UNIVERSITETET I OSLO
Institutt for informatikk

Emacs for nybegynnere

Kompendium 33

Dag Langmyhr

Høsten 2002

Emacs for nybegynnere

Dag Langmyhr
Institutt for informatikk
Universitetet i Oslo

Høsten 2002

Sammendrag

Emacs er et såkalt **redigeringsprogram** (på engelsk gjerne kalt «text editor» eller bare «editor») som brukes til å lage og endre tekstfiler. Det er *ikke* et program for dokumentproduksjon som LaTeX, FrameMaker eller Word; man kan for eksempel ikke velge slikt som fet skrift eller sette inn bilder eller matematiske formler. Derimot er Emacs et ypperlig redskap for å skrive ren tekst og spesielt programmer i Java og andre programmeringsspråk.

Emacs er utviklet av i første rekke *Richard M. Stallman*. Det er skrevet i en egen dialekt av programmeringsspråket Lisp og dette gjør det usedvanlig enkelt å utvide Emacs eller tilpasse det egne ønsker. Emacs er en del av GNU-prosjektet som tilbyr programmer hvor alle kan få gratis adgang til kildekoden.

Dette heftet er ment å presentere Emacs for nye brukere, spesielt studenter og ansatte ved Ifi (Institutt for informatikk). Tidligere versjoner har vært laget for å trykkes på papir, men denne er ment å leses på skjermen.[1] Dessuten er den oppgradert til Emacs-versjon 20.7.

Et nybegynnerhefte som dette vil aldri fortelle den hele og fulle sannheten, men hvis det er noe du lurer på etter å ha lest heftet, finner du det helt sikkert i den offisielle «GNU Emacs Manual».

Blindern, 8. juli 2002

Dag Langmyhr

[1] I tilfelle noen lurer på det kan jeg fortelle at dette skrivet er laget med LaTeX (redigert med Emacs selvfølgelig ☺) med dokumentklassen lecnotes og pakken hyperref for å lage hyperlinker.

Innhold

Figurer

Tabeller

Det første Mødes Sødme…
— Bjørnstjerne Bjørnson

Kapittel 1

Eksempel på en redigering

Dette kapittelet tar for seg en enkelt redigeringssesjon for å vise hvorledes det er å arbeide med Emacs. Vi antar at det finnes en liten fil ved navn min.fil som vi skal endre. Fremgangsmåten er akkurat den samme hvis vi skal lage en helt ny fil.

1.1 Oppstart

Hvordan man går frem for å starte en redigering med Emacs varierer med hva slags maskin eller terminal man sitter ved.

Før du starter Emacs, bør du imidlertid forsikre deg om at Emacs ikke allerede går på din maskin. Hvis du finner et felt i programoversikten nederst på skjermen hvor det står «emacs», bør du klikke på dette feltet for å hente frem Emacs-vinduet og lese videre fra avsnittet om *Åpning av fil* på neste side.

1.1.1 Oppstart på en Unix-maskin med X

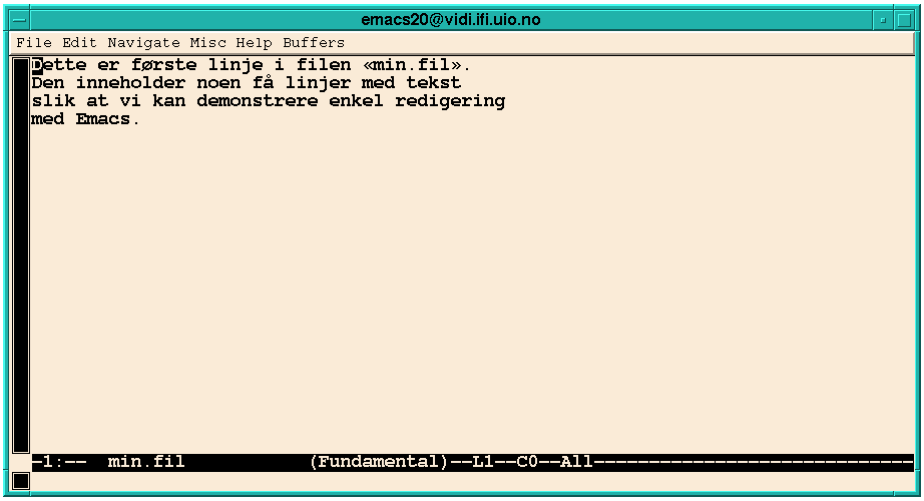
Institutt for informatikk har i dag en maskinpark med mange ulike typer datamaskiner. Felles for dem alle er at de benytter et vindussystem ved navn **X** til å styre skjermen, så Emacs fungerer likt på alle. På disse maskinene finner du dette bildet med en notisblokk i rekken langs venstre skjermkant. Klikk på det med musen for å starte Emacs.

1.1.2 Oppstart på en mer primitiv Unix-maskin

Om du kjører på en Unix-maskin uten bruk av X – for eksempel fordi du kjører hjemmefra med modem mot en av Ifis maskiner – får ikke Emacs et eget vindu, men må benytte kommandovinduet. Da må du starte Emacs med kommandoen

```
emacs
```

hver gang du skal redigere noe.



Figur 1.1: Emacs-vinduet

1.1.3 Oppstart på en Windows-maskin

Hvis du har en Windows-maskin hjemme, kan du få **Ifi-CDen** som inneholder diverse programmer som brukes i undervisningen ved Ifi. Blant disse er Emacs. Når du har installert Emacs fra Ifi-CDen, vil du ha et ikon på skjermen som vist her. Emacs startes da ved å dobbeltklikke på dette ikonet.

1.2 Åpning av fil

Oppgaven er altså å endre litt på filen min.fil, så den må hentes inn i Emacs. Dette gjøres enkelt i tre steg:

1) Trykk på tasten merket F3 .
2) Teksten Find file: ~/ vil nå dukke frem på den nederste linjen i vinduet. Skriv filnavnet min.fil som fortsettelse på det maskinen skriver; syv tegn, hverken mer eller mindre.
3) Trykk så på tasten merket Return eller ↵ .

Så hentes filen vår inn i Emacs.

1.3 Skjermbildet

Etter å ha hentet filen min.fil inn i Emacs, har vi et skjermbilde som ser ut som vist i figur 1.1. Det er dette vi skal arbeide med.

Skjermbildet består av fire deler:

1) Aller øverst er meny-linjen. Her kan vi hente frem kommandomenyer ved å peke på de ulike navnene og trykke ned venstre mus-tast og holde den nede.

2) Under er et vindu mot bufferen som inneholder min.fil. Dette vinduet opptar hele resten av skjermen unntatt de to nederste linjene. Ett eller annet sted i dette vinduet befinner markøren seg; i dette eksemplet står den øverst til venstre. Markørens posisjon indikeres av den sorte firkanten.

3) Statuslinjen (som blir forklart i neste avsnitt) befinner seg i invers skrift (dvs hvit skrift på sort bakgrunn) i nest siste linje av skjermbildet.

4) Mini-bufferen (som blir omtalt i avsnitt 2.3.1.1 på side 11) befinner seg alltid på nederste skjermbildelinje.

1.3.1 Statuslinjen

Denne linjen ligger alltid under hvert vindu på skjermbildet og gir en del statusopplysninger om bufferen. Statuslinjen består av følgende deler:

Status-merket står alltid først på linjen. På en svært kompakt måte forteller det mye om filen som redigeres:

- Aller først står det alltid en strek ('–').
- Deretter angis kodingen som benyttes; hos oss benyttes bare to alternativer:[1]

  – angir ASCII.

  1 angir ISO 8859-1.

  I avsnittet *Tegnsett* på side 13 står det mer om dette.
- Så angis hvilken form for linjeskille som benyttes:

  : betyr at det er brukt Unix-skilletegn.

  **** eller **(DOS)** betyr at linjene skilles slik Windows vil ha det.

  / eller **(MAC)** betyr at linjene skilles på Mac-måten.

  Det står mer om dette i avsnittet *Linjeskille* på side 13.
- Etterpå angis om noe er endret i bufferen. Det er tre muligheter:

  1) -- angir at intet er endret i bufferen.

  2) ** angir at innholdet i bufferen er endret.

  3) %% angir at innholdet i bufferen kommer fra en fil som brukeren ikke har lov til å endre.

Buffer-navnet gir navnet på bufferen (min.fil i figur 1.1 på forrige side). Dette er også siste del av navnet på den tilhørende filen.

Modus for bufferen følger så, omgitt av parenteser. (Les mer om dette i avsnittet *Begrepet modus* på side 13.) I figur 1.1 på forrige side er modus Fundamental.

[1] Kommandoen list-coding-systems gir en fullstendig oversikt over hvilke tegnsett Emacs kjenner til og hvorledes hver enkelt angis i status-merket.

Linjenummeret viser hvilken linje markøren befinner seg i.

Kolonnenummeret angir hvor langt ut på linjen markøren står.

Posisjonen står sist på statuslinjen. Dette kan angi på 4 forskjellige måter:

1) All angir at hele bufferen fikk plass i vinduet.
2) Top angir at første linje i bufferen vises i vinduet.
3) Bot angir at siste linje i bufferen vises i vinduet.
4) *nn*% angir hvor langt ut i bufferen teksten i vinduet befinner seg.

1.4 Enkel redigering

Vi skal nå endre ordet demonstrere til vise.

Først flyttes markøren (den sorte firkanten) slik at den står like foran ordet vi skal forandre. Dette kan gjøres enten ved å benytte pil-tastene, eller, på en maskin utstyrt med mus, ved å peke med musen og så trykke på venstre mus-tast.

Så fjernes ordet ved å trykke 11 ganger på tasten merket Del eller Delete ; det er også mulig å velge kill-word i **Edit**-menyen.

Nå kan det nye ordet vise skrives inn. Alle vanlige tegn som brukeren skriver, legges inn i teksten der markøren står.[2]

1.5 Å skrive bufferen til fil

Endringene vi har gjort nå, er bare gjort internt i Emacs-bufferen. Vi må derfor sørge for at Emacs oppdaterer den aktuelle filen, og dette gjøres ved å trykke på tasten merket F4 . Emacs spør da

```
Save file /uio/platon/mn-l0/dag/min.fil? (y or n)
```

Svaret er 'y' (for *yes*), siden vi ønsker å ta vare på endringene vi gjorde.

Noen ganger får man spørsmålet[3]

```
Select coding system (default iso-8859-1):
```

Bare trykk på retur-tasten (merket Return eller ↩).

[2] Egentlig er de vanlige tastene bundet opp til funksjonen self-insert-command som altså legger det tegnet som ble skrevet, inn i bufferen. Det er således ingenting i veien for å binde kommandoer til vanlige taster.

[3] Dette spørsmålet skyldes at Emacs først trodde vi redigerte en ASCII-fil men etterhvert har skjønt at det er en fil i tegnsettet ISO 8859-1 og vil gjerne ha bekreftet dette. I avsnittet *Tegnsett* på side 13 står det mer om dette.

1.6 Avslutning

Hvis man kjører Emacs i et eget vindu (og det gjør man nesten alltid), trenger man ikke avslutte Emacs før man er helt ferdig med arbeidet og skal forlate maskinen. Da bør man imidlertid absolutt avslutte Emacs for å være sikker på at alle endrete filer er skrevet tilbake.

Når man vil avslutte Emacs, gjøres dette ved å velge save-buffers-kill-emacs i **File**-menyen. Ellers står det mer om dette i avsnittet *Avslutning* på side 18.

Navnet «Emacs» står for
Escape-Meta-Alt-Control-Shift.
— Ondsinnet rykte

Kapittel 2

Grunnleggende begreper

Dette kapittelet tar for seg diverse begreper som alle brukere av Emacs bør kjenne til.

2.1 Notasjon

2.1.1 Notasjon for taster

I dette dokumentet vil tastene bli benevnt på to forskjellige måter:

1) Det vanligste er at de blir benevnt slik Emacs selv ville nevnt dem (se avsnitt 2.1.2 nedenfor), det vil si som A, f, ESC eller C-x.

2) Når det skal gjøres tydelig at det dreier seg om tastetrykk på en vanlig tast eller en spesialtast, tegnes tasten slik: [A] eller [F3] eller [Esc]. Om det er nødvendig med flere tastetrykk samtidig, angis dette ved å skrive en '+' mellom, som i

   [Ctrl] + [X] eller [Shift] + [A]

2.1.2 Kontrolltegn

Emacs bruker en litt annen notasjon for kontrolltegn enn det som er vanlig:

- En del kontrolltegn har egne navn, som TAB, ESC, LF, SPC[1] og DEL.
- C-x betegner ASCII-tegnet **kontroll-x**. Dette kontroll-tegnet dannes ved å trykke på [Ctrl] + [X], det vil si ved å holde tasten merket [Ctrl] eller [Control] nede samtidig som man trykker på [X].
- M-x angir tegnet **meta-x** som dannes ved å trykke på **meta-tasten** samtidig med [X]. Denne meta-tasten er merket [Alt] på de fleste maskiner.

  I Emacs er det også mulig å angi et meta-tegn ved først å trykke på [Esc] og deretter på den aktuelle tasten.

[1] SPC betegner en blank, altså et trykk på mellomromstasten.

- C-M-x angir en kombinasjon av meta- og kontroll-tegn. Det dannes ved å trykke på tre taster samtidig: Meta + Ctrl + X .

  Alternativt kan man taste inn dette tegnet ved først å trykke på Esc og deretter på Ctrl + X samtidig.

2.2 Kall på kommandoer

All redigering i Emacs skjer ved å kalle på forskjellige redigeringskommandoer. Dette kan gjøres på flere måter.

2.2.1 Bruk av mus

Omtrent alle datamaskiner er utstyrt med en **mus** eller lignende pekeredskap som kan styre en markør på skjermen. Noen kommandoer kan utføres ved å bruke musen til å peke med og så trykke på en av de tre knappene på den.

Den aller vanligste operasjonen når man jobber med Emacs er å flytte Emacs-markøren til en eller annen posisjon på skjermen. Dette kan man for eksempel gjøre ved å peke med musen og så klikke med venstre mus-tast. Les mer om flytting i Emacs i avsnittet *Manøvrering i bufferen* på side 21.

2.2.2 Menyer

Emacs er utstyrt med menyer slik at de aller vanligste kommandoene kan uføres ved å velge i en av disse menyene. En bruker får frem en meny ved å peke på dens navn på øverste linje og trykke ned venstre mus-tast. Man kan så peke på det alternativet man ønsker, og til slutt slippe mus-tasten for å få det utført.

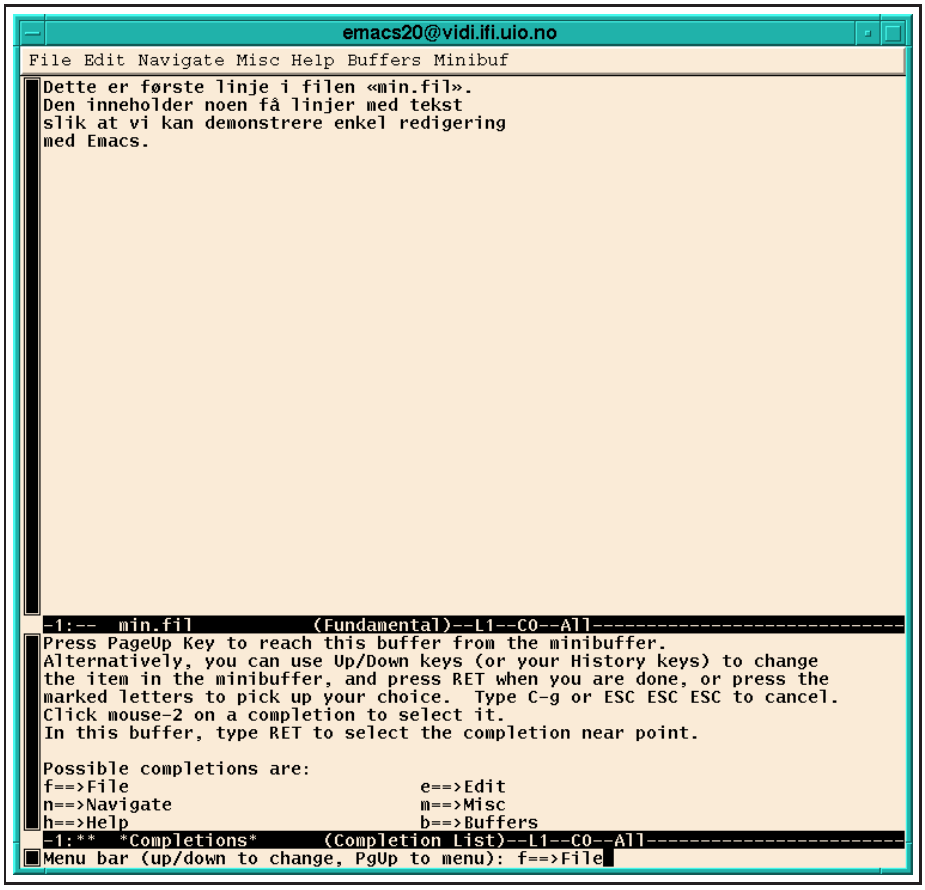
Som eksempel kan nevnes muligheten for å bytte om to ord. Dette kan gjøres ved å velge transpose-words som finnes i **Edit**-menyen (i nederste linje).

Henter man frem en meny men finner ut at man ikke ønsker noen av alternativene, kan man bare peke utenfor menyen når man slipper mus-tasten.

I kapittelet *De viktigste kommandoene* på side 17 finnes en oversikt over alle de viktigste Emacs-kommandoene. Der står det også angitt om de finnes i en meny, og i så fall i hvilken meny.

2.2.2.1 Valg fra menyer uten bruk av mus

Om man av en eller annen grunn ikke kan benytte musen, kan man allikevel velge kommandoer fra menyene. Trykk på F10 så deles skjermen i to slik det er vist i figur 2.1 på neste side. Her kan man velge **File**-menyen ved å trykke på F , **Navigate**-menyen ved å trykke på N osv. Da dukker den aktuelle menyen frem på samme måte, og man kan velge videre.



Figur 2.1: Menyer i Emacs-vinduet

2.2.3 Spesialtastene

Terminaler er utstyrt med diverse spesialtaster i tillegg til det vanlige alfabetiske tastaturet. Imidlertid varierer det sterkt hvor mange slike spesialtaster som finnes, hva de er merket med, og hvor de er plassert. Omtrent alle tastaturer har dog **funksjonstaster** merket F1 , F2 og så videre opp til F12 .

Ved å binde de aller vanligste kommandoene i Emacs til slike funksjonstaster vil redigeringsarbeidet kunne gå raskere; det vil også være enklere for nybegynnere.

Anta at du ønsker å flytte til linje 244 i en fil (fordi du for eksempel har fått melding fra kompilatoren om en feil i den linjen). Da kan du trykke

F6 2 4 4 Return

2.2.4 Kontrolltastene

Fordi det ikke finnes så mange spesialtaster på tastaturet, benytter man også kontrolltastene til å få utført kommandoer. Eksempelvis kan man bytte om to tegn ved å trykke på C-t.

Selv om det er mange flere kontrolltaster enn spesialtaster, er det allikevel ikke nok av dem. Derfor benyttes noen av kontrolltastene som innledning til sekvenser av to eller flere tastetrykk; dette gjelder spesielt C-c og C-x. For eksempel kan man splitte et vindu i to ved å trykke C-x 2;[2] mer informasjon om splitting av vinduer finnes i avsnittet *Bruk av flere vinduer* på side 32.

De aller fleste kontrolltastene er bundet til samme kommando hele tiden, men noen forandrer binding i de forskjellige modi (se avsnittet *Begrepet modus* på side 13). Nærmere opplysninger kan brukeren få ved å benytte kommandoene b og m til hjelpe-kommandoen (se kapittelet *Hjelp fra Emacs* på side 39).

Til høyre i de fleste menylinjene står det nevnt hvilke kontrolltaster man kan bruke for å få utført kommandoen.

2.2.5 Kommandonavn

Selv med mus, menyer, spesial- og kontrolltaster har vi ikke nok kombinasjonsmuligheter til å dekke alle de hundrevis av kommandoer som finnes i Emacs.[3] Derfor har alle kommandoene i Emacs et navn, og det er oftest et navn som beskriver ganske godt hva den gjør, som for eksempel end-of-line eller backward-delete-char-untabify. Den første av disse to flytter markøren til slutten av linjen, og den andre fjerner ett tegn (det forrige) samtidig som den, om nødvendig, endrer en TAB til det riktige antall blanke.

Man kan alltid få utført en kommando ved å skrive M-x og så angi navnet, som f.eks.:

Meta + X end-of-line Return

2.2.6 Mange måter å oppnå det samme på

Tanken bak Emacs er at brukeren utfører *kommandoer*, og at det skal finnes flere muligheter til å bruke de vanligste kommandoene. Anta for eksempel at jeg ønsker å flytte Emacs-markøren til starten av linjen. Det kan jeg gjøre på følgende måter:

- Peke med musen på starten av linjen og klikke med venstre mus-tast.
- Bruke musen til å velge beginning-of-line i **Navigate**-menyen.
- Foreta samme menyvalg ved å trykke

  F10 N B

- Trykke på tasten Home (om det finnes en slik tast).
- Trykke på kontrolltasten C-a.
- Angi kommandonavnet:

[2] Man kan oppnå det samme ved å trykke på F12. Dette er et eksempel på at de vanligste Emacs-kommandoene kan utføres på flere måter.

[3] Et lite ord til trøst: Selv om det finnes mange hundre kommandoer i Emacs, vil en vanlig bruker klare seg lenge med 20–25 av dem, for eksempel de som finnes i menyene.

Meta + X beginning-of-line Return

Hvilken av dem jeg velger er opp til meg selv og min arbeidsstil. Det er imidlertid to gode grunner til å bruke musen så lite som mulig:

- Det er raskere å bruke tastaturet.
- Mange får slitasjeskade (såkalt «**musesyke**») ved å bruke musen for mye.

2.3 Terminologi

Dette avsnittet forklarer noen av de begrepene Emacs benytter.

2.3.1 Begrepet buffer

I Emacs redigerer man ikke filer, men **buffere**. Første gang en fil refereres leses innholdet av den inn i en buffer, og all behandling skjer deretter på denne bufferen. Først når brukeren eksplisitt ber om det (se avsnittet *Å gjemme unna buffere* på side 20), blir innholdet av bufferne skrevet tilbake til sine respektive filer.

2.3.1.1 Mini-bufferen

Mini-bufferen er en spesiell buffer som alltid kan sees på nederste linje på skjermen. Den benyttes når Emacs trenger navn på filer, kommandoer, søkestrenger eller lignende. Mini-bufferen er i de fleste sammenhenger en buffer på lik linje med andre buffere, og vanlige redigeringskommandoer fungerer også her.

2.3.2 Begrepet ramme

Når man kjører et system med muligheter for flere vinduer (som X og Windows), kan også Emacs benytte flere vinduer til å vise sine buffere. I Emacs-terminologien kalles et slikt vindu for en **ramme** (på engelsk «frame»).[4] Avsnittet *Bruk av rammer* på side 34 gir ytterligere informasjon.

2.3.3 Begrepet vindu

Brukeren kan se på innholdet av en buffer gjennom et **vindu**. Hver ramme vil vanligvis bestå av ett vindu mot én buffer, men det er mulig å splitte opp rammen i flere vinduer som viser innholdet av hver sin buffer. Det er også mulig å ha flere vinduer mot samme buffer.

2.3.3.1 Begrepet aktivt vindu

Selv om det kan være mange vinduer i mange rammer, er det kun ett som er aktivt av gangen; dette vises ved at markøren (se neste avsnitt) er synlig i dette vinduet. All redigering skjer i det aktive vinduet.

[4] Det er historiske årsaker til at Emacs benytter dette uvanlige ordet.

2.3.4 Markøren

Hver buffer har sin egen **markør** (på engelsk «cursor») som peker inn i bufferen, men kun markøren i det aktive vinduet er synlig. Markøren vises vanligvis som en firkantet, sort boks. De fleste kommandoene lar operasjonene skje på eller nær det stedet markøren peker.

På de maskinene som er utstyrt med mus, finnes det i tillegg en **mus-markør**, og denne må for all del ikke blandes sammen med den vanlig Emacs-markøren. Mus-markøren markeres gjerne med en liten skråpil, og den har en helt annen funksjon.

2.3.5 Begrepet område

Et **område** (på engelsk «region») er definert som en brukerangitt sammenhengende tekst i en buffer. Den ene yttergrensen utgjøres av et spesielt usynlig **merke** (kalt «mark») mens markøren utgjør den andre.

De to vanligste måtene å markere et område på er følgende:

1) Sett først et merke et eller annet sted i bufferen. Dette kan gjøres ved å flytte markøren dit merket skal stå og deretter utføre kommandoen set-mark-command som setter merket. Denne kommandoen er koblet til tasten C-SPC, men man kan alternativt velge å bruke set-mark-command i **Navigate**-menyen.

   Etterpå kan markøren flyttes til den andre enden av området med piltastene.

2) På de maskinene som har mus, kan denne benyttes til å sette både markøren og merket og dermed definere et område. Flytt først mus-markøren til den ene enden av området og trykk og hold nede venstre mus-tast. «Dra» så mus-markøren over området[5] til den andre enden og slipp tasten.

2.3.5.1 Aktive områder

Et område kan være enten aktivt eller passivt.[6] Forskjellen ligger i at operasjoner på et område, som for eksempel å fjerne det, bare er tillatt på aktive områder. Vi kan se at et område er aktivt ved at det er markert på skjermen, enten ved at det har en annen farge eller ved at det er understreket.

Et nymarkert område er alltid aktivt, og det forblir aktivt inntil bufferen blir endret, for eksempel ved at tekst blir skrevet inn eller fjernet.

2.3.5.2 Passive områder

Et område kan altså være passivt, og man er da sikret at det ikke blir utført noen operasjoner på det. Området er der imidlertid fremdeles, og kan lett gjøres aktivt igjen, for eksempel ved å utføre kommandoen exchange-point-and-mark (bundet til C-x C-x) to ganger.

[5] Hvis området strekker seg over mer enn én skjermfull, er det bare å flytte mus-markøren mot kanten av vinduet mens man ennå holder nede venstre mus-tast; da vil vinduet rulle.

[6] Hvis man ikke ønsker dette skillet mellom aktive og passive områder, kan man endre Emacs-variabelen transient-mark-mode.

2.3.6 Begrepet modus

Enhver buffer er i en eller annen **modus** (på engelsk «mode») avhengig av hva den skal brukes til. Eksempelvis finnes det modi tilpasset de forskjellige programmeringsspråkene, LaTeX, brevskriving og mye annet. Vanligvis velges modus utifra endelsen på filnavnet, men det er også mulig å gi kommandoer som endrer modus.

Ideen med modus er at det er ønskelig med spesialkommandoer tilpasset særtilfeller. Enkelte kommandoer er for eksempel nyttige når man redigerer et Java-program, men meningsløse når man skriver et brev. Løsningen er altså å ha de fleste kommandoene felles, men sørge for at de mer spesielle kun finnes i egnede modi.

2.4 Nyttige ting å vite

Nå er dette kapittelet snart ferdig; det er bare et par ting som alle brukere bør vite før de setter i gang for alvor. (I kapittelet *Flere nyttige ting å vite* på side 41 står det flere nyttige opplysninger som man kan trenge når man har holdt på en stund.)

2.4.1 Tegnsett

Ulike datamaskiner og operativsystemer har ofte ulike tegnsett, det vil si hvilke tegn de kjenner til og hvorledes de er kodet. For Ifi-brukere er følgende tegnsett mest aktuelle:

ISO 8859-1 (også kjent som ISO Latin-1) er det mest utbredte tegnsettet i Vest-Europa; se tabell 2.1 på neste side.

ASCII («American Standard Code for Information Interchange») er et gammelt og velkjent tegnsett; det består det første 128 tegnene i ISO 8859-1, det vil si venstre halvpart av tabell 2.1 på neste side.

Mac har sitt eget tegnsett, også det basert på ASCII.

Unicode er et nyere tegnsett der man bruker 16 bit til å lagre hvert tegn.

En verdifull egenskap ved Emacs er at den selv oppdager hvilket tegnsett som benyttes og fortsetter å bruke dette.[7]

2.4.2 Linjeskille

Dessverre bruker de ulike operativsystemene forskjellige teknikker for å angi hvorledes en linje slutter. Følgende varianter finnes:

Unix bruker LF (ASCII-kode 10) til å skille linjene.

Windows skiller linjene med først en CR (ASCII-kode 13) og så en LF.

Mac har valgt å bruke bare CR til å skille linjene.

Emacs er heldigvis i stand til selv å oppdage hvilken form for linjeskift som brukes i en fil, og så fortsette å bruke den.

[7] Om Emacs har en feilaktig oppfatning av hva tegnsettet skal være, kan dette endres. Kommandoen list-coding-systems gir en oversikt over hvilke tegnsett Emacs kjenner til.

ISO 8859–1

0	000 00	32		040 20	64	@	100 40	96	‘	140 60	128	200 80	160		240 A0	192	À	300 C0	224	à	340 E0
1	001 01	33	!	041 21	65	A	101 41	97	a	141 61	129	201 81	161	¡	241 A1	193	Á	301 C1	225	á	341 E1
2	002 02	34	"	042 22	66	B	102 42	98	b	142 62	130	202 82	162	¢	242 A2	194	Â	302 C2	226	â	342 E2
3	003 03	35	#	043 23	67	C	103 43	99	c	143 63	131	203 83	163	£	243 A3	195	Ã	303 C3	227	ã	343 E3
4	004 04	36	$	044 24	68	D	104 44	100	d	144 64	132	204 84	164	¤	244 A4	196	Ä	304 C4	228	ä	344 E4
5	005 05	37	%	045 25	69	E	105 45	101	e	145 65	133	205 85	165	¥	245 A5	197	Å	305 C5	229	å	345 E5
6	006 06	38	&	046 26	70	F	106 46	102	f	146 66	134	206 86	166	¦	246 A6	198	Æ	306 C6	230	æ	346 E6
7	007 07	39	’	047 27	71	G	107 47	103	g	147 67	135	207 87	167	§	247 A7	199	Ç	307 C7	231	ç	347 E7
8	010 08	40	(	050 28	72	H	110 48	104	h	150 68	136	210 88	168	¨	250 A8	200	È	310 C8	232	è	350 E8
9	011 09	41	)	051 29	73	I	111 49	105	i	151 69	137	211 89	169	©	251 A9	201	É	311 C9	233	é	351 E9
10	012 0A	42	*	052 2A	74	J	112 4A	106	j	152 6A	138	212 8A	170	ª	252 AA	202	Ê	312 CA	234	ê	352 EA
11	013 0B	43	+	053 2B	75	K	113 4B	107	k	153 6B	139	213 8B	171	«	253 AB	203	Ë	313 CB	235	ë	353 EB
12	014 0C	44	,	054 2C	76	L	114 4C	108	l	154 6C	140	214 8C	172	¬	254 AC	204	Ì	314 CC	236	ì	354 EC
13	015 0D	45	–	055 2D	77	M	115 4D	109	m	155 6D	141	215 8D	173	-	255 AD	205	Í	315 CD	237	í	355 ED
14	016 0E	46	.	056 2E	78	N	116 4E	110	n	156 6E	142	216 8E	174	®	256 AE	206	Î	316 CE	238	î	356 EE
15	017 0F	47	/	057 2F	79	O	117 4F	111	o	157 6F	143	217 8F	175	¯	257 AF	207	Ï	317 CF	239	ï	357 EF
16	020 10	48	0	060 30	80	P	120 50	112	p	160 70	144	220 90	176	°	260 B0	208	Đ	320 D0	240	ð	360 F0
17	021 11	49	1	061 31	81	Q	121 51	113	q	161 71	145	221 91	177	±	261 B1	209	Ñ	321 D1	241	ñ	361 F1
18	022 12	50	2	062 32	82	R	122 52	114	r	162 72	146	222 92	178	2	262 B2	210	Ò	322 D2	242	ò	362 F2
19	023 13	51	3	063 33	83	S	123 53	115	s	163 73	147	223 93	179	3	263 B3	211	Ó	323 D3	243	ó	363 F3
20	024 14	52	4	064 34	84	T	124 54	116	t	164 74	148	224 94	180	´	264 B4	212	Ô	324 D4	244	ô	364 F4
21	025 15	53	5	065 35	85	U	125 55	117	u	165 75	149	225 95	181	µ	265 B5	213	Õ	325 D5	245	õ	365 F5
22	026 16	54	6	066 36	86	V	126 56	118	v	166 76	150	226 96	182	¶	266 B6	214	Ö	326 D6	246	ö	366 F6
23	027 17	55	7	067 37	87	W	127 57	119	w	167 77	151	227 97	183	·	267 B7	215	×	327 D7	247	÷	367 F7
24	030 18	56	8	070 38	88	X	130 58	120	x	170 78	152	230 98	184	¸	270 B8	216	Ø	330 D8	248	ø	370 F8
25	031 19	57	9	071 39	89	Y	131 59	121	y	171 79	153	231 99	185	1	271 B9	217	Ù	331 D9	249	ù	371 F9
26	032 1A	58	:	072 3A	90	Z	132 5A	122	z	172 7A	154	232 9A	186	º	272 BA	218	Ú	332 DA	250	ú	372 FA
27	033 1B	59	;	073 3B	91	[	133 5B	123	{	173 7B	155	233 9B	187	»	273 BB	219	Û	333 DB	251	û	373 FB
28	034 1C	60	<	074 3C	92	\	134 5C	124	|	174 7C	156	234 9C	188	¼	274 BC	220	Ü	334 DC	252	ü	374 FC
29	035 1D	61	=	075 3D	93	]	135 5D	125	}	175 7D	157	235 9D	189	½	275 BD	221	Ý	335 DD	253	ý	375 FD
30	036 1E	62	>	076 3E	94	^	136 5E	126	~	176 7E	158	236 9E	190	¾	276 BE	222	Þ	336 DE	254	þ	376 FE
31	037 1F	63	?	077 3F	95	_	137 5F	127		177 7F	159	237 9F	191	¿	277 BF	223	ß	337 DF	255	ÿ	377 FF

Tabell 2.1: Tegnsettet ISO 8859-1

2.4.3 Farger

For å gjøre det lettere for brukerne å redigere for eksempel et program, bruker Emacs farger til å vise hva som er nøkkelord, kommentarer, osv. Dette er en god hjelp, men det er viktig å være klar over at Emacs kan ta feil. Hvis du er uenig med Emacs om hvorledes et ord skal oppfattes, bør du stole på din egen oppfatning (i hvert fall inntil kompilatoren også forteller deg at du har gjort en feil.)

2.4.4 Å avbryte en kommando

Alle kommandoer kan avbrytes, uansett hvor langt de er kommet; dette skjer ved å benytte kontrolltasten C-g.

2.4.5 Når alt går galt

Når man starter å bruke et nytt program, vil alle oppleve at det skjer overraskende ting. Da er det godt å vite at Emacs har en **angretast**, en kommando som «reparerer» eventuelle flauser man har begått. Denne kan man bruke så mange ganger man vil for å omgjøre det man tidligere har gjort.

Tasten F7 benyttes som angretast her ved Ifi, og kommandoen som den setter i gang heter undo. Se forøvrig avsnittet *Angring* på side 37.

2.4.6 Innføringskurs

Nybegynnere anbefales sterkt å benytte det innføringskurset som finnes innebygget i Emacs. Det kjøres ved å trykke på F1 for å starte brukerstøttesystemet, og så trykke på T.

We were not born to sue, but to command.
— William Shakespeare

Kapittel 3

De viktigste kommandoene

Dette kapittelet tar for seg alle de viktigste kommandoene i Emacs.

3.1 Hjelp og informasjon

Emacs er meget godt utbygget når det gjelder interaktiv støtte til brukeren.

Navn	K-tast	F-tast	Meny	Operasjon
ifi-view-keys			**Help**	Viser oversikt over funksjons- og mus-tastene
help	C-h	F1		Diverse hjelpekommandoer
command-apropos	C-h a	F1 a	**Help**	Let etter kommandoer
describe-function	C-h f	F1 f	**Help**	Beskriv hva en kommando gjør
describe-key-briefly	C-h c	F1 c	**Help**	Fortell hvilken kommando en tast er bundet til
describe-key	C-h k	F1 k	**Help**	Beskriv kommandoen tasten er bundet til

Tabell 3.1: Kommandoer for hjelp og informasjon

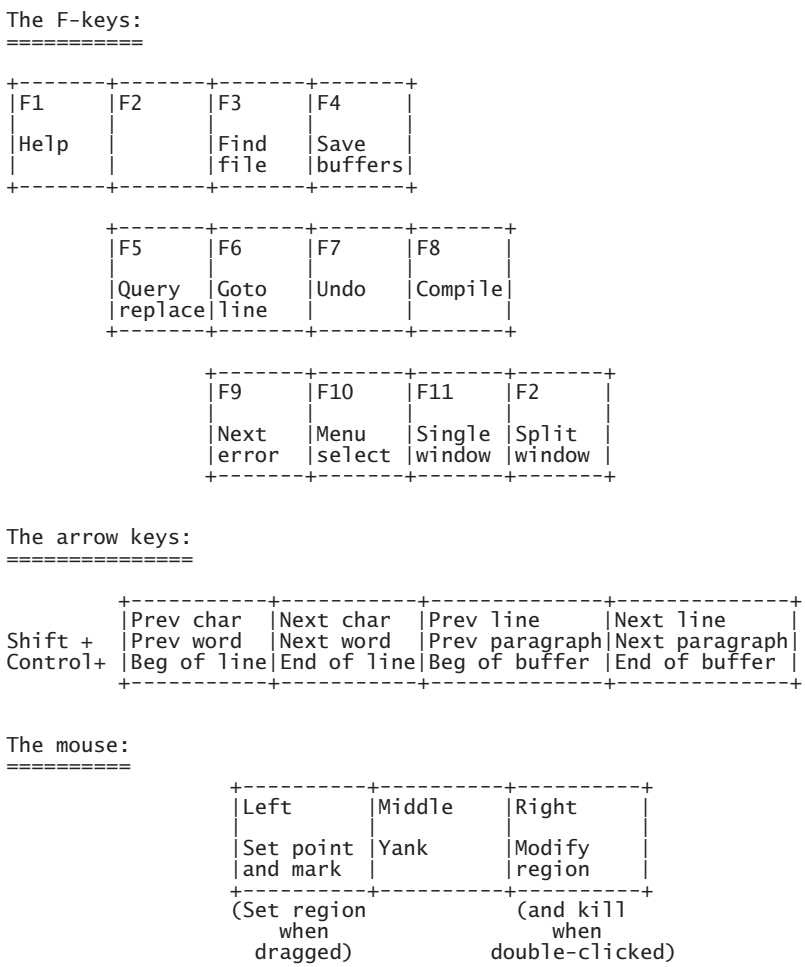
3.1.1 Informasjon om funksjons- og mus-tastene

For å gjøre det enklere for brukerne å benytte funksjons- og mus-tastene når de kjører Emacs, har vi her ved Ifi laget en kommando som heter ifi-view-keys[1] og som gir en oversikt over hvilken operasjon som er koplet til hver av disse tastene. Oversikten ser ut som vist i figur 3.1 på neste side.

3.1.2 Hjelp under kjøringen

Selve det innebygde hjelpesystemet i Emacs er som sagt meget godt utbygget. Vi får kontakt med dette systemet ved hjelp av tasten F1 eller C-h og gjennom menyen **Help**. Hele brukerstøttesystemet er beskrevet i kapittelet *Hjelp fra Emacs* på side 39.

[1] Alle kommandoer hvis navn starter med ifi- er laget her ved Institutt for informatikk.



Figur 3.1: Oversikt over funksjons- og mus-tastene

3.2 Avslutning

Navn	K-tast	F-tast	Meny	Operasjon
save-buffers-kill-emacs	C-x C-c		**File**	Avslutt Emacs
suspend-emacs				Avslutt Emacs midlertidig

Tabell 3.2: Kommandoer for avslutning

Som nevnt tidligere avsluttes aldri en Emacs-kjøring; det eneste unntaket er når brukeren skal logge ut. Emacs avsluttes med kommandoen save-buffers-kill-emacs; bruk av denne kommandoen vil skrive ut innholdet av alle endrede buffere på sine tilhørende filer (etter å ha spurt brukeren om han eller hun ønsker å ta vare på dem) og deretter avslutte kjøringen.

Det er fullt mulig å logge ut uten å avslutte Emacs først, men dette *frarådes sterkt!* Hvis man gjør det, vil alle endringer man ikke har skrevet til fil gå tapt! En slik fadese sikrer man seg altså mot ved først å avslutte Emacs og så logge ut.

3.2.1 Suspensjon

Når man kjører fra en primitiv terminal, for eksempel hjemmefra via modem, er det ofte behov for å ta en kortere pause i redigeringen for så å fortsette siden. Dette kalles å **suspendere** kjøringen. For å suspendere kjøringen benyttes kommandoen suspend-emacs. Da er det mye raskere å starte Emacs etterpå, og man gjenopptar redigeringsarbeidet nøyaktig der man var.

Prøver man å logge ut etter en slik midlertidig avslutning av Emacs, vil man få feilmeldingen 'There are stopped jobs'. Man må da starte Emacs igjen og avslutte normalt før man logger ut.

3.3 Å gjemme unna buffere

Navn	K-tast	F-tast	Meny	Operasjon
save-buffer	C-x C-s		**File**	Gjem innholdet av aktiv buffer på fil
write-file	C-x C-w		**File**	Gjem aktiv buffer, men spør brukeren om filnavn først
save-some-buffers	C-x s	F4	**File**	Gjem alle endrede buffere

Tabell 3.3: Kommandoer for å gjemme unna buffere

Kommandoen save-buffer vil skrive ut innholdet av bufferen i det aktive vinduet på den tilhørende filen; dette gjøres imidlertid kun hvis filen er endret, ellers skjer ingenting.

Kommandoen write-file fungerer akkurat som save-buffer, men den vil først spørre brukeren om navnet på filen hvor innholdet av bufferen skal gjemmes. Denne kommandoen benyttes når man ønsker å endre navnet på en buffer og dens tilhørende fil.

Kommandoen save-some-buffers vil skrive innholdet av *alle* endrede buffere ut på sine tilhørende filer; her spørres brukeren for hver fil om den skal gjemmes.

Ingen av disse kommandoene vil avbryte redigeringen; de vil bare skrive ut innholdet av filen eller filene, og så kan brukeren fortsette der han eller hun var.

3.4 Manøvrering i bufferen

Navn	K-tast	F-tast	Meny	Operasjon
backward-char	C-b	←		Flytt ett tegn mot venstre
forward-char	C-f	→		Flytt ett tegn mot høyre
backward-word	M-b	Shift + ←	**Navigate**	Flytt ett ord mot venstre
forward-word	M-f	Shift + →	**Navigate**	Flytt ett ord mot høyre
next-line	C-n	↓	**Navigate**	Flytt én linje ned
previous-line	C-p	↑	**Navigate**	Flytt én linje opp
beginning-of-line	C-a	Home	**Navigate**	Flytt til starten av linjen
end-of-line	C-e	End	**Navigate**	Flytt til slutten av linjen
goto-line		F6	**Navigate**	Flytt til angitt linjenummer
backward-paragraph	M-{	Shift + ↑	**Navigate**	Flytt ett avsnitt bakover
forward-paragraph	M-}	Shift + ↓	**Navigate**	Flytt ett avsnitt forover
scroll-down	M-v		**Navigate**	Flytt én side bakover
scroll-up	C-v		**Navigate**	Flytt én side forover
beginning-of-buffer	M-<	Ctrl + ↑	**Navigate**	Flytt til starten av bufferen
end-of-buffer	M->	Ctrl + ↓	**Navigate**	Flytt til slutten av bufferen
exchange-point-and-mark	C-x C-x		**Navigate**	Flytt til andre enden av området

Tabell 3.4: Kommandoer for manøvrering i bufferen

Dette avsnittet tar for seg diverse kommandoer for å flytte markøren rundt i bufferen.

3.4.1 Flytting på tegnnivå

Kommandoen backward-char flytter markøren tilbake ett tegn, altså mot venstre. Hvis markøren sto på første tegn på linjen, vil den bli plassert etter siste tegn på forrige linje.

Kommandoen forward-char flytter på tilsvarende måte markøren frem ett tegn. Den vil også skifte linje hvis det ikke er flere tegn igjen på linjen.

3.4.2 Flytting på ordnivå

Kommandoen backward-word flytter markøren til første tegn av det ordet markøren befinner seg i, eller til starten av foregående ord hvis markøren står mellom to ord.

På tilsvarende måte flytter kommandoen forward-word markøren forbi siste tegn av det ordet markøren er i, eller neste ord hvis markøren står mellom to ord.

Kombinasjoner av trykk gjør det mulig å raskt plassere markøren på f.eks. første tegn i neste ord:

Shift + → Shift + → Shift + ←

3.4.3 Flytting på linjenivå

Kommandoen next-line vil flytte markøren til samme posisjon på neste linje. Flyttes markøren utenfor skjermen, vil teksten automatisk rulle.

Tilsvarende vil kommandoen previous-line flytte til forrige linje.

Kommandoen beginning-of-line som normalt er bundet til tasten Home og dessuten til Control + ←, flytter markøren til starten av nåværende linje.

På samme måte flytter kommandoen end-of-line markøren til slutten av linjen; denne er vanligvis bundet til End og dessuten til Control + →.

3.4.3.1 Flytting til angitt linje

Kommandoen goto-line vil be brukeren om et linjenummer og så flytte markøren dit.

3.4.4 Flytting på avsnittsnivå

Kommandoen backward-paragraph flytter markøren til starten av det **avsnittet** markøren befinner seg i. Det vil variere fra én type buffer til en annen hva som ligger i begrepet «avsnitt». For vanlig tekst vil det normalt være tekst adskilt av blanke linjer.

Kommandoen forward-paragraph vil tilsvarende flytte markøren ett avsnitt forover.

3.4.5 Flytting på sidenivå

Når man utfører kommandoen scroll-down, vil teksten på skjermen rulle *nedover*, slik at markøren flyttes *bakover* i teksten, altså *nærmere starten*. På noen tastaturer finnes en tast merket Page up; den er bundet til denne kommandoen.

Det motsatte skjer når man utfører scroll-up; da flyttes markøren nærmere slutten av bufferen. Tasten Page down kan benyttes til dette (hvis den finnes).

3.4.6 Flytting på områdenivå

Kommandoen exchange-point-and-mark benyttes til å flytte til den andre enden av et området. Som nevnt i avsnittet *Begrepet område* på side 12 er et området er definert som teksten som ligger mellom markøren og det usynlige merket, og kommandoen bytter posisjonen til disse to. Dermed er markøren kommet til den andre enden av området, og området selv er uendret.

Kommandoen fungerer enten området er aktivt eller passivt, og området vil bli aktivt etterpå.

3.4.7 Flytting på buffernivå

Kommandoen beginning-of-buffer flytter markøren til starten av bufferen.

Tilsvarende vil kommandoen end-of-buffer flytte markøren til slutten av bufferen.

3.4.8 Flytting på vindusnivå

Flytting fra ett vindu til et annet omtales i avsnittet *Flytting mellom vinduer* på side 33.

3.4.9 Generell flytting

På de maskinene som er utstyrt med mus, er det mulig å benytte musen til å flytte til et vilkårlig sted på skjermen, uansett i hvilket vindu. Det er bare å peke med mus-markøren og så trykke på venstre tast. Det vinduet mus-markøren peker på, blir også det aktive vinduet.

3.5 Søking

Navn	K-tast	F-tast	Meny	Operasjon
isearch-backward	C-r		**Navigate**	Start inkrementell søking bakover
isearch-forward	C-s		**Navigate**	Start inkrementell søking forover

Tabell 3.5: Kommandoer for søking

3.5.1 Inkrementell søking

Kommandoen isearch-forward gir mulighet for såkalt **inkrementell søking** i teksten. Når man utfører kommandoen, vil meldingen 'I-search:' dukke opp i mini-bufferen. Trykker brukeren nå for eksempel en A vil Emacs øyeblikkelig lete etter første forekomst av 'A' eller 'a' i teksten.[2] Gir brukeren nå en X , vil Emacs lete videre etter første forekomst av 'AX' osv. Letingen avsluttes ved at brukeren trykker på Return eller en funksjonstast.

På tilsvarende måte foretas inkrementell søking bakover med kommandoen isearch-backward.

3.5.2 Kommandoer under søking

Noen taster har spesielle funksjoner under inkrementell søking:

- Delete fjerner siste tegn i søkestrengen og flytter samtidig markøren tilbake til den posisjonen den tidligere fant for den søkestrengen som gjelder nå.
- Return avslutter søkingen og lar markøren forbli der den er.
- C-r snur søkingen, slik at det nå søkes bakover.
- C-s snur også søkingen, slik at det blir søkt forover.

Tegnene C-s og C-r kan også benyttes til å finne flere forekomster av samme søkestreng. Hvis man for eksempel leter forover, vil hvert nytt trykk på C-s finne neste forekomst av søkestrengen.

Hvis man starter søkingen midt i filen og kommer til slutten og får meldingen 'Failing I-search', vil et nytt trykk på C-s fortsette søkingen fra starten av filen. Dette markeres i mini-bufferen ved at det står Wrapped først. Det tilsvarende gjelder ved søking bakover.

[2] Ved inkrementell søking skilles det normalt ikke mellom store og små bokstaver. Variabelen case-fold-search kan endres hvis man ønsker dette.

3.6 Innsetting av tekst

Navn	K-tast	F-tast	Meny	Operasjon
auto-fill-mode			**Edit**	Skift til auto-justering
quoted-insert	C-q			Sett inn spesielle tegn
insert-file	C-x i		**File**	Sett inn kopi av en fil

Tabell 3.6: Kommandoer for innsetting av tekst

Som nevnt tidligere, settes ny tekst inn i en buffer rett og slett ved å skrive den.

Vanligvis må brukeren selv sette inn linjeskift når linjen blir for lang, men ved å utføre kommandoen auto-fill-mode skifter Emacs til en modus hvor dette gjøres automatisk.[3] Kaller man auto-fill-mode en gang til, vil man gå ut av denne modusen.

Auto-fill-modusen er en *lokal* modus. Det vil si at når man setter den, gjelder dette bare for den bufferen som da er aktiv. Hver enkelt buffer kan være i denne modusen eller ikke uavhengig av de andre bufferne.

Det er mulig å sette auto-fill-modus automatisk; dette er beskrevet i avsnittet *Endring av variable* på side 44.

3.6.1 Innsetting av kontrolltegn

Kommandoen quoted-insert brukes til å legge inn i teksten tegn som vanligvis blir oppfattet som kommandoer til Emacs. Ønsker man altså å legge et slikt tegn inn i bufferen, må man først trykke Ctrl + Q og så tasten for det aktuelle tegnet, for eksempel Tab .

Hvis man skal legge inn et tegn det ikke finnes en tast for, kan man ordne dette ved først å trykke på C-q og så skrive de tre sifrene som er tegnets *oktale* representasjon i ISO 8859-1,[4] vårt standard tegnsett. Eksempelvis kan man legge inn en *à* (som har oktal kode 340), ved å taste

Ctrl + Q 3 4 0 ↩

3.6.2 Innkopiering av fil

Kommandoen insert-file kopierer innholdet av en angitt fil inn i den aktive bufferen.

[3] Man kan se at Emacs er i denne modusen ved at ordet Fill står ved siden av modusnavnet i moduslinjen.

[4] I tabell 2.1 på side 14 er en oversikt over tegnsettet. Den *oktale* koden for hvert tegn står øverst til høyre i ruten.

3.7 Fjerning av tekst

Navn	K-tast	F-tast	Meny	Operasjon
backward-delete-char-untabify		Backspace		Fjern ett tegn mot venstre; omform TAB til blanke
backward-delete-char				Fjern ett tegn mot venstre
delete-char	C-d	Delete		Fjern ett tegn mot høyre
backward-kill-word	M-DEL		**Edit**	Fjern ett ord mot venstre
kill-word	M-d		**Edit**	Fjern ett ord mot høyre
kill-line	C-k		**Edit**	Fjern resten av linjen
kill-region	C-w		**Edit**	Fjern det markerte området

Tabell 3.7: Kommandoer for fjerning

Dette avsnittet tar for seg alternative måter å fjerne tekst på, fra enkelttegn til store områder.

3.7.1 Fjerning av tegn

Kommandoen backward-delete-char-untabify fjerner tegnet til venstre for markøren. Hvis tegnet er et tabulatortegn (TAB) vil det bli omformet til det riktige antall blanke før én blank fjernes. Kommandoen er bundet opp til tasten Backspace på alle våre maskiner og terminaler.[5]

Det finnes også en lignende kommando backward-delete-char som ikke omdanner tabulatortegn, i tilfelle noen har behov for det.

Kommandoen delete-char fjerner ett tegn mot høyre, altså det tegnet som markøren peker på. Den er bundet til tasten Delete .

3.7.2 Fjerning av ord

Kommandoen kill-word fjerner neste ord i teksten; hvis markøren står midt i et ord, fjernes den delen av ordet som står etter markøren.

Tilsvarende vil kommandoen backward-kill-word fjerne det foregående ordet eller begynnelsen av et ord hvis markøren står midt i det.

3.7.3 Fjerning av linjer

Kommandoen kill-line sletter den linjen markøren står på fra og med markørens posisjon og ut. Ett trykk til vil fjerne skillet mellom linjen og den neste. Det er altså nødvendig med to trykk for å fjerne en linje hvis det står noe på den; det første blanker og det andre fjerner den tomme linjen.

[5] Ikke alle vil være enige i at Backspace og Delete skal fungere slik det er beskrevet her; se avsnittet *Modifikasjon av «Delete» og «Backspace»* på side 43 om du ønsker å endre oppsettet.

3.7.4 Fjerning av større områder

For å fjerne et helt område, må området være markert (slik det er beskrevet i avsnittet *Begrepet område* på side 12) og det må våre *aktivt*. Da vil kommandoen kill-region fjerne området.

3.8 Flytting av tekst

Navn	K-tast	F-tast	Meny	Operasjon
transpose-chars	C-t		**Edit**	Bytt om to tegn
transpose-words	M-t		**Edit**	Bytt om to ord
fill-paragraph	M-q		**Edit**	Brekk om et avsnitt
yank	C-y		**Edit**	Hent tilbake fjernet tekst
yank-pop	M-y		**Edit**	Hent tilbake tekst fjernet tidligere

Tabell 3.8: Kommandoer for flytting

Dette avsnittet tar for seg hvorledes man kan flytte tekst på ulike måter.

3.8.1 Ombytting av tegn og ord

Ofte gjør man feil og bytter om to tegn i teksten. Heldigvis er dette enkelt å rette i Emacs ved å benytte kommandoen transpose-chars som er bundet til C-t. Den lar tegnet markøren peker på, bytte plass med tegnet foran.

Når markøren er plassert sist på linjen, bytter de to foregående tegnene plass. Dette er usedvanlig nyttig når man skriver inn ny tekst for det er en vanlig feil å stokke fingrene på tastaturet. Med denne kommandoene er slike feil enkle å rette.

Man kan også bytte om rekkefølgen av to ord ved å benytte kommandoen transpose-words som er bundet til M-t.

3.8.2 Ombrekking av avsnitt

Når man arbeider med en tekst, vil den etter hvert se ganske rar ut, med noen lange og noen meget korte linjer. Da kan det lønne seg å brekke om teksten ved å benytte kommandoen fill-paragraph. Den fjerner alle linjeskift i avsnittet og setter inn nye der den synes det passer. Det er opp til hver enkelt modus å definere en algoritme for god ombrekking.

3.8.3 Generell flytting

I Emacs kan man også flytte tekst fra ett sted til et vilkårlig annet sted. Dette gjøres ved at den først fjernes. All tekst som fjernes blir lagt i en egen **klippe-buffer** (kalt «kill buffer» i Emacs).[6] Så kan teksten hentes derfra igjen ved å utføre kommandoen yank. De som har en tast merket Insert på sitt tastatur, kan bruke den.

Fremgangsmåten er altså:

1) Fjern det som skal flyttes. Flere fjernekommandoer rett etter hverandre regnes som én fjerning.

[6] Enkelttegn som fjernes blir ikke tatt vare på; dette gjelder kun større enheter, det vil si ord, linjer og hele områder.

2) Flytt markøren dit hvor teksten skal flyttes. Dette kan godt være i et annet Emacs-vindu eller et vilkårlig annet vindu.

3) Benytt kommandoen yank.

3.8.4 Tilbakehenting i flere nivåer

Klippe-bufferen er ikke en vanlig buffer, men en *ringbuffer*. Her lagres ikke bare siste fjerning, men de 10 siste. Dette gjør det mulig å hente tilbake tekst som ble fjernet lang tid tilbake.

Kommandoen som benyttes til dette heter yank-pop og den er bundet til tasten M-y. Ønsker man å hente tilbake tekst fra klippe-bufferen, er fremgangsmåten som følger:

1) Plassér markøren der teksten skal settes inn.

2) Utfør først kommandoen yank slik det er beskrevet i forrige avsnitt. Det sist fjernede område blir da satt inn.

3) Etterpå utføres en yank-pop. Den teksten som ble satt inn av yank-kommandoen blir fjernet, og neste tekst fra klippe-bufferen settes inn i stedet.

   Det er meget viktig at det ikke utføres noen andre kommandoer mellom yank og yank-pop.

4) Utfør en ny yank-pop for å få tak i neste nivå i klippe-bufferen. Hver gang forsvinner den siste teksten man satte inn, og teksten i neste nivå kommer i stedet.

3.8.5 Kopiering av tekst

Siden det er mulig å hente frem teksten i klippe-bufferen flere ganger, gir det en grei mulighet til å kopiere tekst. Kopieringen skjer altså ved at teksten først fjernes og så settes inn de stedene den skal stå.

3.9 Utskifting av tekst

Navn	K-tast	F-tast	Meny	Operasjon
query-replace	M-%	F5	**Edit**	Skift en tekst ut med en annen

Tabell 3.9: Kommandoer for utskifting

Kommandoen query-replace brukes til systematisk utskifting av tekst. Den er bundet opp til M-% og dessuten til tasten F5 .

Kommandoen spør først etter en tekst den skal lete etter og deretter hva denne teksten skal byttes ut med; begge tekstene avsluttes med RET. Så vil den gå gjennom bufferen og stoppe og spørre brukeren for hver forekomst den finner. Brukeren kan da gi flere forskjellige svar; de viktigste er:

- ? gir en oversikt over alle mulige svar.
- (blank) foretar en utskifting.
- Del foretar ingen utskifting.
- . (punktum) foretar én utskifting og avslutter så.
- Esc avbryter utskiftingen.
- C-r starter en rekursiv redigering, slik det står omtalt i neste avsnitt.
- ! foretar utskifting i resten av bufferen uten å spørre brukeren flere ganger.

3.9.1 Rekursiv redigering

Når man går gjennom en fil og foretar systematisk utskifting av tekst, hender det ofte at man oppdager andre ting man ønsker å rette på. I Emacs kan man da starte en **rekursiv redigering**; man kan foreta de rettelsene man vil, og så fortsette utskiftingen der man slapp.

En rekursiv redigering startes med å gi en C-r når Emacs ber om en utskiftingskommando. Emacs markerer da at den er inne i en rekursiv modus ved å sette modusnavnet på statuslinjen i klammer ([. . .]). Det er mulig å gå inn i så mange rekursive nivåer man vil.

Når vi er ferdig med den rekursive redigeringen og ønsker å gå videre med utskiftingen, benyttes kommandoen exit-recursive-edit som er bundet til C-M-c.

3.10 Klipping og liming i vindussystemet

Når man benytter en maskin som kjører et vindussystemet (enten det er X eller Windows), har man et par ekstra muligheter til å flytte tekst mellom de forskjellige vinduene på skjermen. Denne muligheten benytter vindussystemets funksjoner for slik «klipping og liming», og det er for eksempel mulig å flytte tekst fra ett Emacs-vindu til et annet, eller fra et kommandovindu til et Emacs-vindu og omvendt.

All tekst som fjernes i Emacs, legges som tidligere nevnt i klippebufferen til Emacs. Det øverste nivået i denne bufferen, altså det som sist ble fjernet, deles med alle vinduer på skjermen.

For eksempel kan man flytte tekst fra et Emacs-vindu til et kommandovindu ved å gjøre følgende:

1) Fjern teksten fra Emacs-bufferen ved å bruke kill-line, kill-word eller tilsvarende.
2) Flytt mus-markøren til kommandovinduet og trykk på midtre mus-tast (om du kjører Unix) eller velg Paste/Lim inn om du kjører Windows.

3.10.1 Klipping i Emacs med musen

Det er mulig å benytte musen til å klippe noe fra Emacs-bufferen. Markér først et område, for eksempel ved å dra musmarkøren over området med venstre mus-tast trykket ned (slik det står beskrevet i avsnittet *Begrepet område* på side 12). Da er området kopiert over i klippe-bufferen (selv om det ikke er fjernet), og kan limes inn i en annen buffer eller et annet vindu på skjermen.

Klippingen nevnt i forrige avsnitt fjerner som sagt ikke den teksten som står i det markerte området. Ønsker man det, kan man **dobbelt-klikke**[7] med høyre mus-tast; da blir teksten fjernet etter å ha blitt kopiert over i klippe-bufferen.

3.10.2 Liming i Emacs med musen

Man kan godt bruke musen til å sette inn tekst i en Emacs-buffer, enten det er snakk om tekst som tidligere er fjernet med Emacs-kommandoer eller klippet vekk med musen i det samme vinduet eller et annet vindu. Det er bare å plassere Emacs-markøren der man vil ha teksten, for eksempel ved å peke på posisjonen og så klikke med *venstre* mus-tast, og deretter klikke med *midtre* mus-tast (om du kjører Unix) eller velge Paste/Lim inn om du kjører Windows.

Det er en vanlig feil å glemme å flytte Emacs-markøren før man setter inn teksten. Det er viktig å huske at teksten plasseres der Emacs-markøren angir, ikke der mus-markøren peker.[8]

---

[7] Å dobbelt-klikke vil si å trykke raskt to ganger på en mus-tast.

[8] Det er en Emacs-variabel ved navn mouse-yank-at-point som angir om tekst skal settes inn i henhold til Emacs-markøren eller mus-markøren. Vår standardoppsett angir det første.

3.11 Bruk av flere vinduer

Navn	K-tast	F-tast	Meny	Operasjon
split-window-vertically	C-x 2	F12	**Misc**	Splitt Emacs-vinduet i to
split-window-horizontally	C-x 3		**Misc**	Splitt Emacs-vinduet vertikalt i to
delete-window	C-x 0		**Misc**	Fjern vinduet
delete-other-windows	C-x 1	F11	**Misc**	La kun ett vindu være igjen
find-file	C-x C-f	F3	**File**	Hent inn en fil
find-file-other-window	C-x 4 C-f		**File**	Hent inn filen i et annet vindu
other-window	C-x o			Flytt markøren til et annet vindu

Tabell 3.10: Kommandoer for flere vinduer

Som tidligere nevnt, er det mulig å splitte Emacs-skjermbildet i flere vinduer. Dette beskrives nærmere i dette avsnittet.

3.11.1 Splitting av vinduer

Kommandoen split-window-vertically vil dele det aktive vinduet i to vinduer over hverandre. Begge vinduene vil skue inn i samme buffer.

Av og til kan man ønske å ha to vinduer ved siden av hverandre i stedet for over hverandre. Dette er mulig å oppnå ved å benytte kommandoen split-window-horizontally.

3.11.2 Fjerning av vinduer

Kommandoen delete-window vil fjerne det aktive vinduet. Det å fjerne et vindu betyr ikke at innholdet ødelegges på noen måte; det innebærer bare at det ikke vises for øyeblikket.

Kommandoen delete-other-windows vil fjerne alle vinduer unntatt det aktive fra skjermen.

Hvis maskinen er utstyrt med mus, kan man også benytte musen til å fjerne vinduer. Pek på statuslinjen under et vinduet og klikk med høyre mus-tast for å fjerne akkurat det vinduet, eller med midtre mus-tast for å fjerne alle andre vinduer.

3.11.3 Innlasting av filer

Kommandoen find-file vil hente frem den oppgitte filen, lese den inn i en buffer og la det aktive vinduet skue inn i den bufferen. Hvis filen tidligere er blitt lest inn i en buffer, vil ikke filen bli lest en gang til; den første bufferen benyttes i stedet.

Kommandoen find-file-other-window vil gjøre akkurat det samme som find-file, men den vil la et annet vindu enn det aktive se inn i bufferen. Hvis det kun er ett vindu på skjermen, vil kommandoen splitte vinduet først.[9]

3.11.4 Flytting mellom vinduer

Kommandoen other-window vil velge et nytt aktivt vindu på skjermen. Ved å benytte kommandoen flere ganger vil alle vinduene på skjermen bli aktive etter tur, også mini-bufferen (se avsnitt 2.3.1.1 på side 11) om det er noe tekst der.

De maskinene som er utstyrt med mus, kan benytte venstre mus-tast på musen til å velge aktivt vindu. Det er bare å peke på statuslinjen til det aktuelle vinduet og klikke med venstre mus-tast.

[9] For å være helt nøyaktig: Kommandoen vil splitte vinduet hvis det inneholder flere linjer enn angitt av variabelen split-height-threshold, som normalt er 12. Mange, deriblant jeg selv, liker ikke at skjermvinduet splittes i for mange små vinduer, så jeg har endret denne variabelen til 40. Hvorledes dette gjøres er vist i avsnittet *Endring av variable* på side 44.

3.12 Bruk av rammer

Navn	K-tast	F-tast	Meny	Operasjon
make-frame				Lag en ny ramme
find-file-other-frame	C-x 5 f		**File**	Les inn fil i ny ramme
delete-frame	C-x 5 0			Fjern rammen

Tabell 3.11: Kommandoer for rammer

Som forklart i avsnittet *Begrepet ramme* på side 11, kan man benytte flere rammer når man skal se på Emacs-bufferne. Kommandoen make-frame lager en ny ramme og viser et nytt vindu mot den aktive bufferen.

Kommandoen find-file-other-frame fungerer akkurat som find-file-other-window (se avsnittet *Innlasting av filer* på side 32) men lager en ny ramme for å vise vinduet mot filen.

Hvis man ikke trenger en ramme lenger, kan man bare ikonifisere den. Alternativt kan man fjerne den for godt ved å utføre delete-frame.

3.13 Kompilering

Navn	K-tast	F-tast	Meny	Operasjon
ifi-compile		F8	**Misc**	Kompilér programmet
next-error	C-x ‘	F9	**Misc**	Finn neste feilmelding

Tabell 3.12: Kommandoer for kompilering

Det er mulig å styre kompileringen av for eksempel Java-programmer og LaTeX-filer fra Emacs. Dette gjør det raskere å starte kompileringen, man er sikker på at filene blir skrevet på disken hvis de er endret, og man kan benytte mulighetene Emacs har til å vise feilmeldinger sammen med tilsvarende linje i kildekoden.

Kommandoen ifi-compile kan benyttes når programmer skal kompileres. Denne kommandoen er utviklet ved Ifi og er en variant av standardkommandoen compile. Forskjellen er at ifi-compile vil forsøke å gi et bedre forslag til kompileringskommando basert på vår bruk av maskinene.

Når ifi-compile utføres, spør den først etter hvilken kommando som skal benyttes ved kompileringen; brukeren kan fritt redigere forslaget fra Emacs. Så vil kommandoen opprette et eget vindu hvor kompileringen foregår.

3.13.1 Fremvisning av feilmeldinger

Hvis kompilering ga feilmeldinger, kan brukeren benytte den spesielle kommandoen next-error til å finne ut hvor feilene er. Kommandoen vil vise neste feilmelding i ett vindu og den aktuelle programlinjen i kildekoden i et annet.

3.14 Buffer-operasjoner

Navn	K-tast	F-tast	Meny	Operasjon
list-buffers	C-x C-b		**Buffers**	Vise alle buffere som er i bruk
ps-print-buffer			**File**	Skriv bufferen på en skriver
ps-print-buffer-with-faces			**File**	Skriv bufferen på fargeskriver

Tabell 3.13: Kommandoer for buffere

Kommandoen list-buffers vil lage et vindu på skjermen som inneholder navnene på alle buffere i bruk og deres tilhørende filnavn. Brukeren kan så velge hvilken buffer han eller hun vil se på ved å flytte markøren til linjen med den aktuelle bufferen. Mulige kommandoer er da:

- ? vil gi en oversikt over alle kommandoer det er mulig å gi.
- F vil hente frem den aktuelle bufferen.
- O vil plassere den aktuelle bufferen i et annet vindu enn der hvor buffer-oversikten står.

Hvis man kjører på en maskin som er utstyrt med mus, er det mulig å benytte musen til å velge en buffer. Menyen **Buffers** inneholder et utvalg av de eksisterende buffere, og det er bare å velge derfra.

3.14.1 Utskrift til skriver

Kommandoene ps-print-buffer og ps-print-buffer-with-faces skriver innholdet av den aktive bufferen ut på en skriver; førstnevnte skriver i sort-hvitt mens den andre skriver i farger. På en Unix-maskin brukes alltid standardskriveren som er definert med omgivelsesvariabelen PRINTER; under Windows brukes samme skriver som sist gang noe ble skrevet ut fra programmet gsview32.

3.15 Andre kommandoer

Navn	K-tast	F-tast	Meny	Operasjon
undo	C-/	F7	**Edit**	Reparér det siste som ble gjort
recenter	C-l		**Misc**	Frisk opp skjermen
capitalize-word	M-c		**Edit**	Skift til stor forbokstav
downcase-word	M-l		**Edit**	Skift til små bokstaver
upcase-word	M-u		**Edit**	Skift til store bokstaver
ifi-change-font			**Misc**	Skift typesnitt
ispell-word	M-$		**Misc→ispell**	Sjekk stavingen av et ord
ispell-region			**Misc→ispell**	Sjekk det aktive området
ispell-buffer			**Misc→ispell**	Sjekk hele bufferen
ispell-kill-ispell			**Misc→ispell**	Avslutt ispell-prosessen

Tabell 3.14: Andre kommandoer

3.15.1 Angring

Dette er en av de alle nyttigste kommandoene i Emacs! Hvis man gjør et eller annet man angrer etterpå, kan dette rettes opp ved hjelp av kommandoen undo. Hver utførelse av denne kommandoen retter opp igjen siste kommando som ble foretatt. En bruker kan gjøre dette så mange ganger han eller hun vil.

Det er bare én ting å passe på når man bruker angretasten. For å spare plass i lageret blir enkelte kommandosekvenser slått sammen til én kommando; dette gjelder særlig fjerning og innsetting av tekst. Følgelig må man passe nøye på slik at man ikke plutselig har angret mer enn man ønsket.

Hvis man plutselig angrer på angringen, er det likevel en mulighet til å gjenopprette det tapte. Skriv først bufferen ut på filen sin (se avsnittet *Å gjemme unna buffere* på side 20). Derefter vil bruk av undo-kommandoen rette opp de tidligere angringene!

3.15.2 Oppfrisking av skjermen

Kommandoen recenter brukes til to ting:

- Den vil rulle vinduet slik at den linjen som markøren er i, vil bli plassert midt i vinduet. Dette er nyttig når man ønsker å se flere linjer rundt der man for øyeblikket arbeider.
- Den vil tegne opp alle vinduene på nytt. Dette er noen ganger nødvendig hvis det er kommet grums der.

3.15.3 Omforming til store og små bokstaver

Overaskende ofte trenger man å omforme bokstavene i et ord til små eller store bokstaver. I Emacs har vi tre kommandoer til dette:

- capitalize-word (bundet til M-c) omformer den første bokstaven i et ord til en store bokstav, og de etterfølgende bokstavene i samme ord til små bokstaver. Hvis markøren står midt i et ord når man utfører kommandoen, regnes ordet å starte der.
- downcase-word (bundet til M-d) vil omforme alle bokstavene i ordet til små bokstaver. Også her gjelder at hvis markøren står midt i et ord, regnes ordet å starte der.
- upcase-word (bundet til M-u) vil tilsvarende omforme til store bokstaver.

3.15.4 Skifte av tegnsnitt

Kommandoen ifi-change-font vil forandre skriftsnittet som benyttes av Emacs. Det kommer da frem en meny som brukeren kan velge i; alternativene i menyen vil avhenge av hva slags skjerm man sitter ved.

3.15.5 Stavesjekking

Emacs har innebygget et grensesnitt mot programmet ispell som kan sjekke om ord er riktig stavet. Kommandoen ispell-word benyttes til å sjekke enkeltord. Det vil sjekke det ordet markøren peker på, og hvis ispell tror det er galt stavet, vil det komme med forslag om riktig staving. Disse forslagene kommer øverst på skjermen, og ved å taste tegnet angitt i parentes foran alternativet, blir ordet byttet ut med dette alternativet.

Kommandoen ispell-region vil gå gjennom hele det markerte området (hvis det er aktivt) og sjekke alle ordene der.

Kommandoen ispell-buffer benyttes når man vil sjekke et helt dokument.

Hvis bufferens navn slutter på .tex, vet ispell at den inneholder TEX eller LATEX-kode, og den vil overse strukturer som har TEX og LATEX å gjøre.

3.15.5.1 Valg av språk

Vår utgave av Emacs og ispell kan sjekke tekst på mange ulike språk; disse velges i menyen **Misc→ispell**. Før man skifter språk, må man stoppe eventuelle ispell-prosesser som går med **Misc→ispell→Kill Process**.

Når nøden er størst, er hjelpen nærmest.
— Gammelt norsk ordtak

Kapittel 4

Hjelp fra Emacs

Bruk av hjelpefunksjonene skjer gjennom tasten F1 eller Control + H . Deretter gis et tastetrykk som angir hva slags hjelp man ønsker; de forskjellige mulighetene er:

- ? gir en oversikt à la denne over alle hjelpefunksjonene.
- A gir mulighet til å spørre etter kommandoer. Man oppgir et ord, og systemet vil da lete etter kommandoer hvis navn inneholder dette ordet.

  Hvis man for eksempel ønsker å få oversikt over hvilke kommandoer som kan brukes til å fjerne ting, skriver man

  F1 A kill Return

- B gir en liste over alle kontrolltastene i bruk og hvilke kommandoer de er koblet til.
- C gir opplysning om hvilken kommando som er knyttet til en angitt tast eller tastesekvens. Ønsker man for eksempel å vite hvilken kommando tastene C-x s er bundet opp til, skriver man

  F1 C Ctrl + X S

  og får svaret

  ```
  C-x s runs the command save-some-buffers
  ```

- Shift + C gir en oversikt over et tegnsett, vanligvis det man for tiden bruker.
- F gir en full beskrivelse av en kommando. Den bruker fra én linje til flere hundre linjer på å fortelle alt brukeren trenger å vite om kommandoen. Se også K -hjelpen.
- I gir adgang til et generelt informasjonssystem som gir oversikt over flere programmer, blant annet Emacs selv. Her ved Ifi har vi imidlertid satset mer på andre kommunikasjonskanaler enn denne, så kun få av våre programmer har dokumentasjon i dette informasjonssystemet.
- K gir den samme informasjonen som F -hjelpen, men her angis en kontroll- eller spesialtast istedenfor navnet på kommandoen.

- M gir opplysninger om den modus bufferen er i. Spesielt vil den nevne alle kommandoer som kun finnes for den aktuelle modusen.
- T starter et innføringskurs i Emacs. Dette anbefales sterkt for nybegynnere.
- W gir den inverse informasjonen av C -hjelpen. Her oppgir brukeren et kommandonavn, og Emacs angir hvilke taster som er koblet til den kommandoen. Hvis Emacs sier «... is not on any keys.», kan kommandoen kun kalles ved å bruke M-x; se avsnittet *Kommandonavn* på side 10.

  Når brukeren spør om kommandoer som er koblet til spesialtastene, svarer Emacs med navnet på tasten. Emacs kjenner til alle de navnene som er nevnt i tabell 5.2 på side 44.

Lad Veien gaa, hvorhen den vil…
— Henrik Ibsen

Kapittel 5

Veien videre

Dette kapittelet er for de som har kommet seg over den første startfasen og gjerne vil vite litt mer om dette omfattende produktet.

5.1 Flere nyttige ting å vite

Navn	K-tast	F-tast	Meny	Operasjon
recover-file				Gjenopprett fil i henhold til sikkerhetskopi
revert-buffer				Les fil inn i buffer på nytt
kill-this-buffer			**File**	Fjern en buffer fullstendig
universal-argument	C-u			Angi tallparameter til kommando
global-set-key				Definér tast

Tabell 5.1: Kommandoer for viderekomne

5.1.1 Sikkerhetskopier

For å hindre at filer blir ødelagt hvis maskinen skulle stoppe opp, sørger Emacs for at det automatisk tas sikkerhetskopier.

5.1.1.1 Sikring av forrige versjon

Som nevnt tidligere foretas all redigering på buffere i maskinens hurtiglager, og disse skrives kun ut på fil når brukeren eksplisitt ber om det. Når filene skrives ut, sørger Emacs for at forrige versjon av filen blir tatt vare på. Dette gjøres ved at forrige versjon omnavnes, nærmere bestemt ved at det legges en ~ bak navnet.

For å hindre at slike sikkerhetskopier fyller opp de allerede ganske fulle lagringsdiskene ved instituttet, vil slike filer bli automatisk fjernet når de er tre dager gamle.

5.1.1.2 Sikkerhetskopier under redigeringen

For å hindre at timelangt arbeid med redigering skal gå tapt hvis strømmen går, tar Emacs dessuten sikkerhetskopier med jevne mellomrom, nærmere bestemt hver gang brukeren har foretatt 300 tastetrykk eller det er gått 5 minutter. Sikkerhetskopien får filens navn med en # foran og bak. Når redigeringen avsluttes, blir denne sikkerhetskopien fjernet.

Når man starter redigeringen igjen etter en maskinfeil, vil Emacs selv sjekke hvilke sikkerhetskopier som finnes. Hvis man prøver å redigere en ødelagt fil, det vil si en fil hvor det finnes en sikkerhetskopi, vil Emacs melde fra. Brukeren får da valget mellom å gjenopprette filen i hehold til sikkerhetskopien eller la det være. For å gjenopprette filen, må brukeren utføre kommandoen recover-file. Dette gjøres altså ikke automatisk.

5.1.2 Endring av filer

Av og til kan det skje at en fil man redigerer i Emacs blir endret av et annet program. Dette kan skyldes at flere brukere arbeider med samme fil, at brukeren har flere utgaver av Emacs gående samtidig, eller at et program genererer en ny utgave av filen.

Emacs sjekker om slikt skjer, og hvis den oppdager en slik endring gir den meldingen

```
File has changed on disk; really want to edit the buffer?
```

Hvis man svarer y (for *ja*), kan man få redigere det innholdet man har i bufferen, og ikke (i denne omgang) ta hensyn til at filen er endret.

Når man så skal gjemme unna filen, forteller imidlertid Emacs at

```
Disk file has changed since visited or saved. Save anyway?
```

Svarer man y, vil bufferen bli skrevet ut, og endringene som tidligere var gjort på filen (av det andre programmet) går tapt. Velger man i stedet å svare n, mister man de endringene man selv har gjort i Emacs-bufferen.

5.1.2.1 Oppdatering av buffer

Hvis man oppdager at en fil er endret av et annet program, kan det hende man ønsker å oppdatere Emacs-bufferen slik at den er i samsvar med filen. Til dette benyttes kommandoen revert-buffer.

5.1.3 Fjerning av buffere

Emacs arbeider som tidligere nevnt med buffere, hvor hver buffer inneholder hele den opprinnelige filen med de endringer som er foretatt. Alt dette lagres i hurtiglageret, og etter hvert kan dette ta ganske mye plass. Da kan det være aktuelt å fjerne noen buffere man ikke lenger trenger. Til dette benyttes kommandoen kill-this-buffer.

Plassmangel i Emacs er imidlertid sjelden noe problem. Det er plass til adskillige megabyte i våre maskiner, så hvis man logger ut hver dag (som alle fornuftige brukere bør gjøre) og starter med en ny, frisk utgave av Emacs hver morgen, vil man neppe noensinne merke at maskinen går full.

5.1.4 Parametre til kommandoer

Alle kommandoer kan gis en numerisk parameter; denne angir vanligvis hvor mange ganger kommandoen skal utføres. Parameteren angis med et kall på kommandoen universal-argument (som også kan kalles ved å trykke på tasten C-u) fulgt av selve parameteren. Ønsker man for eksempel å flytte 217 linjer frem gjøres dette enkelt ved å skrive

Ctrl + U 2 1 7 ↓

5.1.5 Automatisk fullføring av navn

Emacs har innebygget en mulighet for å fullføre navn der den er i stand til det; dette gjelder både navn på filer og kommandoer. Når brukeren tror han eller hun har skrevet nok av navnet til at det er entydig, kan vedkommende trykke på tasten Tab . Da vil Emacs fylle ut resten av navnet så langt den klarer. Hvis den oppgitte delen av navnet er flertydig, vil Emacs presentere en liste over alle mulighetene; da kan man velge blant disse ved å peke med musen og klikke midtre mus-tast.

For å få utført kommandoen recover-file er det for eksempel nok å skrive

Meta + X R E C O Tab Return

Man kan også droppe Tab foran en Return hvis det man har oppgitt er entydig.

5.1.6 Initieringsfiler

Når Emacs startes opp, vil den aller først lese filen en initieringsfil hos den enkelte bruker; på et Unix-system heter denne filen `~/.emacs`, mens på en Windows-maskin benyttes `C:\_emacs`. Denne filen inneholder Lisp-kode som vil bli utført ved hver oppstart. Brukeren kan endre Emacs-oppsettet ved å modifisere denne filen.

Initieringsfilen er godt kommentert, så ved å hente den inn i Emacs burde det være lett å se hvilke muligheter man har. Unngå å gjøre andre endringer i starten av filen enn de som er nevnt i kommentarene; legg heller endringer på slutten av filen.

5.1.6.1 Modifikasjon av «Delete» og «Backspace»

Ikke alle liker den innebygde bindingen av de to tastene Delete og Backspace . Dette kan man endre ved å redigere initieringsfilen. De aktuelle linjene er disse:

```
; Uncomment the line that specifies how you want the Backspace key
; (usually marked with '<--') and Delete key (sometimes marked with
; 'Del') should work:
;   (setq ifi-setup-del-bs 'll) ; Both keys delete to the left
;   (setq ifi-setup-del-bs 'lr) ; DEL deletes to the left, BS to the right
    (setq ifi-setup-del-bs 'rl) ; DEL deletes to the right, BS to the left
;   (setq ifi-setup-del-bs 'rr) ; Both keys delete to the right
```

Ved å sette inn et kommentategn («;») foran nest siste linje og fjerne et annet, kan man velge nøyaktig hvilken kombinmasjon man ønsker.

backspace	begin	break	cancel	clearline	delete
deletechar	deleteline	down	end	execute	f1
f2	f3	f4	f5	f6	f7
f8	f9	f10	f11	f12	f13
f14	f15	f16	f17	f18	f19
f20	f21	f22	f23	f24	f25
f26	f27	f28	f29	f30	f31
f32	f33	f34	f35	find	help
insert	insertchar	insertline	kp-0	kp-1	kp-2
kp-3	kp-4	kp-5	kp-6	kp-7	kp-8
kp-9	kp-add	kp-backtab	kp-decimal	kp-divide	kp-enter
kp-equal	kp-f1	kp-f2	kp-f3	kp-f4	kp-multiply
kp-numlock	kp-separator	kp-space	kp-subtract	kp-tab	left
menu	next	print	prior	redo	reset
right	select	system	undo	up	user

Tabell 5.2: Oversikt over spesialtaster

5.1.6.2 Binding av spesialtaster

Mange brukere vil etterhvert ønske å binde kommandoer til de ubenyttede kontroll- eller funksjonstastene. Dette gjøres ved å legge kall på global-set-key sist i initieringsfilen.

Ønsker for eksempel en bruker å binde kommandoen other-window til tasten F10 gjøres dette ved å legge linjen

```
(global-set-key  [f10]  'other-window)
```

sist i filen. I tabell 5.2 står navnene på alle de tastene Emacs kjenner til.[1] (Legg merke til den lille «'»-en foran kommandonavnet; den *må* være der.)

Det er mulig å angi at en tast skal kombineres med Shift eller Ctrl; dette gjøres ved å sette S- eller C- foran navnet. Hvis jeg for eksempel ønsker å binde find-file-other-window til Shift + F3, kan jeg legge inn følgende linje til slutt i initieringsfilen min:

```
(global-set-key [S-f3] 'find-file-other-window)
```

Hvis man lurer på hva en tast kalles av Emacs, kan man lett finne det ut ved å spørre Emacs om hva som er bundet til den. Hvis jeg for eksempel lurer på hva høyre piltast kalles, kan jeg spørre

F1 C →

og får da svaret right.

5.1.6.3 Endring av variable

En annen side av tilpasningen er å endre variable som styrer oppsettet; dette gjøres med kommandoen setq. Ønsker man for eksempel å hindre at skjermvinduet splittes i for mange små vinduer (se fotnote 9 på side 33), kan man legge følgende linje i initieringsfilen:

[1] De tastene som starter med kp- befinner seg på det spesielle numeriske tastaturet («key pad») som står for seg selv til høyre.

```
(setq split-height-threshold 40)
```

En annen ting man kan ønske å endre, er å få såkalt **auto-fill** i alle tekstfiler. Dette innebærer at det automatisk settes inn linjeskift når man skriver utenfor en viss posisjon[2] på linjen, men bare i tekstfiler, som e-post-brev og filer hvis navn slutter på .txt eller .tex. Den aktuelle linjen i initieringslinjen skal se slik ut:

```
(add-hook 'text-mode-hook 'turn-on-auto-fill)
```

5.1.6.4 Egne kommandoer

Emacs er programmerbar, så hvis man kan programmere i LISP, kan man lage sine egne Emacs-kommandoer og legge dem i initieringsfilen. Selv har jeg laget en funksjon som skjøter den linjen der markøren befinner seg med neste linje:

```
(defun my-join-to-next-line ()
  "Join this line to the next and fix whitespace at joint."
  (interactive "*")
  (delete-indentation t))
```

Etter dette er kommandoen my-join-to-next-line klar til bruk og kan kalles med M-x. Hvis jeg vil knytte den til en tast, kan jeg gjøre det med global-set-key slik det ble beskrevet i avsnittet *Binding av spesialtaster* på forrige side.

[2] Initielt er denne posisjonen satt til å være 70, men den kan endres ved å modifisere variabelen fill-column. Skriv

FT V fill-column

for mer informasjon om denne variabelen.

5.2 Ting av mer variabel nytte

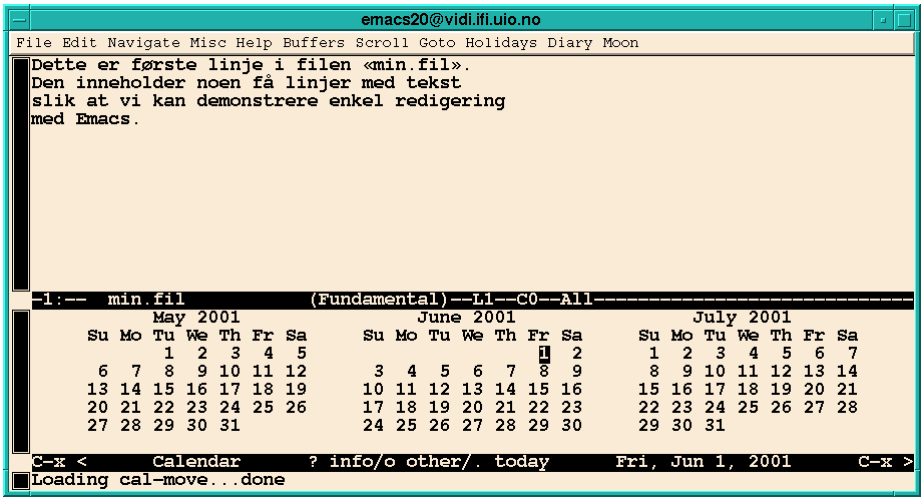
Navn	K-tast	F-tast	Meny	Operasjon
calendar				Vis en kalender
hanoi				Start en visuell løsning av problemet med Hanois tårn
gomoku				Spill bondesjakk
doctor				Start en sesjon hos psykologen

Tabell 5.3: Kommandoer av variabel nytte

Emacs er som tidligere nevnt programmerbar, og mange mennesker verden rundt har brukt mye tid på å utvide Emacs med diverse nyttige og unyttige rariteter.

5.2.1 Kalender

Emacs er utstyrt med en ganske avansert kalender. Ved å utføre kommandoen calendar får man frem en kalender med forrige, denne og neste måned nederst i vinduet, som vist i figur 5.1.



Figur 5.1: En kalender

I kalenderviduet finnes mange nyttige kommandoer som man kan få en oversikt over ved å taste F1 m. Noen interessante er disse:

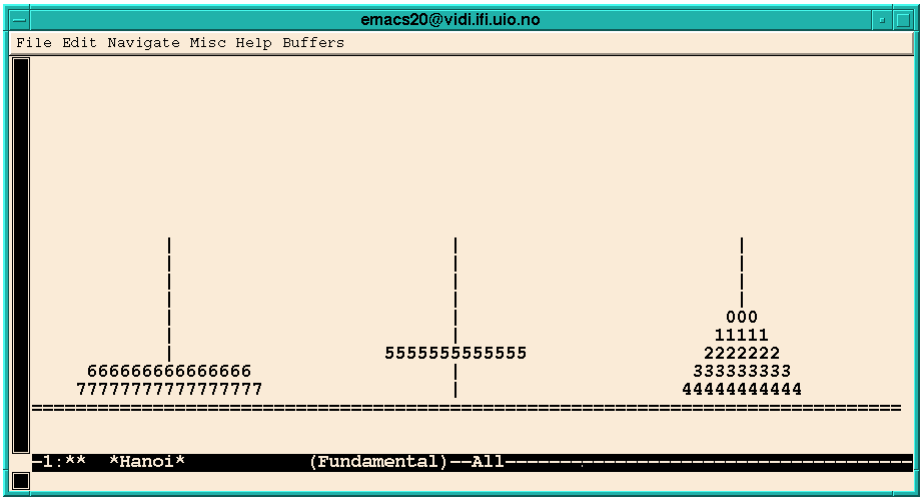
X Vis alle høytidsdager i de tre månedene. Kommandoen holidays vil åpne et vindu som forklarer hva de er.

Shift + S Fortell om soloppgang og solnedgang i dag:

```
Jul 25, 2001: Sunrise 4:44am (MET DST),
    sunset 10:02pm (MET DST) at Blindern (17:17 hours daylight)
```

Shift + M viser en oversikt over månefasene i de tre månedene.

P F Vis datoen i den kalenderen som ble innført under den franske revolusjon i 1789.



Figur 5.2: Hanois tårn med 8 ringer

Man kan også lage en «7. sans» over avtaler, og denne kan kobles opp mot kalenderen. Les mer om dette ved å taste F1 m.

5.2.2 En demonstrasjon av Hanois tårn

Hanois tårn er et spill som går ut på å flytte diverse ringer fra én pinne til en annen ved hjelp av en tredje pinne. Underveis er det aldri lov å legge en ring oppå en som er mindre.

Dette spillet er bygget inn som «tegnefilm» i Emacs. Ved å utføre

C-q 8 M-x hanoi

får man se en 8-rings versjon av Hanois tårn bli løst på skjermen; en posisjon underveis mot løsningen er vist i figur 5.2.

5.2.3 Bondesjakk

Hvis du trenger en pause, kan Emacs tilby et slag bondesjakk, som kalles **Go moku** i resten av verden. Kommandoen gomoku starter spillet som er selvforklarende.

5.2.4 Eliza hos psykologen

Kommandoen doctor starter en seanse hos psykologen. Her kan man fortelle (på engelsk) om sine problemer, bli utspurt om disse, og kanskje få hjelp. Et slikt program kalles ofte **Eliza** etter det første av denne typen. Starten på en konsultasjon er vist i figur 5.3 på neste side; mine utsagn er vist i kursiv. Som vi kan se, varierer legens svar fra det tilsynelatende intelligente til det skjære ugrammatikalske nonsens.

I am the psychotherapist. Please, describe your problems. Each time you are finished talking, type RET twice.

I am very unhappy.

Are you unhappy often?

Yes, every weekend when I visit my father and mother.

Tell me something about your family.

I had a very happy childhood.

Do you get happy often?

Only on Saturdays when I watch the football match on television.

What happened when you watch the football match on television?

I bet on the results and hope to win a lot of money.

Why do you say and might want to win?

Figur 5.3: En sesjon med doctor

Register