pakker opp .tar-filer Pakke ned: `tar -xcf` *`nav-på-tarfil fil[er]`* Pakke opp: `tar -xvf` *`nav-på-tarfil`*
bzip2	Komprimerer fil til .bz2
bunzip2	Dekomprimerer .bz2-filer
gzip	Komprimerer fil til .gz
gunzip	Dekomprimerer .gz-filer

Hvordan finne hjelp!

Dersom man er usikker på en Linux-kommando så prøv følgende hjelpefunksjoner:

```
«kommando» --help
«kommando» -h
man «tema»              (Trykk «q» for å gå ut av man).
man -k «nøkkelord»      (For å søke etter ting som inneholder nøkkelordet)
```

Man kan også søke etter hjelp om funksjoner i C-standardbiblioteket med `man`.

Eksempel:

```
man -k strtod
strtod (3) - convert ASCII string to floating point number
```

Det som skrives ut fra man kommandoen er navnet på tema/kommando etc. som passer med nøkkelordet (her er du kun én oppføring som passer med `strtod`), hvilken seksjon man-siden hører til (her er det seksjon 3, som har med C-standardbiblioteket å gjøre), samt en kort beskrivelse av oppføringen.

Hvis det hadde vært treff i flere seksjoner enn én, så kan man spesifisere fra hvilken seksjon man ønsker å hente man-siden fra.

```
man 3 strtod
```

Hvis man-sidene for C-standardbiblioteket av en eller annen grunn mangler på maskinen du sitter på, kan man også finne dem på verdensveven:

http://www.linuxsavvy.com/resources/linux/man/man3/

En annen uvurderlig kilde til informasjon om programmering og Linux er:

www.google.com

For mer informasjon om bash og hvordan man arbeider i shell, se:

http://www.itk.ntnu.no/ansatte/Hendseth_Sverre/bash/index.html

For en mer kuriøs beskrivelse av datarelaterte begreper, se «The Jargon File»:

http://www.catb.org/~esr/jargon/html/go01.html

Sverres introduksjon til C:

http://www.itk.ntnu.no/ansatte/Hendseth_Sverre/c/index.html

Oppgaver

Oppgave 1

a) Lag et program som skriver ut «Hello World!» på skjermen. Kall filen som inneholder koden for «print.c». (Tips: Se K&R kapittel 7.2).

b) Modifiser print.c. Programmet skal ta nå ta et argument som angir hvor mange ganger «Hello World!» skal skrives ut på skjermen. Hvis man for eksempel skriver

```
./print 5
```

så skal «Hello World!» skrives ut 5 ganger på skjermen.

Tips! Bruk `man` til å finne informasjon om funksjonene `atoi` eller `strtol`.

c) Du skal nå lage et program i filen «name.c». Dette programmet skal kunne lese inn fornavn og etternavn fra tastaturet. Disse skal skrives til skjermen og antall bokstaver i fornavnet og etternavnet skal også beregnes. (Tips: Se K&R kapittel 7.4).

Tips! Se `man scanf, strlen, fgets, stdio`. Det er enklest å sette et maksimalt antall tegn på navnene, ved å lagre navnene i variabler av typen

```
char fornavn[MAKS_ANTALL_TEGN];
```

Oppgave 2

Lag en kalkulator. Den skal kunne multiplisere, subtrahere, multiplisere og dividere. Resultatene skal skrives ut på skjermen. Pass spesielt på tilfellet der man prøver å dele med null.

Hvordan dere velger å ta input er opp til dere selv. Det enkleste er å ta inn tallene og operatoren hver for seg. Dere kan også forsøke å lese inn et regnestykket fra en tekststreng.

Oppgave 3

Lag en sorteringsalgoritme for en array av positive heltall. Dere taster inn en array fra tastaturet. Antall elementer er vilkårlig (men kan begrenses oppad). En sortert array skal så skrives ut på skjermen.

Forslag til sorteringsalgoritme

«Insertion sort» sammenlignes ofte med hvordan man sorterer kortene man har på hånden under et kortspill. Man bygger opp en sortert liste fra venstre mot høyre, slik at de *N* første kortene alltid er sortert.

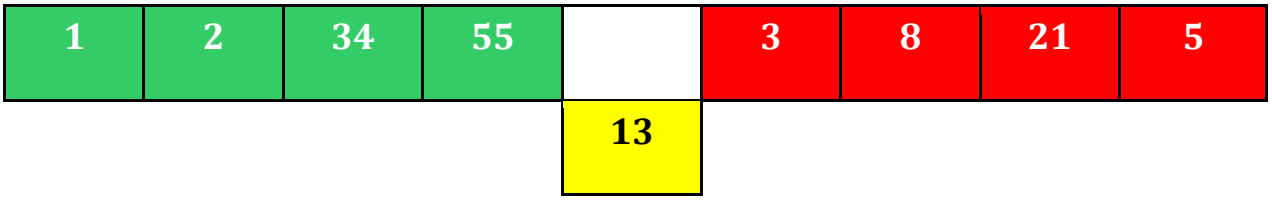
Så plukker man kort nummer N+1. Målet er nå å plassere det slik at de N+1 første kortene blir i sortert rekkefølge. Hvis kortet man plukket ut er større enn kort N, så gjør man ingen ting, for da er de N+1 første kortene allerede sortert. Hvis det er mindre, så flytter man kort N til plassen N+1, og sammenligner kortet man plukket med kort N-1. Hvis det er større så setter man inn kortet man plukket på plass N, hvis ikke flytter man kort N-1 én plass opp og prøver videre.

Når man har fått plassert kortet, og de N+1 første kortene er blitt sortert, så hopper man videre og sorterer det neste kortet.

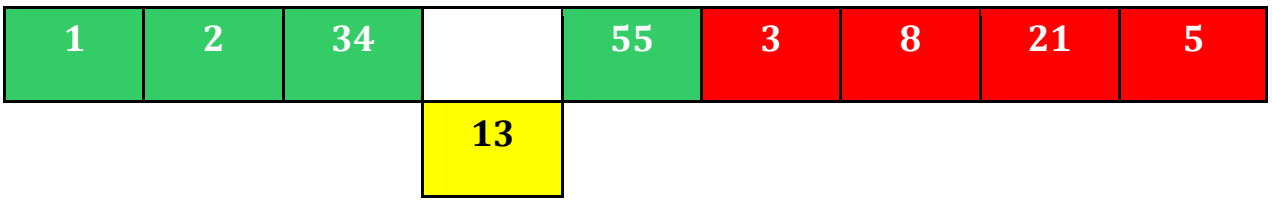
Eksempel:

La oss si at vi allerede har sortert de fire første kortene:

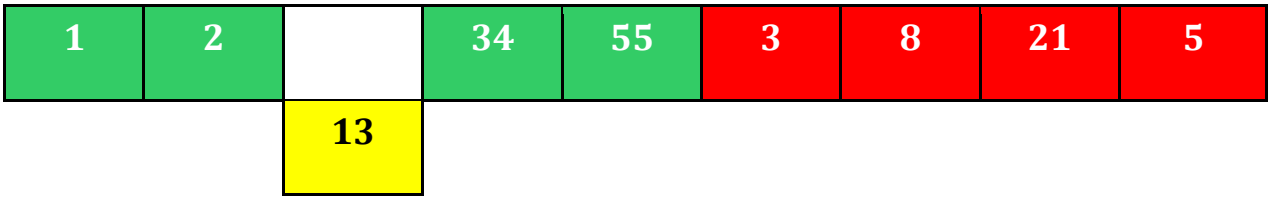
Da plukker vi ut det femte kortet:



Siden det siste sorterte kortet (55) er større enn 13, så står ikke 13 på riktig plass i listen av de 5 første kortene. Derfor flytter vi 55 en plass opp.



Nå sammenligner vi 13 med 34. Igjen står 13 på gal plass, så vi flytter 34 et hakk opp.



Nå sammenligner man 13 med 2. I forhold til 2 så står 13 nå på riktig plass, og vi kan sette den inn i listen.

Nå er de 5 første elementene i sortert rekkefølge. Man tar nå ut 3 på samme måte som vi tok ut 13, og flytter opp tall til 3 havner på sin riktige plass. Dette kan man gjenta til hele listen er sortert.