1 Inngangur.

Í eftirfarandi riti verður gerð grein fyrir undirstöðuatriðum Unix-stýrikerfisins. Talsverð áhersla verður lögð á almenn Unix-forrit ásamt nokkrum viðbótum sem hafa verið þróaðar til almennrar dreifingar. Öllum viðbótum er lýst að fullu í handbókinni og þær eru einnig fáanlegar um Unix-tölvunetið (án endurgjalds).

1.1 Gangsetning.

Til að notandi geti strax hafist handa við einfalda ritvinnslu skal hér bent á nauðsynlegar skipanir til að ritfæra texta og prenta. Í eftirfarandi dæmum er feitletrað það sem notandinn ritar en viðbrögð tölvunnar eru með venjulegu letri.

- login: **verknúmer** Notandi ritar verknúmer og aðgangsorð
  password: **lykilorð** (ath. aðgangsorð birtist ekki á skjánum).
- Vélin birtir boð, t.d. "Velkomin til starfa". Hugsanlega birtast einnig ýmsar aðrar upplýsingar, um opnunartíma, um að tölvupóstur hafi borist notanda o.s.frv.
- %-merkið er kvaðning stýrikerfisins og merkir að skipanatúlkur vélar er tilbúinn að taka við skipunum frá notanda. Hugsanlegt er að kvaðningin sé táknuð með $ í stað %.
- **emacs skrá**
  Rita **emacs, stafbil** og **heiti** á þeim texta sem á að skrifa eða breyta. **emacs** er heiti ritþórsins.
- Síðan ritar notandinn þann texta sem hann hefur hug á.
- Nota má bendilhnappa eða eftirfarandi skipanir til að flytja sig til í textanum:

^P
upp um eina línu
"previous line"

^B að fyrri staf "backward character"

^F að næsta staf "forward character"

^N
niður um eina línu
"next line"

[ath. ^F þýðir að "control"-lykli er haldið niðri meðan stutt er á F].

- Skránni forðað: **^X^M**, **^X^S** eða **^XS** (misjafnt eftir kerfum og eftir því hvaða útgáfa af emacs er í notkun).
- Keyrslu emacs er hætt með því að styðja á: **^X^C**
- Skráin er prentuð með eftirfarandi skipunum:
  pr skrá | lp
  eða
  nroff -mm skrá | lp

Í 4. kafla er greint frá næstu skrefum í ritvinnslu. Í 8. og 12. kafla eru ræki legri upplýsingar.

1.2 Hvað er Unix?

Unix er stýrikerfi ásamt safni hjálparforrita. Stýrikerfið og flest hjálparforritin voru þróuð af starfsmönnum Bandaríska símafyrirtækisins AT&T en mörgu hefur verið bætt við af starfsmönnum háskólans í Berkeley.

Unix-stýrikerfið þótti byltingarkennt á sínum tíma og fjölmörg önnur stýrikerfi hafa sótt margar hugmyndir til þess (MS-DOS svo dæmi sé tekið).

Nokkur einkenni Unix eru þessi:

- Forrit eru lítil. Hverju forriti er yfirleitt ætlað að leysa eitt afmarkað verkefni. Flest forrit eru skrifuð þannig að þau lesi af lyklaborði. Flóknari kerfi eru síðan smíðuð með því að tengja saman þessi einföldu forrit.
- Auðvelt er að endurskilgreina inntak og úttak. Hægt er að láta forrit, sem að öðru jöfnu les af lyklaborði og skrifar á skjá, lesa úr skrá eða skrifa í skrá. Einnig getur úttak eins forrits orðið inntak þess næsta, t.d.

```
forrit1 <gögn | forrit2 >niðurst.
```

Hér er inntak forrits1 sótt í skrána gögn en úttaki forrits1 síðan beint um svonefnda pípu að forriti2. Loks er úttak forrits2 skrifað í skrána niðurst.

- Fjöldi forrita fylgir UNIX. Mörg hundruð forrit fylgja kerfinu, m.a. fyrir gagnagrunns- og textavinnslu, forritaþróun, vélasamskipti og tölvupóstsendingar.

- Sveigjanleiki er mikill. Einstaklega auðvelt er að aðlaga kerfið að eigin óskum. Til dæmis má safna skipunum í skrá og gefa síðan skipunina

chmod +**x** *skrá*

og er 'skrá' þá orðin að forriti sem unnt er að keyra með því einfaldlega að rita '**skrá**' í skipanalínuna. Í skránni mega vera allar þær skipanir sem hægt er að gefa af lyklaborðinu.

- Litið er á tæki sem tengjast kerfinu sem venjulegar skrár.

Uppbygging stýrikerfisins sjálfs er einnig nokkuð frábrugðin flestum öðrum kerfum. Opin hönnun þess hefur í gegnum tíðina reynst notendum vel. Hún hefur þýtt að oft hefur nýr kerfishugbúnaður verið boðinn fyrr fyrir Unix en önnur stýrikerfi. Þetta á til dæmis við um nethugbúnað.

1.3 Um þessa handbók.

Tilgangurinn með handbókinni er einkum sá að leiðbeina byrjendum og freista þess að útskýra á hvern hátt UNIX-kerfið vinnur. Þannig á handbókin að geta veitt byrjendum stuðning. Í síðari köflum eru einnig veittar fyllri upplýsingar sem geta m.a. aðstoðað kerfisstjóra Unix-véla.

Í 2. kafla, "Helstu Unix-skipanir", er örstutt yfirlit yfir nokkrar skipanir Unix, í 3. kafla, "Pípur o.fl.", er fjallað um inntak og úttak í Unix. Textavinnsla í Unix er nokkuð flókin vegna þess hve hægt er að leysa margbrotin verkefni með henni. Því er fjallað um textavinnslu í nokkrum hlutum. Í kafla 4, "Einföld textavinnsla", er einfalt yfirlit yfir það sem flestir nota, en síðar

er rætt mun ýtarlegar um Emacs-ritþórana í 8. kafla og um umbrotsforritið troff í 12. kafla. Athuga ber að Emacs er langtum öflugri en venjulegur ritþór og sumar útgáfur hans mynda heilsteypt vinnuumhverfi.

Um Unix-kerfið er fjallað nánar í ýmsum köflum : Kafli 5 fjallar um skipanatúlkana csh og sh. Forritunarumhverfinu er stuttlega lýst í 9. kafla og drepið er á strengjameðhöndlun og Unix-ritþórana (vi, ed og sed) í kafla 10 og forritið awk er kynnt í kafla 11.

Í þessu riti er fjallað um fjölda tóla og tækja. Notkun þeirra er m.a. sýnd með því að smíðað verður gagnagrunnskerfið *Forleikur* (sem er reyndar í notkun á Hafrannsóknarstofnun). Kerfinu er lýst í kafla 7 en í kafla 11 verður smíði þess lýst frá grunni.

Sú Unix-bók sem minnist ekki á tölvupóst og ráðstefnur má sín lítils og er fjallað um þetta efni í 6. kafla.

Að lokum fylgja ögn tæknilegri kaflar :

14.	Íslenska og Unix. Um íslenskustaðla, stafrófstöflur ofl.
15.	Allt í steik. Um helstu vandræði sem hrjá byrjendur og viðbrögð við þeim.
16.	Ýmis kerfisforrit.
17.	Kynning á Unix-notendahópnum.

Reynt verður að hafa eitthvað af íslenskum greinum um Unix sem viðauka. Einnig eru fáanlegir viðaukar um tölvupóst ofl.

.nr H1 1 .ds HF 6 6 2 2 2 2 2 .ds HP +2 +1 .nr Hs 3

2 Helstu Unix skipanir.

Hér á eftir fylgir örstuttur listi yfir nokkur algengustu Unix-forrit (skipanir). Nánari upplýsingar um hvert þeirra má fá með því að rita **man skipun**. Einnig má fá lista yfir allar skipanir þar sem *lykilorð* kemur fyrir í lýsingu með því að gefa oskipunina **man -k** *lykilorð*. Til dæmis má gefa skipunina **man -k***file* til að fá upplýsingar um heiti skipanna sem hafa að gera með skrár.

Athugið, að þótt hér séu allsstaðar notuð íslensk skráarnöfn, þá gengur slíkt ekki alltaf í Unix kerfum. Athugið einnig að skýr greinarmunur er gerður á hástöfum og lágstöfum í Unix-skráarnöfnum. Engin skipun hér að neðan skilst ef hún er ekki slegin inn í lágstöfum.

2.1 Skrárskipanir.

Notendur geta geymt upplýsingar (tölur, texta) í **skrám**, sem tölvan sér um að geyma. Þegar notandi fær fyrst aðgang að tölvunni er honum/henni úth-

lutað **skráasafni** (*directory*). Litið er svo á að skráasafnið *innihaldi* skrár notandans og að notandinn sé ávallt staddur á ákveðnu safni. Þannig er úthlutaða skráasafnið kallað **heimasafn** (*home directory* eða *default directory*). Notandi getur smíðað ný (undir-) söfn til að halda verkefnum aðgreindum. Til eru skipanir til að skoða skrár á skjá, prenta skrár, smíða og eyða skrám og söfnum og svo framvegis. Unnt er að skipta um safn og er þá stundum talað um að fara á nýtt *vinnusafn* (hér er einnig oft talað um vinnusvæði). Þ.e. safnið, sem verið er að vinna með (notandi er staddur á) hverju sinni er kallað **vinnusafn** (*current directory* eða *working directory*).

Skráasöfnin verða því eins og tré á hvolfi og er talað um að notandi sé að vinna á tilteknum stað í trénu. Undirsöfnin eru þá **niður** úr núgildandi safni. Flestir Unix-notendur búa til nýtt safn fyrir hvert nýtt verkefni eða málaflokk. Heimasafnið inniheldur oft lítið annað en önnur söfn.

ls	List Files. Gefur yfirlit yfir skrár í vinnusafni.
ls	listi yfir skrár í stafrófsröð
ls *.sh	listi yfir skrár sem enda á .sh
ls -a	lista yfir **allar** skrár (líka sem byrja á ".").
ls -R	lista yfir skrár á vinnusafni og á öllum undirsöfnum.

cat conCATenate. Skoða skrár á skjá. **cat skrá** sendir innihaldið í **skrá** á skjáinn. Sjá nánar dæmi í kafla 3.

pg *Pages*. Skoða skrár síðu fyrir síðu. Eftir fyrstu síðuna kemur **:** neðst á skjánum. Sláið á færsluhnappinn (RETURN/ENTER) til þess að fá upp næstu síðu, eða **q** til að hætta. Sjá **man pg**. Hér má einnig nefna **more** sem er fullkomnara forrit til sama brúks.

pr Sendir skrá á skjá, uppsett með haus o.fl. Ritið 'pr skrá' til að fá skrána á skjáinn, eða

```
pr skrá | lp
```

til að fá hana á prentara. Nota má '-h"texti"' til að setja haus á síður og einnig má stilla línufjölda á síðu, prenta má fleiri en einn dálk á síðu osfrv. Sjá **man pr**.

lp Line printer. Stundum kallað lpr. Dæmi : **lp skrá** sendir skrá beint á prentara. Þessi skipun er mjög háð því, á hvaða vél notandinn er. Ýmsir valmöguleikar eru til, t.d. má oft nota '-orofi' til að gefa skipun (rofa) til prentforritsins. Þess ber að geta að best er að nota **lp** ekki beint heldur ávallt gegnum pípur;

```
pr skrá | lp -ol
```

Gefur 'landscape' prentun (á hlið) á laserjet. Notið 'lpstat', 'lpstat -t' eða 'lpstat -o' til að kanna hvernig staðan er í prentarabiðröðinni. Skipunin 'cancel' er síðan notuð til að stöðva útprentun, t.d.

```
cancel lj-23012
```

Hér er lj-23012 verknúmerið, sem kemur út úr lpstat -o. Sjá **man lp**.

rm *Remove.* Eyðir skrá(m). Notkun : **rm listi**, eyðir öllum skrám, sem eru taldar upp á skipanalínunni.

mv *Move.* Flytur skrána til, þ.e. býr til á hana nýtt nafn. Skipunin **mv a b** breytir nafninu á skrá sem hét **a** í **b**. Einnig er hægt að flytja margar skrár inn á eitt **safn**, t.d. **mv a b c safn**.

cp *Copy.* Tekur afrit, *kópíu* af skrá. Rita má **cp a b** til að smíða skrá, b, sem hefur eins innihald og a, en líka **cp a b c d safn** til að taka afrit af mörgum skrám inn á eitt safn. Nöfn nýju skránna verða þá eins og þeirra gömlu.

cd, pwd *Change directory* og *print working directory.* Skipunin **cd safn** flytur notandann yfir á annað skráasafn, þ.e. skiptir um vinnusafn. Sjá nánar í skel-kaflanum, kafla 5, og svo auðvitað **man cd**.

mkdir, rmdir Skipunin **mkdir** (**make directory**) býr til nýtt safn og skipunin **rmdir** eyðir því. Athugið að t.d. **mkdir safn** breytir ekki vinnusafni notanda, heldur býr til nýtt skráasafn (**safn**) sem er undirsafn í vinnusafni.

2.2 Samskipti notenda.

write Kemur á beinu sambandi við annan notanda. Til að koma skilaboðum til notandans atli (sem verður að vera tengdur), má rita **"write atli"** og síðan þau skilaboð, sem Atli á að fá. Hjá Atla birtis þá á skjánum **Message from gunnar** og síðan fylgja skilaboðin. Ef Atli hefur áhuga á að svara getur hann strax ritað **write gunnar**. Þá fer allt sem annar ritar einnig á skjá hins, uns ritað er **^D** (haldið niðri "control" lykli, meðan ýtt er á D) fremst í línu. Skilaboðin birtast línu fyrir línu. Þannig er hægt að leiðrétta línuna áður en hún er send. Óþægilegt er að fá á sig línu þegar svar er samið, þess vegna er ágætt að nota eftirfarandi aðferð: Þegar hinn aðilinn má byrja er sett lítið **o** aftast í línu (o stendur fyrir "over"). Svo er beðið átekta þar til lína frá honum birtist, en ekki byrjað að svara fyrr en o-línan birtist. Síðustu skilaboðin má svo enda á **oo** (over and out). Sjá nánar **man write**.

mail/elm Tölvupóstur (sjá kafla 6). Ritið 'elm' til að lesa/senda tölvupóst. Ef elm er ekki til, ætti að reyna að setja það upp, ellegar prófa mailx. Mail er yfirleitt ekki notað beint af mannfólki, heldur aðeins af öðrum forritum. Sjá **man elm**.

news Ráðstefnukerfi. Notað til að margir notendur geti haft samskipti sín á milli. Hver og einn notandi getur

1. lesið þær ráðstefnur sem hann hefur áhuga á,
2. sent inn fyrirspurnir, skeyti eða greinar
3. svarað öðrum innan ráðstefnukerfisins ellegar með tölvupósti.

Ritið 'rn' til að lesa fréttir. Ýtið á 'bil' í rn, til að halda áfram, 'q' nokkrum sinnum til að hætta. Sjá nánar kafla 6 eða **man rn**.

2.3 "Forrit sem meðhöndla innihald skráa."

Unix er ansi ólíkt öðrum stýrikerfum hvað varðar þann fjölda forrita sem fylgja kerfinu. Þetta er sérlega áberandi þegar litið er yfir þau forrit, sem fylgja

með Unix kerfum og eru ætluð til almennrar textavinnslu og gagnavinnslu.

Þótt strangt til tekið sé meira en nóg af gagnagrunns- og textavinnslu- forritum með Unix kerfinu, finnst flestum nauðsynlegt að fá gagnagrunnskerfi, sem veita fleiri möguleika. Í þessarri handbók verður sýnt, hvernig byggja má upp og nota slíkt gagnagrunnskerfi. Kerfið nefnist **Forleikur** og í eftirfarandi lista yfir Unix forrit verður einnig vísað í tilsvarandi Forleiks-forrit, sem gera sambærilega hluti, en vinna á Forleiksgagnagrunninum. Forleikur er hugsað sem undirstöðusafn skipana. Almennur notandi getur nýtt sér þetta skipanasafn til að byggja upp skipanir til að vinna verk sín.

sort Raðar skrá. Nota '-n' ef raða skal tölulega (annars raðast 10 og 2 sem stafastrengir og þá 10 á undan 2).

```
sort < skrá1 > skrá2
```

```
sort -n < skrá1 > skrá2
```

Athuga ber, að sort getur einnig raðað skrám með föstum dálkum. Slíkar skrár voru lesnar og skrifaðar áður fyrr í forritunarmáli sem nefndist Fortran en heyrir nú að mestu sögunni til. Ritið **man sort** fyrir frekari upplýsingar og sjáið *sorttable* í kafla 7 fyrir tilsvarandi skipun í Forleik.

uniq Tekur út (og/eða telur) endurteknar línur. Ef notað er '-c', er bætt inn dálki, sem telur, hve margar línur voru eins. Yfirleitt notað með sort, t.d.

```
sort -n < skrá | uniq
```

til að fá eitt eintak af hverri línu, eða

```
sort -n < skrá | uniq -c
```

til að fá eitt eintak af hverri línu, með fjöldadálk fyrir framan. Nánari upplýsingar fást eins og alltaf með **man**-skipuninni, **man uniq**. Tilsvarandi Forleiks-skipun er *count* (kafla 7).

wc Telur línur, orð og stafi í skrá. Sjá dæmi í kafla 3.

cut Velur dálka úr skrá. Notkun :

```
cut -clisti
```

eða

```
cut -flisti
```

þar sem 'listi' er upptalning á, hvaða dálka eigi að plokka úr inntaki. Valmöguleikinn '-c' er notaður fyrir fasta dálka (sbr Fortran), en '-f' er notað ef dálkar eru aðskildir með tab-tákni. Nota má '-dstafur', t.d. '-d":"' ef ":" skilur milli dálka.
Dæmi :

```
cut -c1-3,8,12-14 < skrá
```

Klippir út dálka 1-3, þá þann áttunda og loks dálka 12-14. Tilsvarandi Forleiks-skipun er 'project' (kafla7).

awk Öflugt forrit fyrir texta- og gagnavinnslu. (sjá kafla 11). Awk getur flest það sem fokdýr gagnagrunnskerfi geta.

grep, egrep Leitar að strengjum í skrá.

```
grep 'strengur' < skrá
grep 'strengur'  skrár
egrep 'flókinn strengur' < skrá
```

Hér getur "flókinn strengur" verið geysiflókinn, m.a. má nota '|' sem "eða" milli mismunandi strengja, '^' táknar upphaf línu, '$' táknar enda línu, '.' táknar "einhvern staf", '*' á eftir staf táknar "0 eða fleiri". Sjá nánar í kafla 3 og um strengi í kafla 10.

join, paste Join setur skrár saman, línu fyrir línu, skv. lykli. Notkun :

```
join skrá1 skrá2  [ > skrá3]
```

Tilsvarandi Forleiks-skipun er 'jointable', sem lýst er allítarlega í kafla 7.
Skipunin **paste** er svipuð join, en límir tvær skrár saman hlið við hlið, línu fyrir línu, án notkunar lykils. Sjá kafla 3.

touch Touch er svolítið skrýtið forrit, sem gerir ekkert við skrárnar, en breytir tímanum á skránni. Þannig er gömul skrá látin líta út eins og henni hafi verið nýlega breytt með **touch skrá**. Touch er mikið notað með make ofl.

2.4 Ritþórar.

vi, ed Ritþórar (sjá kafla 10). Yfirleitt kjósa notendur fremur einhvern **emacs** ritþór (sjá kafla 4 og 8), en nauðsynlegt er að kunna einnig undirstöðuatriði **vi** og **ed**. Notkun er

```
vi skrá
```

og síðan er notað **ZZ** til að geyma skrána og hætta vinnslu. Stundum þarf að nota "escape" fyrst. Sjá nánar kafla 10.

2.5 Annað, ýmislegt.

tar Tape AR-chiver. Skrifar skráasafn á band eða diskettu. Helstu valmöguleikar eru

c create	Núllstilla og skrifa út á bandið.
r	Bæta við
x extract	Lesa inn af bandinu.
t	Rita aðeins upplýsingar á skjáinn (ekki skrifa neitt).
v verbose	Rita fullar upplýsingar á skjáinn
f file	Tiltaka aðra bandstöð (eða skrá).

Dæmi :

tar cv skrár	Núllstillir bandið og skrifar 'skrár' þangað. Skrárnar mega líka vera vinnusvæði og eru þá allar skrár og undirvinnusvæði sett út.
tar cvf tæki skrár	Núllstilla "tæki" og senda skrárnar þangað. Athugið, að "tæki" má vera á forminu "/dev/mt0", eða bara venjuleg skrá.
tar xvf tæki	Lesa allar skrár, sem hafa verið skrifaðar á tækið.

compress Þjappar skrám. Kemur af tölvunetinu. Notkun : 'compress skrár'. Setur viðbótina '.Z' aftan við nafnið. Notið svo 'uncompress' eða 'compress -d' til að breyta tilbaka. Nota má 'zcat' til að skoða skrá á skjánum.
Compress er oft notað með tar til að flytja fullt af skrám um tölvunetið. Þá er fyrst gefin tar skipunin **tar cvf skra.tar skrár** til að setja "skrár" saman með tar í eina stóra skrá, sem hér heitir "skra.tar". Næst er compress notað til minnka skrána um allt að helming, **compress skra.tar** og verður þá til "skra.tar.Z". Sú skrá er flutt milli véla og síðan er hún tekin í sundur aftur :

```
uncompress skra.tar.Z
tar xvf skra.tar
```

asa Breytir Fortran stýritáknum í stýritákn fyrir prentara. Dæmigerð notkun er með lp :

```
asa < skrá | lp
```

at At les skipun af lyklaborði og framkvæmir síðar. Þannig getur notandi t.d. óskað þess að skipanirnar "make" og "run" séu settar af stað klukkan 10 að kvöldi. "run" getur verið forrit, skipanaskrá, eða hvað sem er. Til að gera þetta er eftirfarandi ritað af lyklaborði:

```
at 2200
make
run
^D
```

diff Ber saman innihald tveggja skráa og skilar upplýsingum um hvar og hverning skránum ber ekki saman. Oft notað í tengslum við **patch** til að uppfæra eina skrá til samræmis við aðra.

```
diff skrá1 skrá2 > breytingar
```

tail Skilar ákveðnum fjölda lína aftast úr skrá. Má einnig nota til að fylgjast með innihaldi skráa sem forrit skrifa meðan þau eru í keyrslu. Sjá **man tail**.

split Brýtur skrá upp í minni búta. Sjá einnig **csplit** sem brýtur skrá upp eftir ákveðnum skilyrðum.

3 Inntakssvig, Úttakssvig og Pípur.

Mörg Unix forrit (t.d. cat) gera ráð fyrir að inntaks skrár séu taldar upp á skipanalínunni, t.d.

```
cat skrá1 skrá2
```

og forritið sendir útkomuna beint á skjáinn. (cat setur þannig skrárnar sama í eina úttaksskrá, þ.e. á skjáinn) Ef setja skal úttak úr slíkri skipun í skrá, má gera það með

```
cat skrá1 skrá2 > skrá3
```

Síðan má nota önnur forrit til að vinna úr úttakinu, t.d.

```
wc  skrá3
```

sem gefur línu, orða- og stafafjöldann í skránni. Í stað þess að vinna úr úttakinu á þennan hátt má nota *pípur*. Skipanirnar tvær

```
cat skrá1 skrá2 > skrá3
wc  skrá3
```

verða þá að einni

```
cat skrá1 skrá2 | wc
```

og ekki þarf að búa til skrá fyrir milliniðurstöður.

Ef skráarnafn er ekki gefið upp á skipanalínunni gera forritin yfirleitt ráð fyrir að inntakið komi beint af lyklaborðinu (og endi á ^D, sem er "end-of-file" merkið þaðan). Þannig má rita

```
cat  >  dæmi
texti á
þremur
línum
^D
```

til að smíða skrá að nafni "dæmi", sem inniheldur 3 línur af texta. Einnig er unnt að rita

```
wc
texti á
þremur
línum
^D
```

og fá á skjáinn niðurstöðuna

```
3  4  21
```

Sama niðurstaða hefði fengist með því að gefa skipunina

```
wc < dæmi
```

Þar sem "dæmi" er skráin, sem var smíðuð með síðustu cat-skipun.
Til að telja fjölda stafa, orða og lína í "skrá1" og "skrá 2" má því rita

- **wc** *skrá1 skrá2*

  sem gefur niðurstöður af forminu:

10	50	500	skrá1
20	100	700	skrá2
30	150	1200	Total

  ellegar

- **cat** *skrá1 skrá2* > *skrá3*
  wc < *skrá3*

  sem gefur

30	150	1200

  Síðasta dæmið má eins og áður var sýnt einfalda, með því að tengja úttak úr einu forriti beint við inntakið í það næsta með pípum.

- **cat** *skrá1 skrá2* | **wc**

  sem gefur nákvæmlega sömu viðurstöðu og (2) án þess að "skrá3" sé búin til. Fjölmörg dæmi um tengingu forrita verða gefin í kafla 7.

Til að kynna aðeins einfalda gagnavinnslu í Unix má taka sem dæmi eftirfarandi tvær skrár, gögn.ex1 :

1234567890123411111
9876543210987654321
3234567890323433333
4234567890423444444

og gögn.ex2 :

9876567890987699999
9876567890987656789
7876567890787677777
6876567890687666666

Skipunin paste getur sett skrár saman "á hlið", línu fyrir línu, þannig gefur

```
paste gögn.ex*
```

eftirfarandi niðurstöðu á skjáinn :

1234567890123411111 9876567890987699999
9876543210987654321 9876567890987656789
3234567890323433333 7876567890787677777
4234567890423444444 6876567890687666666

Eitt tabb-tákn aðskilur dálkana sem paste skilar af sér. Venjulega er þó join notað í stað paste, því join kann að nota lykla til að tengja saman skárnar.

Setja má skrárnar saman hverja á eftir annarri með cat, t.d.

```
cat gögn.ex*
```

og á skjáinn kemur niðurstaðan :

1234567890123411111
9876543210987654321
3234567890323433333
4234567890423444444
9876567890987699999
9876567890987656789
7876567890787677777
6876567890687666666

Til að pilla út dálka má nota cut. T.d. fást dálkar 5-9 og dálkar 10-12 með skipuninni

```
cut -c5-9,10-12 < gögn.ex1
```

56789012
54321098
56789032
56789042

Síðasta gagnagrunnsaðgerðin er að taka út valdar línur. Til þess má m.a. nota grep-fjölskylduna. Til dæmis má ná í þær línur, sem innihalda '1' með :

```
grep '1' < gögn.ex1
```

Það eru aðeins tvær línur, sem innihalda 1 :
1234567890123411111
9876543210987654321
Leita má að miklu flóknari hlutum, t.d. tölum á bilinu 1-4 ellegar 6:

```
egrep '[1-4]|6' < fortran.ex1
```

1234567890123411111
9876543210987654321
3234567890323433333
4234567890423444444
Að lokum ber að geta þess, að mjög mörg Unix forrit geta ýmist lesið af lyklaborði ellegar tekið skráarnöfn af skipanalínunni. Þannig má rita

```
grep 'strengur' skrá
```

og

```
grep 'strengur' < skrá
```

og fá sömu niðurstöðu. Í fyrra tilvikinu sér grep um að opna og lesa skrána, en í því síðara sér stýrikerfið um að tengja grep-skipunina við skrána, og grep heldur að það sé að lesa af lyklaborðinu.
Til að leita að streng í mörgum skrám má nota t.d.

```
egrep '[1-4]|6' gögn.ex?
```

og fá þá út línurnar auk upplýsinga um, hvaðan þær komu :

gögn.ex1:1234567890123411111
gögn.ex1:9876543210987654321
gögn.ex1:3234567890323433333
gögn.ex1:4234567890423444444
gögn.ex2:9876567890987699999
gögn.ex2:9876567890987656789
gögn.ex2:7876567890787677777
gögn.ex2:6876567890687666666

4 Gagnagrunnskerfi

4.1 Inngangur

Hér á eftir fylgir lýsing á einföldu gagnagrunnskerfi, sem hefur verið skrifað á Hafrannsóknastofnun. Kerfið er einfalt í uppsetningu og fylgir Unix-ímyndinni um einföld tól og pípur milli þeirra. Sjá nánari lýsingu á notkun í viðauka IV, og lýsing á, hvernig má á einfaldan hátt smíða slíkt gagnagrunnkerfi í kafla 11.

Gagnagrunnskerfið nefnist Forleikur, því það er hugsað sem grunnur, sem unnt er auðveldlega að byggja úr. Kerfið er byggt upp með Prelude frá VenturCom, Inc að fyrirmynd, en inniheldur færri möguleika. Forleikur er ókeypis og gengur á allar Unix vélar.

Gögn eru geymd á dálkaformi, þar sem tab-táknið aðskilur dálkana. Fyrstu tvær línur í hverri skrá eru haus, og inniheldur fyrsta línan nöfn dálkanna en sú næsta inniheldur mínusa.

Dæmi:

Hlutur	Fjoldi	Aths
1	10	Nóg til
2	5	Þarf að panta
5	800	Þarf að selja
3	0	Vantar
10	15	Gott ástand

Athuga ber, að nöfn dálka hér, (eins og yfirleitt gildir um flest í Unix) mega innihalda venjulega enska stafi og undirstrikunarmerkið (*underscore*, _). Ekki er ráðlegt að reyna séríslenska stafi, né heldur punkta og mínusa. Sum slíkra tákna munu ganga í stöku Forleiks-forritum, en ólíklegt er að þau gangi t.d. í **select** og **compute**.

4.2 Helstu skipanir

addcol	Bætir inn nýjum, tómum dálki.
check	Kannar hvort skráin sé lögleg (fj. dálka réttur í öllum línum m.v. fyrstu línu).
count	Bætir inn fjöldadálki og telur inn í hann hve oft eins línur koma fyrir.
compute	Reiknar gildi í dálka.
dbdict	"database dictionary"
jointable	Setur raðaðar skrár saman skv lykli í fyrsta dálki.
math	Reiknar summur, meðalt. og staðalfrávik af hverjum dálki.
matrix	Snýr skrá af (x, i, j)-formi yfir á fylkjaform.
number	Bætir inn dálki með línunúmeri.
plokk	Pillar valdar línur úr gagnaskrá, skv stýriskrá.
project	Velur dálka skv nöfnum.
select	Velur línur skv skilyrði.
sideview	Skoðar skrána á hlið á skjánum.
see	Sýnir skrána, með stýritáknum.
sorttable	Raðar skrá (á lykil). Þetta er forleiks-útgáfan af sort.
subtotal	Leggur saman gildi tiltekinna dálka innan flokkunar skv. öðrum dálkum.
union	Setur saman skrár (eins og cat, nema tekur aðeins haus úr fyrstu skrá).
preprint	Undirbýr töflu fyrir tbl (og nroff eða troff).
pretobmdp	Snýr gögnum á BMDP form.
pretoglim	Snýr gögnum á Glim form.

4.3 Einföld dæmi um notkun

Gerum í eftirfarandi ráð fyrir að skráin temp innihaldi

Hlutur	Fjoldi	Aths
1	10	Nóg til
2	5	Þarf að panta
5	800	Þarf að selja
3	0	Vantar
10	15	Gott ástand

4.3.1 Project og addcol.

Project dregur ákveðna dálka úr skránni og er notkunin 'project dálkar < skrá', t.d.

```
project Hlutur Aths < temp
```

sem gefur eftirfarandi á skjáinn :

Hlutur	Aths
1	Nóg til
2	Þarf að panta
5	Þarf að selja
3	Vantar
10	Gott ástand

Ef gefið er nafn á dálki, sem ekki er til, þá smíðaður nýr dálkur með því nafni.

Addcol bætir inn nýjum dálki í skrá. Þannig er **addcol** einungis til tímasparnaðar, því að sjálfsögðu væri einnig unnt að nota **project** með því að telja fyrst upp alla gömlu dálkana og bæta síðan þeim nýja við.

4.3.2 Select

Select velur línur úr skrá. Þannig má velja línur með a.m.k. 10 í fjöldadálki með skipuninni

```
select 'Fjoldi >= 10' < temp
```

Og birtist þá eftirfarandi á skjánum :

Hlutur	Fjoldi	Aths
1	10	Nóg til
5	800	Þarf að selja
10	15	Gott ástand

Til að fá númer þeirra hluta, sem er lítið til af má nota select og project saman :

```
select 'Fjoldi >= 10' < temp | project Hlutur
```

Svarið kemur nær samstundis :

Hlutur
1
5
10

Þarna er augljós kostur þess, hvernig Forleiks-forritin eru uppbyggð, því að auðvelt er að tengja saman eins mörg forrit og hvern lystir.

Unnt er að velja samkvæmt mjög flóknum skilyrðum með **select** skipunum. Til reiðu eru eftirfarandi aðgerðir:

Aðgerð	Þýðir	Dæmi
<	minna en	dálkur < 20
>	stærra en	dálkur1 > dálkur2
<=	minna eða jafnt	dálkur1 <= log(dálkur2)
>=	stærra eða jafnt	dálkur >= 20
&&	og	dálkur1 <=3 && dálkur2 >=14
||	eða	dálkur1 <=3 || dálkur1 >=1

Athugið merkingu **og** hér. Samsett skilyrði á forminu *Skilyrði1* **og** *skilyrði2* er yppfyllt þá og því aðeins að bæði skilyrðin séu uppfyllt. Þannig velur skipunin **select 'dálkur1 <=3 && dálkur1 >=1'** þær línur, þar sem dálkurinn *dálkur1* er á bilinu 1-3 (a.m.k. 1 og ekki meira en 3). Tilsvarandi, þá þýðir **eða** að a.m.k. annað skilyrðið sé uppfyllt. Raða má saman skilyrðum með **og** og **eða** á milli, en muna þó að nota sem mest af svigum, til að augljóst sé, í hvaða röð á að gera samanburðinn. Enginn ætti t.d. að rita

```
select 'dálkur1 <=3 || dálkur1 >=1 && dálkur2 > dálkur3'
```

nema viðkomandi sé viss um, hvort þessi skipun þýði

```
select 'dálkur1 <=3 || (dálkur1 >=1 && dálkur2 > dálkur3)'
```

eða

```
select '(dálkur1 <=3 || dálkur1 >=1) && dálkur2 >
dálkur3'
```

Að sjálfsögðu er frumskilyrði að notandi gerir sér grein fyrir muninum á þessum tveimur skipunum.

4.3.3 Sorttable.

Röðun vefst fyrir mörgum, því unnt er að raða á marga vegu. Yfirleitt er texta raðað með einhverri stafrófsröðun en tölum er raðað með venjulegri tölulegri röðum. Sorttable raðar skv stafrófsröðun, nema ef '-n' er notað.

Við stafrófsröðun raðast venjulegir stafir í stafrófröð, og tölur raðast eins og þær væru bara stafastrengir.

Dæmi :

```
sorttable < temp
```

gefur

Hlutur	Fjoldi	Aths
1	10	Nóg til
10	15	Gott ástand
2	5	Þarf að panta
3	0	Vantar
5	800	Þarf að selja

Hér er öllum línum raðað í stafrófsröð. Athugið, að fyrst koma strengir, sem byrja á '1', þá þeir,sem byrja á 2 osfrv. Ef notuð er töluleg röðun verður þetta þannig :

```
sorttable -n < temp
```

Hlutur	Fjoldi	Aths
1	10	Nóg til
2	5	Þarf að panta
3	0	Vantar
5	800	Þarf að selja
10	15	Gott ástand

4.3.4 Jointable

Gerum nú ráð fyrir að við höfum gefið skipunina

```
sorttable < temp > temp.srt
```

Þannig að skráin temp.srt inniheldur :

Hlutur	Fjoldi	Aths
1	10	Nóg til
10	15	Gott ástand
2	5	Þarf að panta
3	0	Vantar
5	800	Þarf að selja

Ef önnur skrá, t.d. test, inniheldur

Hlutur	Thyngd
1	43.2
10	21
2	9
3	10
5	34

þá má gefa skipunina

```
jointable test temp.srt > junk
```

og fá út skrána junk, sem inniheldur :

Hlutur	Thyngd	Fjoldi	Aths
1	43.2	10	Nóg til
10	21	15	Gott ástand
2	9	5	Þarf að panta
3	10	0	Vantar
5	34	800	Þarf að selja

Athugið sérstaklega, að forritið **jointable** gerir ráð fyrir stafrófsröðun!!! **Ekki má nota 'sorttable -n' með jointable.** Valmöguleikinn '-n' í sorttable er eingöngu nauðsynlegur fyrir lokaútgáfur af töflum, og langeðlilegast er að vinna með gögn í stafrófsröð. Ef gögn eru röðuð tölulega verður að endurraða þeim án '-n' áður en jointable er notað.

4.3.5 Compute.

Forritið **compute** reiknar gildi í dálka. Nota má allskyns reiknisetningar þ.m.t. lógariþma ofl.

Nú má reikna út magn af hverjum hlut í nýjan dálk :

```
addcol magn < junk | compute 'magn=Fjoldi*Thyngd'
```

og út kemur ný tafla :

Hlutur	Thyngd	Fjoldi	Aths	magn
1	43.2	10	Nóg til	432
10	21	15	Gott ástand	315
2	9	5	Þarfað panta	45
3	10	0	Vantar	0
5	34	800	Þarf að selja	27200

4.3.6 Subtotal

Forritið **subtotal** reiknar hlutsummur ákveðinna dálka, fyrir gefin gildi annarra dálka. Notkun :

```
subtotal by fastirdálkar on summudálkar
```

Athugið, að skráin þarf að vera röðuð á föstu dálkana. Inntaksskráin er einfaldlega lesin línu fyrir línu, athugað, hvort föstu dálkarnir séu eins og í næstu línu á undan og summudálkarnir lagðir saman meðan svo er. Þegar kemur að línu, sem ekki er eins og næsta lína á undan, eru föstu dálkarnir skrifaðir út ásamt summunum. Þvínæst byrjað umm á nýtt.

Ef inntaksskrá inniheldur t.d.

x	y	z	w
1	2	3	4
1	2	5	4
1	2	4	8
5	8	9	3
5	9	8	7
5	9	8	7
6	9	8	7

þá gefur skipunin **subtotal by x y on z w** eftirfarandi niðurstöður:

x	y	z	w
-	-	-	-
1	2	12	16
5	8	9	3
5	9	16	14
6	9	8	7

Dálkarnir mega vera í hvaða röð sem er í inntakinu og einnig mega vera aukadálkar í inntaki, en þeim er sleppt í úttaki.

5 Emacs

5.1 Inngangur - sögulegur bakgrunnur.

Emacs var upphaflega saminn við Gerfigreindardeildina (Artificial Intelligence) í MIT. Sá Emacs var skrifaður í LISP fyrir TWENEX (eða TOPS-20) stýrikerfið á Digital system 20-60 vélar. Aðalhöfundur var Richard Stallman. Allir seinni tíma emacs-ar eru stælingar og/eða útvíkkanir á þessum Emacs. Næst ber að nefna Emacs fyrir Unix, skrifaður í C af Gossling. Sá er nú seldur sem Unipress Emacs og er góð eftirlíking af þeim upphaflega. Koma síðan á sjónarsviðið fjöldi eftirlíkinga, gerðir fyrir Unix, en eru flestir aðeins smækkaðar útgáfur af þeim upprunalega, nema CCA Emacs. Smærri útgáfurnar skortir forritunarmöguleika þann, sem felst í því, að í alvöru Emacs er hægt að skrifa forrit í LisP eða MLisp. Þau forrit eru síðan notuð sem viðbætur við innbyggðar skipanir í Emacs-inum.

Næsta stökk í Emacs málum kemur með tilkomu GNU Emacs. Þar er Richard Stallman kominn aftur, að þessu sinni ekki hjá MIT, heldur hjá GNU Project, sem skrifar og dreifir ókeypis hugbúnaði.

Ef við skilgreinum alvöru Emacs sem þann sem keyrir á Unix vél og getur amk allt sem sá upphaflegi getur, þá eru til

1. GNU Emacs
2. Unipress Emacs
3. CCA Emacs

Af þessum er aðeins GNU ókeypis, en gengur ekki á hvaða vél sem er og er ekki studdur neitt af þeim sem þróa og dreifa honum. Hins vegar virðist GNU vera langbestur allra emacsa á markaðnum í dag. Ef hann gengur á viðkomandi tölvu, er eindregið mælt með honum. Hinir tveir kosta pening en

fyrir vikið ábyrgjast fyrirtækin, að þeir gangi á umræddum vélum og reynt er að gera við galla.

Fáir notendur nýta til fullnustu alla þá möguleika, sem t.d. Lisp forritun í GNU Emacs býður upp á . Því er hugsanlegt, að á tiltekinni vél sé meira en nóg að setja upp smærri útgáfu (sem þá gengur líka á t.d. 16 bita vél).

Af smærri útgáfum (ókeypis) má nefna:

uemacs	**Micro Emacs.** Ræður við ECMA á **alla 8-bita** skjái (líka Dec og Roman 8) Gengur á vt100. Góð handbók.
mg	**Micro GNU Emacs. Íslenska á alla 8 bita skjái.** Mjög frekur á vél.
jove	**Góð handbók.** Nokkuð öflugur. Engin 8 bita íslenska. Góður ritþór, en er í vandræðum með vt100 og 8 bita íslensku.

Í íslenskumálum mælum við eindregið með, að notað sé ECMA. Ef skjáir eru blandaðir, þarf mg, uemacs eða GNU, en athuga ber, að mg getur verið hættulegur á fjölnotenda vélum (grípur vélina eins og hann getur. Þetta er þó auðvelt að laga.)

Með tilkomu hinna ýmsu minni útgáfa af ritþórnum hefur því miður tapast ýmislegt, sem var áður talinn óaðskiljanlegur hluti af Emacs. Í góðum Emacs eru hlutir, sem eru yfirleitt ekki í öðrum ritþórum :

Command completion.	Í miðri skipun má rita ? til að fá lista yfir möguleika á, hvernig hægt er að klára skipunina. Nota má <esc> til að klára hálf-ritaða skipun. Rita má <bil> til að klára næsta hluta hálfritaðrar skipunar. Sjá 8.10.
"Info"	Stóru Emacsarnir hafa innbyggt öflugt valmyndadrifið hjálparforrit, **info**, sem veitir fullar upplýsingar um allar emacsskipanir. Hér er að finna allar upplýsingar, sem eru í handbókunum.
Abbrev mode	Emacs hefur tök á að muna skammstafanir, þannig að ef t.d. er ritað 'hafro', þá sé því samstundis breytt í Hafrannsóknastofnunin.
Lisp	Stóru emacsarnir bjóða upp á innbyggt fullkomið forritunarmál, Lisp eða MLisp. Notandi getur skrifað Lisp forrit og látið Emacs framkvæma það. Slíkt býður upp á geysimikla möguleika og eru t.d. fáanleg Lisp forritunarsöfn til að lesa tölvupóst og fréttir, Emacs getur hagað sér eins og vi eða EDT osfrv.

5.2 Skipanayfirlit

Í eftirfarandi eru þær skipanir, sem koma úr skipuninni

esc-Xdescribe-bindings

Hér eru meðtaldar nokkrar bindingar, sem er bætt inn með þeirri .uemac-src skrá, sem er lýst síðar.

abort-command	^G
add-mode	^XM
add-global-mode	M-M
apropos	
backward-character	^B
	M-D
begin-macro	^X(
beginning-of-file	M-<
beginning-of-line	^A
bind-to-key	M-K
buffer-position	^X=
case-region-lower	^X^L
case-region-upper	^X^U
case-word-capitalize	
case-word-lower	M-L
case-word-upper	M-U
change-file-name	^XN
change-screen-size	M-^S
change-screen-width	M-^T
clear-and-redraw	^L
clear-message-line	
copy-region	M-W
count-words	M-^C
ctlx-prefix	^X
delete-blank-lines	^X^O
delete-buffer	^XK
delete-mode	
delete-global-mode	M-^M
delete-next-character	^D
delete-next-word	
delete-other-windows	^X1
delete-previous-character^H	^?
delete-previous-word	M-^H
	M-^?

delete-window	^X0
describe-bindings	
describe-key	^X?
detab-line	^X^D
end-macro	^X)
end-of-file	M->
end-of-line	^E
entab-line	^X^E
exchange-point-and-mark	^X^X
execute-buffer	
execute-command-line	
execute-file	
execute-macro	^XE
execute-macro-1	
execute-macro-2	
...	
execute-macro-40	
execute-named-command	M-X
execute-procedure	M-^E
exit-emacs	^C
	^X^C
fill-paragraph	M-J
filter-buffer	^X#
find-file	^X^F
forward-character	^F
	M-C
goto-line	M-G
goto-matching-fence	M-^F
grow-window	^X^
	^XZ
handle-tab	^I
hunt-forward	
hunt-backward	
help	M-?
i-shell	^XC
incremental-search	^XS
insert-file	^X^I
insert-space	
insert-string	
kill-paragraph	M-^W

kill-region	^W
kill-to-end-of-line	^K
list-buffers	^X^B
meta-prefix	^]
move-window-down	^X^N
move-window-up	^X^P
name-buffer	M-^N
newline	^M
newline-and-indent	^J
next-buffer	^XX
next-line	^N
	M-B
next-page	^V
next-paragraph	M-N
next-window	^XO
next-word	M-F
open-line	^O
pipe-command	^X@
previous-line	^P
	M-A
previous-page	^Z
	M-V
previous-paragraph	M-P
previous-window	^XP
previous-word	
query-replace-string	M-^R
	M-Q
quick-exit	M-Z
quote-character	^Q
read-file	^X^R
redraw-display	M-^L
	M-!
resize-window	^XW
restore-window	
replace-string	M-R
reverse-incremental-search^XR run	M-^E
save-file	^X^M
	^X^S
save-window	
scroll-next-up	M-^Z

scroll-next-down	M-^V
search-forward	^S
search-reverse	^R
select-buffer	^XB
set	^XA
set-encryption-key	M-E
set-fill-column	^XF
set-mark	M-
	M-.
	^@
shell-command	^X!
shrink-window	^X^Z
split-current-window	^X2
store-macro	
store-procedure	
transpose-characters	^T
trim-line	^X^T
unbind-key	M-^K
universal-argument	^U
unmark-buffer	M-
update-screen	
view-file	^X^V
wrap-word	M-FNW
write-file	^X^W
write-message	
yank	^Y

5.3 Extended' skipanir

Hægt er að gefa margar skipanir, sem ekki eru bundnar við lykla. Á ýmsar hefur verið minnst áður, t.d.

esc-X describe-bindings

sem gefur (í uemacs) lista yfir allar skipanir og lykla, sem þær eru tengdar við.

Allar skipanir er unnt að gefa með nafni, ef fyrst er notað esc-X (sem er kölluð 'execute-named-command' eða 'execute-extended-command') og svo ritað fullt nafn á skipuninni. Þetta er eina leiðin til að framkvæma þær skipanir, sem ekki eru tengdar við lykla.

Ekki væri tilveran mjög spennandi ef allir emacsar væru eins og því er svolítil fjölbreytni varðandi format á extended skipunum. Nöfn á extended

skipunum eru ekki eins í öllum emöcsum. Þannig má gefa af lyklaborði skipunina

```
<escape>xbind-to-key forward-character  <escape>C
```

í uemacs til að láta <escape>-C gefa skipunina "forward-character". Tilsvarandi skipun í Unipress Emacs er eins, en í MicroGnuEmacs er notað

```
<escape>xglobal-set-key<return><escape>Cforward-char
```

Skipunin til að binda lykla heitir þannig "global-set-key" í mg, en "bind-to-key" í uemacs.

Yfirleitt er lengra from skipana í mg sem næst eins og í GNU Emacs, en Unipress Emacs og uemacs eru líkari hinum upprunalega.

5.4 Endurskilgreining lykla (uemacs ofl)

Til að breyta emöcsum þannig að hverjum líki má m.a. endurbinda lyklaborðið. Þetta er gert með bind-to-key skipun í uemacs. Til dæmis senda örvatakkar á vt52-skjá fyrst escape og svo einn staf. Til að láta þessa takka virka í uemacs má gefa eftirfarandi skipanir :

```
bind-to-key forward-character     M-C
bind-to-key backward-character    M-D
bind-to-key previous-line         M-A
bind-to-key next-line             M-B
```

Slíkar skipanir eru oft settar í skrá eins og .uemacsrc, sem uemacs les þegar hann er settur af stað.

Athugið, að einn ókostur við slíkar bindingar er að eitthvað annað tapast í staðinn (t.d. er M-C = escape C venjulega bundið við "upper case word").

Ef mismunandi emacsar eru notaðir, t.d. v.þ.a. einn er betri til forritunarvinnu, en annar til textavinnslu, er mjög gott að geta breytt lyklaborðinu þannig að samræmi sé milli heltu skipana.

5.5 .uemacsrc-skrár. (uemacs)

Þegar emacs er settur af stað, er byrjað á að lesa ýmsar skrár, sem segja til um, hvernig almenn hegðun hans á að vera. Til dæmis les uemacs skrána .uemacsrc á heimasvæði ($HOME) notandans, ef sú skrá er til. Í slíka skrá

getur notandi raðað skipunum, t.d. lyklabindingum, til að láta uemacs haga sér eftir þörfum.

Eftirfarandi .uemacsrc skrá er hentug fyrir HP-skjá.

```
;  Emacs.Rc:  Startup File for MicroEMACS 3.8
;            This file is executed everytime the
;            editor is entered
write-message "[Setting up....]"
;       ***** Rebind the arrow key group
bind-to-key forward-character           M-C
bind-to-key backward-character  M-D
bind-to-key previous-line               M-A
bind-to-key next-line           M-B
;       *****
bind-to-key save-file                   ^X^M
bind-to-key exit-emacs                   ^C
bind-to-key set-mark                     ^@
add-global-mode WRAP
;       ***** Make the keyboard active
write-message "^[&s1A"
bind-to-key fill-paragraph M-J
bind-to-key query-replace-string M-Q
```

Síðan er skilgreiningin

```
   alias em  'uemacs
!* ; echo "^[&s0A"'
```

sett í .cshrc. Þetta, ásamt write-message er gert til að (1) uemacs geri lyklaborðið á HP-skjá virkt, þegar uemacs fer af stað og (2) lyklaborðið hætti að vera virkt, þegar uemacs er hætt. (Athugið, að ^[táknar escape, sem má setja inn með því að nota quote-charater (^Q) í emacs).

Tilsvarandi ská fyrir mg heitir .mg, en er talsvert frábrugðin í uppsetningu, þótt sömu möguleikar séu til staðar. Dæmigerð .mg skrá getur litið þannig út :

```
(auto-fill-mode)
```

```
   (global-set-key "
^X
^M"        'save-buffer)
   (global-set-key "
^X
^S"        'save-buffers-kill-emacs)
   (global-set-key "
eG"   'goto-line)
```

Þar sem mg er eftirlíking af GNU Emacs, er .mg eins upp sett og .emacs, en sú síðarnefnda er notuð fyrir GNU.

Í Unipress Emacs er notuð skrá að nafni .emacs_pro og getur litið þannig út :

```
   (bind-to-key "ESC-prefix" "
e[23 ")
   (bind-to-key "quote-character" "
eOQ")
   (bind-to-key "apropos" "
e[28 ")
   (bind-to-key "set-mark" "
^[[4 ")
   (read-abbrev-file "/u1/gunnar/.abbrev")
```

Hér er verið að binda vt220-lykla, hlaða inn styttingaskrá ofl. Í Unipress Emacs inniheldur .emacs_pro skráin MLisp forrit og leyfir mjög mikla fjölbreytni, m.a. að hlaða sjálfvirkt inn mismunandi lyklaskilgreiningum eftir því, hvaða skjár er notaður.

Af ofangreindu má sjá, kosti og galla Emacs-fjölskyldunnar. Í fyrsta lagi er unnt að breyta eigin vinnuumhverfi eins og hvern lystir, en í öðru lagi er ansi mikill munur á hvernig það er gert, eftir því hvaða Emacs er notaður.

5.6 Info (Gnu Emacs og Unipress Emacs)

Stóru emacsarnir eru með innbyggt hjálparforrit, 'info', sem er geysiöflugt. Kallað er á 'info' með <esc>-Xinfo innan í emacs. Til að komast út úr Info aftur er notað ^C, en hjálp um info fæst ef ritað er h meðan info er í gangi. Kerfið er valmyndadrifið að fullu og inniheldur allar upplýsingar um emacs.

Yfirleitt eru meiri upplýsingar fáanlegar í info heldur en í handbókum (að vísu er GNU Emacs handbókin um 300 bls og ætti að nægja flestum).

Innan Info er unnt að velja atriði skv valmynd, fara niður í "info-trénu", fara upp, til hliðar osfrv. Einnig er unnt að bæta við upplýsingum og má þannig bæta inn staðbundnum leiðbeiningum.

5.7 (M)Lisp (Gnu Emacs og Unipress Emacs)

Upphaflegi emacsinn var skrifaður í Lisp og honum fylgdu forritunarmöguleikar í skyldu máli, MLisp (Mock Lisp). Þeirri venju er fram haldið í Unipress Emacs, en í GNU Emacs er boðið upp á forritun í Lisp.

Þannig er hægt að skrifa forrit í Lisp, sem síðan má gefa nafn, geyma í skrá og framkvæma hvenær sem henta þykir. Slíkt forrit má tengja við lykil og gera þannig að innbyggðri skipun í Emacs.

Allar skipanir í Emacs eru tiltækar í Lisp-forritunum.

Athuga ber, að auk Lisp-möguleikans er auðvitað hægt að nota lyklaborðsforritun (keyboard macro), sem er oft nægilegt, en ekki nærri eins öflugt.

Þegar verið er að skrifa MLisp forrit, ellegar gefa skipanir, sem á eftir að nota oft er einfaldast í Unipress Emacs að nota þau vinnubrögð að gefa skipanirnar af lyklaborðinu en láta Emacs um að smíða MLisp forritið. Þetta er gert með skipuninni

```
M-X start-generating-mlisp
```

Þannig verður aðferðin :

1. Setja Emacs af stað.
2. Koma upp tveimur gluggum.
3. Gefa "start-generating-mlisp" skipunina í öðrum glugganum.
4. Fara í hinn gluggann.
5. Framkvæma skipanirnar.
6. Gefa "stop-generating-mlisp" skipunina.
7. Geyma MLisp kóðann í skrá.

Síðan má setja inn "defun" skipun til að gefa forritinu nafn, setja inn "load" skipun í .emacs_pro skrána til að ná í forritið þegar Emacs er settur af stað, og að lokum má setja "bind-to-key" skipun í .emacs_pro til að forritið sé aðgengilegt gegnum einn takka.

5.8 Notkun með make (Gnu ofl)

Innan í alvöru-emacs má gefa skipanirnar :

<esc>-Xcompile kallar á make
<esc>-Xnext-error fer með bendil í næstu villu

Þegar **compile** skipunin er gefin, kallar emacs á make (sjá kafla 9), sem sér um að þýða forrit. Ef villur koma í þýðingu er **next-error** notað til að fara í næstu villu. Emacs sér um að fletta milli skráa osfrv.

Hér þarf því aldrei að "fara út úr" Emacs, sem er mikill kostur því Emacs er stórt forrit, sem tekur talsverðan tíma að komast af stað.

5.9 Emacs notendahandbók.

Hér á eftir verður farið kerfisbundið í gegnum einstök atriði Emacs, svo sem glugga, skrár, leitarskipanir osfrv.

5.9.1 Hvað eru skipanir ?

Venjulegur texti er settur inn óbreyttur, eins og hann kemur frá lyklaborðinu. Í Emacs er talað um að stafalyklarnir séu bundnir við "self-insert" skipunina – m.ö.o. þegar ýtt er á stafinn 'a', er framkvæmd skipunin "self-insert", sem setur stafinn 'a' inn í skrána, sem verið er að breyta.

Ef haldið er niðri "control"-takkanum, meðan ýtt er á staf, sendir lyklaborðið frá sér annað sett af táknum, "control-stafi". Þeir stafir eru tengdir (bundnir) skipunum og mynda grunnskipanirnar í Emacs, t.d. færsluskipanir, skipanir til að geyma breyttan texta, o.s.frv. Þessar skipanir eru táknaðar með því að rita hatt á undan viðkomandi skipun, t.d. táknar ^A að halda skuli niðri control takka og ýta á A (lítið eða stórt). Með þessu móti má binda flestar tilfærslu- og leitarskipanir við einn lykil.

Ekki eru nærri nógu margir stafir á lyklaborði til að nægi fyrir þær skipanir, sem nota þarf dags daglega. Á sumum erlendum lyklaborðum er til svonefndur "meta"-lykill, sem vinnur á svipaðan hátt og "control"-lykillinn, þ.e. lætur lyklaborðið senda alveg nýtt stafasett. Þar sem fæst lyklaborð hafa "meta", eru metaskipanir yfirleitt framkvæmdar með því að nota <escape> sem forskeyti. Þannig er M-A framkvæmt með því að ýta fyrst á <escape>-takkann og síðan á A (stórt eða lítið).

Þar sem enn vantar mikið upp á að hægt sé að binda allar skipanir við meta- og control-lykla, eru leyfð forskeyti, og þá yfirleitt notað ^X.

Unnt að gefa skipun með fullu nafni, með því að rita fyrst M-X og síðan fullt nafn á skipuninni. Þessi aðferð er heldur seinleg og er því yfirleitt aðeins notuð, ef þörf er á, t.d. fyrir lítt notaðar skipanir.

Hægt er að auka á skipanaforðann með því að pikka inn skipanarunu af lyklaborðinu og nota "begin-macro" (^X() og "end-macro" (^X)) til að geyma rununa. Einnig er hægt að forrita talsvert, gefa forritunum nafn og binda þau síðan við lykla.

Að lokum hafa stærri emacsar innbyggt forritunarmál, Lisp eða MLisp.

5.9.2 Staða og tilfærsla.

Á hverjum tíma sýnir Emacs bendilinn (cursor) á skjánum. Bendillinn gefur til kynna hvar texti lendir ef hann er settur inn. Textinn lendir milli stafsins sem bendillinn er á og næsta stafs á undann. Þessi staðsetning, milli bendils og næsta stafs á undan, er nefnd púnktur (dot). Ef staf er eytt með "delete-next-character" (^D), þá er eytt stafnum undir bendlinum, þ.e. næsta staf fyrir framan púnkt. Ef staf er eytt með "delete-previous-character" (<delete> eða <backspace>), þá er eytt stafnum fyrir aftan púnkt. Þannig er augljós ástæðan fyrir nöfnunum "previous-" og "next-" "character".

Grunnskipanir til að færa sig til í texta eru:

fram um staf	forward-character	^F
afturábak um staf	backward-character	^B
upp um línu	previous-line	^P
næsta lína	next-line	^N

Athuga ber að nöfn skipananna gefa til kynna, við hvaða staf skipunin er bundin (^F-**F**orward).

Skipunin "set-mark" (venjulega tengd ^@, en stundum ^<bil> ellegar M-@) setur "merki" á þann stað, sem púnkturinn er. Síðan má færa bendilinn hvert sem er, en merkið er kyrrt á sínum stað. Skipunin "exchange-point-and-mark" (^X^X) víxlar síðan á púnkti og merki.

Þetta er m.a. notað til að flytja texta milli staða, sbr 8.9.4.

Tilfærsla er ekki eingöngu bundin við færslu um einn staf eða línu. Einnig er unnt að fara í enda línu með ^E, og í upphaf línu með ^A. Til að fara fram og aftur um málsgrein er notað <esc>-N og <esc>-P (í uemacs, en <esc>-[og <esc>-] í GNU).

Skipanir, sem voru nefndar í kaflanum

fram um staf	forward-character	^F
afturábak um staf	backward-character	^B
upp um línu	previous-line	^P
niður um línu	next-line	^N
víxla á punkti og merki	exchange-point-and-mark	^X^X
eyða fyrri staf	delete-previous-character	<delete>
eyða næsta staf	delete-next-character	^D
setja merki	set-mark	M-
		M-.
		^@
eyða merkjum í svæði	unmark-buffer	M-
fram um málsgrein	next-paragraph	M-N
aftur um málsgrein	previous-paragraph	M-P

5.9.3 Að lesa og skrifa skrár.

xxxx yyyy zzz

Skipanir, sem voru nefndar í kaflanum

beginning-of-file	M-<
change-file-name	^XN
end-of-file	M->
execute-file	
find-file	^X^F
insert-file	^X^I
read-file	^X^R
save-file	^X^M
	^X^S
view-file	^X^V
write-file	^X^W

5.9.4 Gluggar, svæði og skrár.

Skipanir, sem voru nefndar í kaflanum

beginning-of-file	M-<
change-file-name	^XN
end-of-file	M->
execute-file	
find-file	^X^F
insert-file	^X^I
read-file	^X^R
save-file	^X^M
	^X^S
view-file	^X^V
write-file	^X^W
delete-other-windows	^X1
delete-window	^X0
grow-window	^X^
	^XZ
move-window-down	^X^N
move-window-up	^X^P
next-window	^XO
previous-window	^XP
resize-window	^XW
restore-window	
save-window	
shrink-window	^X^Z
split-current-window	^X2
case-region-lower	^X^L
case-region-upper	^X^U
copy-region	M-W
kill-region	^W

5.9.5 Að eyða og flytja texta.

Fyrstu skipanir til að eyða texta eru að sjálfsögðu delete-next-character (^D) og delete-previous-character (^H)

Rétt er að nefna einnig

delete-next-word	
delete-previous-word	M-^H
	M-^?

Sú fyrri er yfirleitt bundin M-D, en því miður tapast sú skipun þegar örvatakkar eru notaðir, sem oft senda <escape>-D (t.d. vt52 og hp skjáir).

Til að flytja almennt svæði er notað "set-mark" (^@) í upphafi þess svæðis, sem á að flytja (eða eyða), þvínæst er bendillinn fluttur í hinn enda svæðisins og notað "kill-region" (^W) til að "drepa" svæðið milli púnkts og merkis. Drepið svæði er geymt á sérstökum stað (kill buffer) og hægt að setja inn á öðrum set með "yank" (^Y) skipuninni.

Til að flytja nokkrar aðliggjandi línur er notað

kill-to-end-of-line	^K

sem "drepur" línuna, þ.e. flytur hana í "kill buffer". Séu nokkrar línur drepnar í röð eru þær settar saman í "kill buffer". Síðan er notað "yank" (^Y) til að ná í línurnar.

delete-blank-lines	^X^O
delete-buffer	^XK
delete-mode	
delete-global-mode	M-^M
delete-other-windows	^X1
delete-window	^X0
copy-region	M-W
kill-paragraph	M-^W
kill-region	^W

Skipanir, sem voru nefndar í kaflanum

Eyða næsta staf	delete-next-character	^D
Eyða fyrri staf	delete-previous-character	^H
		^?
Eyða næsta orði	delete-next-word	
Eyða fyrra orði	delete-previous-word	M-^H
		M-^?
Eyða auðum línum	delete-blank-lines	^X^O
Eyða xxx	delete-buffer	^XK
	delete-mode	
	delete-global-mode	M-^M
Eyða öðrum gluggum	delete-other-windows	^X1
Eyða glugga	delete-window	^X0
Flytja svæði	copy-region	M-W
Eyða málsgrein	kill-paragraph	M-^W
Eyða svæði	kill-region	^W

5.9.6 Leitar- og breytingaskipanir.

Algengustu leitarskipanirnar eru

search-forward ^S
search-reverse ^R

Þessar skipanir gera nákvæmlega það sem nafnið gefur til kynna – leita að föstum strengjum í skrám, afturábak eða áfram.

Heldur flóknari eru skipanirnar

incremental-search ^XS
reverse-incremental-search ^XR

en þær leita "incremental", þ.e. leita að strengnum jafnóðum og hann er sleginn inn. Einnig eru til ýmsar útgáfur af "regular-expression-search-forward", en þá er leitað að strengjum og leyft að nota sértákn eins og í vi og ed (sbr kafla 10).

Breytingar eru framkvæmdar ýmist með

replace-string M-R

en þó er mun öruggara að nota

query-replace-string M-^R
M-Q

Skipunin "query-replace-string" (M-Q) framkvæmir breytingar eftir að hafa fengið staðfestingu notanda í hverju tilviki fyrir sig. Gefinn er upp gamall strengur og nýr, og í hvert sinn sem Emacs finnur gamla strenginn er

spurt :

.nf Replace 'gamall strengur' with 'nýr strengur'? .fi

Yfirleitt er þá ýtt á <bil> (játandi) eða 'n' (neitandi), en ef ritað er eitthvert ógilt tákn, kemur listi yfir möguleikana :

.nf (Y)es, (N)o, (!)Do rest, (U)ndo last, (^G)Abort, (.)Abort back, (?)Help: .fi

Athugið, að einn munurinn á alvöru emacs og eftirlíkingu er, að alvöru emacs leyfir notanda "recursive edit", sem þýðir í þessu tilviki, að þegar query-replace bíður eftir svari, er unnt að fá leyfi til að skreppa inn í nýjan emacs, Þar er unnt að breyta hverju sem hver vill, kveðja síðan þann emacs (með ^C) og halda áfram í "query-replace".

Skipanir, sem voru nefndar í kaflanum

search-forward	^S
search-reverse	^R
incremental-search	^XS
reverse-incremental-search	^XR
query-replace-string	M-^R
	M-Q
replace-string	M-R

5.9.7 Hamskipti.

Emacs er alltaf í einhverjum ákveðnum "ham", sem ræður því, hvernig sumar skipanir hegða sér. Þannig er yfirleitt til "text-mode", sem segir Emacs m.a. að ef lína er orðin of löng þegar kemur að orðabili, þá á að skipta niður í næstu línu fyrir neðan. Stærri emacsar kunna á ýmis forritunarmál, t.d. C, Pascal og Lisp og vita hvernig á að draga inn línur osfrv.

Text-mode er nefnt "**WRAP**-mode í uemacs.

Í GNU eru til miklu fleiri hamir, t.d. pascal, c, nroff, text, lisp, vi osfrv.

5.9.8 Styttingar – Abbrev mode (GNU og Unipress).

Hægt er að nota skipunina "define-global-abbrev" í Unipress Emacs til að skilgreina styttingar á algengum orðum, t.d.

.nf define-global-abbrev mri phrase: marine research institute .fi

og þá sér Emacs um eftirfarandi breytingar :

Innslegið	Verður
mri	marine research institute
MRI	Marine Research Institute
Mri	Marine research institute

5.10 Að bjarga sér í Emacs

Aðalvandinn við Emacs er fjölbreytnin. Unnt er að breyta Emacs eins og hverjum þóknast. Auk þess eru nokkrar útgáfur á markaðnum, sem ekki eru allar eins.

Lausnin felst í því sama : Hver notandi getur breytt sínum Emacs eins og honum sýnist. Til að komast af stað er þó nauðsynlegt að finna grunnskipanirnar, en síðan þarf notandinn eingöngu að geta bjargað sér um upplýsingar innan í emacsinum.

Engum lifandi manni dettur í hug að reyna að læra utanað allar Emacs skipanir. Eini möguleikinn til að vinna skynsamlega í Emacs er að kunna það sem nauðsynlegt er hverju sinni og geta síðan bjargað sér og fengiðaðstoð með ýmsum hjálparskipunum.

Engum lifandi manni dettur í hug að syna að læra utanað allar Emacs skipanir. Eini möguleikinn til að vinna skynsamlega í Emacs er að kunna það sem nauðsynlegt er hverju sinni og geta síðan bjargað sér og fengið aðstoð með ýmsum hjálparskipunum.

Uemacs skipun	Uemacs binding	GNU skipun	GNU binding
apropos	Ekki bundið	apropos	^H^A
describe-key	^X?	describe-key-briefly	^HC
describe-key	^HK		
help	M-?	Ekki til	(sbr info)
Ekki til		info	Ekki bundið
describe-bindings	Ekki bundið	describe-bindings	^H^B

Þannig má rita fyrst "<escape> X apropos", fá frá emacsinum spurninguna "Apropos string:", og rita þá t.d. "window" til að fá lista yfir allar gluggaskipanir.

Ef ýtt er óvart á takka og ekki ljóst hvað hann gerði má nota describe-key skipunina til að fá lýsingu.

Að auki má í flestum emöcsum rita <escape> í miðri "extended" skipun til að klára hana og rita ? í miðri skipun til að fá lista yfir, hvernig hægt er að ljúka skipuninni (gengur ekki í uemacs).

Ekki sakar að nefna "undo" (^_) skipunina í GNU, en hún leyfir notanda að taka burt breytingar, sem hafa verið gerðar á skrá. GNU geymir undanfarnar breytingar (alls um 8000 stafi) og unnt er að rita ^_ oft, til að taka burtu margar undanfarnar breytingar á skránni, allt þar til hún er orðin eins og hún var upphaflega. Síðan er einnig unnt að hætta við "undo".

Hægt er að nota <bil> í "extended" skipun til að klára það orð, sem verið er að rita. Þannig ritar enginn fullum fetum "<escape>-Xdescribe-

bindings", heldur "<escape>-Xdesc<bil>bind<bil>" og emacsinn klárar jafnóðum eins mikið og hann getur.

6 MAKE - AÐSTOÐ VIÐ FORRITUN

6.1 Inngangur -hvað gerir make?

Make er forrit, sem aðstoðar forritara (og aðra) við að halda utanum kerfi, sem eru samansett úr forritum í mörgum skrám. Ef sumum forritum er breytt, sér **make** um að þýða nákvæmlega þau forrit, sem hafa breyst.

Forritið **make** gerir þannig stóran hluta af því, sem skipanaskrár gera í öðrum stýrikerfum, en þar er ein mjög algeng notkun skipanaskráa einmitt að sjá um þýðingar á forritum.

Sér til aðstoðar notar **make** skrá sem nefnist **Makefile** (eða **makefile**). Innihald skrárinnar lýsir samböndum milli forritaskráa (t.d. **.c**-skráa), skilgreiningaskráa (venjulega með undirnafni **.h**), og þýddra skráa.

Makefile er skrá, sem inniheldur tvær gerðir lína, línur sem gefa *sambandslýsingu* og línur sem gefa *skipanalýsingu*.

Sambandslýsingarlína er einfaldlega lína, sem segir til um, hvernig ein skrá er háð öðrum. Slíkar línur innihalda *háðu* skrána fremst, þá tvípunkt (:) og síðan *óháðu* skrárnar, þannig :

```
háð :  óháð1 óháð2 ...
```

Make les slíka línu og notar hana til að ákveða, hvort háða skráin sé orðin *úrelt*, en háða skráin telst úrelt ef hún er eldri en einhver óháðu skránna. Við venjulega forritun má nefna, að vélarmálsskráin test.o er úrelt þegar c-forritinu test.c hefur verið breytt. Þá er test.o orðin eldri en test.c og því kominn tími til að endurþýða test.c og smíða nýja test.o.

Skipanalýsingarlínur lýsa hvernig háða skráin er endurnýjuð, ef niðurstaða úr sambandslýsingarlínu er að háða skráin sé úrelt. Skipanalýsingarlínur eru alltaf innsdregnar með einu tabbi, og geta verið margar slíkar línur fyrir eina sambandslýsingu.

Dæmið um test.c og test.o verður þannig :

```
test.o :  test.c
cc -c test.c
```

Athugið, að dæmið er alltof einfalt, því **make** veit allt um svona C-þýðingar og alveg óþarft að gefa þessar upplýsingar í **Makefile**.

En **make** er best útskýrt með dæmum og við setjum fram nokkur dæmi á næstu síðum.

6.2 Einfalt dæmi.

Forritið "exec" er búið til úr forritunum a.c, b.c og c.c með þýðingunum

```
cc -c a.c
cc -c b.c
cc -c c.c
cc -o exec a.o  b.o  c.o
```

Make veit, hvernig á að framkvæma C-þýðingar, svo það þarf aðeins að vita, að exec er búið til úr a.o, b.o og c.o, með síðustu skipuninni. Þetta er gert með því að smíða skrá með nafninu **Makefile**, sem inniheldur:

```
exec:      a.o  b.o  c.o
cc  -o  exec  a.o  b.o  c.o
```

Í fyrsta skipti sem **make** er keyrt, eru öll forritin þýdd og síðan tengd:

```
cc  -c  a.c
cc  -c  b.c
cc  -c  c.c
cc  -o  exec  a.o  b.o  c.o
```

Ef forritinu a.c er nú breytt og síðan ritað "**make**", er aðeins framkvæmt það sem þarf:

```
cc  -c  a.c
cc  -o  exec  a.o  b.o  c.o
```

Make kannaði hvenær forritum hafði verið breytt, komst að því að a.c hafði verið breytt á eftir a.o og því þurfti að þýða aftur. Þar með breyttist a.o og því þurfti að tengja aftur. Innbyggðu upplýsingarnar um hvernig a.o er háð a.c jafngildir að í **Makefile** sé bætt við línunum:

```
a.o :  a.c
cc -c  a.c
```

Ef upp kemur villa í þýðingu einhvers forrits hættir **make** að sjálfsögðu vinnslu (nema ef sérstaklega er beðið um að **make** reyni að halda áfram, t.d. með **make -k**).

6.3 Aðeins flóknara dæmi.

Forritið abc er smíðað úr x.c, y.c og z.c, en í öllum þeim skrám er vísað (með #include) í skrána defs.h. Að auki er vísað í skrána externs.h í y.c og z.c. Eftirfarandi **Makefile** nægir í þessu tilviki:

```
abc :  x.o  y.o  z.o
cc  -o  abc  x.o  y.o  z.o
x.o :  defs.h
y.o  z.o :  defs.h externs.h
```

Ef nota skal aukalega t.d. "-O" við þýðingu, má bæta inn línunni

```
CFLAGS= -O
```

fremst í **Makefile.**

6.4 Fleiri en eitt forrit.

Ef smíða skal forritin xyz og abc, ber að athuga, að skipunin "**make**" býr aðeins til það, sem fyrst kemur fyrir í **Makefile**. Því er ekki nóg að telja aðeins upp, hvernig á að búa til xyz og abc, það þarf einnig að taka fram, að heildarþýðing er háð báðum:

```
kerfi:abc xyz
touch kerfi
abc:    a.o b.o
cc -o abc a.o  b.o  -lm
xyz:    x.o  y.o  z.o
cc -o xyz x.o  y.o  z.o
```

```
y.o:    def.h
```

6.5 Reglur.

Eins og getið er að ofan, veit **make** ýmislegt, sem ekki þarf að taka fram. T.d. þekkir **make** eftirfarandi endingar á skrám:

```
.c      C
.f      Fortran
.l      Lex
.r      Ratfor
.y      Yacc
```

og veit, hvernig á að þýða viðkomandi forrit.

6.6 Breytur.

Make hefur talsvert safn af breytum, sem er unnt að endurskilgreina til að breyta gangi þýðingar.

T.d. er **FC** nafnið á Fortran þýðandanum, og er venjulega **f77**. Ef nota skal **fc** í staðinn, má rita

```
FC=fc
```

framarlega í **Makefile**.

Þegar þýða skal mörg forrit, er óþægilegt að þurfa að telja upp öll forritin oft. Þá er gjarnan settur upp listi fremst í **Makefile**, yfir öll forrit, sem á að þýða og síðan vísað í listann, t.d.

```
OBJ= x.o  y.o  z.o
xyz:  $(OBJ)
cc -o xyz $(OBJ)
y.o:    def.h
```

(sbr dæmið að ofan). Hér er breytan **OBJ** látin geyma listann yfir öll þýdd forrit og síðan er vísað í innihald breytunnar með **$(OBJ)**.

6.7 Að lokum.

Að lokum ber að nefna, að hægt er að skrifa margar bækur um **make**. Eins og fyrr er nefnt kann **make** t.d. á miklu meira en C þýðingar. Innbyggðar reglur í **make** segja til um, hvernig á að þýða Fortran, hvað á að gera við lex- og yacc-skrár osfrv. Einnig veit **make** um SCCS kerfið. Við bendum á aðrar Unix bækur og svo greinar um **make**, sem fylgja með Unix kerfinu.

Hér hefur verið lýst þeim þáttum **make**, sem líklegast er að komi venjulegum notendum Unix til góða. Rétt er að enda kaflann með því að minnast á dæmi.

Þessari handbók var lengi haldið við á mörgum stöðum, m.a. á smátölvu, sem ekki hefur nroff eða troff. Hins vegar er vélin með Unix og tengd ISNET. Þegar kafla er breytt, á að senda hann inn á vél sem heitir **gst** og er líka tengd ISNET. Þar er hægt að prenta handbókina.

Kaflarnir í handbókinni eru geymdir í aðskildum skrám. Einnig er til tóm skrá, UNIX. Ef einhverjum kaflanna er breytt, verður viðkomandi kafli nýrri en skráin UNIX. Þegar **make** er keyrt næst á að senda alla breytta kafla inn á vélina **gst**, með tölvupósti. Eftirfarandi **Makefile** gerir einmitt þetta.

```
UNIX:   chap*
  sh -c 'for i in $?
e
  do
e
  rmail to-manual@gst < $$i
e
  done'
  touch UNIX
```

Hér er notað táknið **$?**, sem í **make** táknar listann yfir allar þær óháðar skrár (í shap*), sem eru nýrri en háða skráin (UNIX). Skipunin, sem á að framkvæma er svolítið flókin, og á mörgum línum. Hér er um að ræða skelskipun (shell-script), sem verður t.d.

```
for i in chap00 chap09 chap07
  do
  rmail to-manual@gst < $i
  done
```

(ef þarf að senda chap00, chap09 og chap07). Athugið að þar sem **$** hefur sérstaka merkingu í **make**, þarf að rita **$$i** til að **make** sendi **$i**.

Fyrir þá, sem þekkja **smail** er rétt að benda á, að á vélinni **gst** er notandinn to-manual ekki til, heldur er to-manual **alias** fyrir forrit, sem tekur skeytið (kaflann) og setur hann á rétt vinnusvæði á þeirri vél.

Svo er **make** keyrt á nóttunni á **gst**, þannig að ef kafla er breytt á mánudagskvöldi á smátölvunni, þá er ný prentuð handbók tilbúin á þriðjudagsmorgni.

.nr H1 9

7 VI, GREP OG SED EÐA: STRENGJAMEðHÖNDLUN

7.1 Aðdragandi

Tilgangurinn með þessum kafla er að kynna hugtakið "regular expression", sem er geysimikið notað í strengjameðhöndlun í Unix.

Til að notandi nái valdi á forritum eins og grep, sed og awk er nauðsynlegt að góð þekking sé fyrir hendi á strengjameðhöndlun.

Einfaldast er að læra strengjameðhöndlun með notkun ed (eða vi), en hún nýtist síðan við fjölmörg önnur Unix tól

7.2 Grundvallaratriði ed (og vi).

Ritþórarnir ed og vi fylgja öllum Unix stýrikerfum og það eitt gerir Unix notendum nauðsynlegt að kunna undirstöðuatriði þeirra.

7.2.1 Ed.

Ed er "línu -editor", þ.e. skipanir í ed vinna á einni línu í einu. Ekki er unnið með skjámyndir eða örvatakka, heldur ritaðar skipanir, sem á að framkvæma. Skipanir má gefa meðan ed er í **skipanaham,** en í **innsetningarham** er texti sleginn inn.

Nokkrar skipanir:

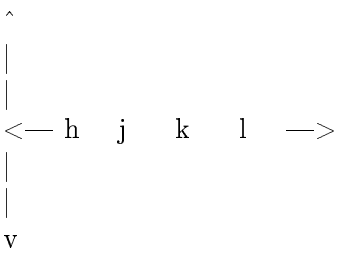
a	append	Bæta texta við aftan við núverandi línu. Fer úr skipanaham í innsetningarham.
s/xyz/asc/	substitute	Breyta strengnum xyz í abc í línunni.
w	write	Skrifa skrá með breytingunum.
q	quit	Hætta vinnslu í ed.

Til að komast úr innsetningarham í skipanaham er sett inn lína sem inniheldur aðeins punkt ("."), m.ö.o. ýtt á <return>, þá punkt og svo <return>.

7.2.2 Vi vs ed

Vi er skjá ritþór (screen-editor), þ.e. vi sýnir texta, sem unnið er með, á öllum skjánum. Í skipanaham má færa bendilinn til með skipununum h, j, k, l. Oft ganga örvatakkarnir einnig. **Varúð:** ekki nota örvatakkana í innsetningarham (insert mode). Til að bæta inn texta fyrir aftan þann stað, sem bendill er á, er notað "a" og fer vi þá í innsetningaham. Vi fer svo tilbaka í skipanaham, þegar ýtt er á <escape> takkann.

Tilfærsla :



Komast í innsetningarham :

Fyrir aftan stafinn	**a**
Fyrir framan stafinn	**i**

Komast úr innsetningarham :

<escape>

Einnig gildir, að skipanir, sem unnt er að gefa í ed er einnig unnt að gefa í vi, en þá er settur tvípunktur (":") fyrir framan þær, t.d.

:s/xyz/abc/	Breyta xyz í abc innan þeirrar línu, sem bendillinn er í.
:w	Skrifa skrána.
:q	Hætta.
:wq	Skrifa og hætta svo.

Til að komast út úr vi má því nota <esc> fyrst og svo :wq ef á að skrifa, eða <esc> og svo :q (eða :q!) til að hætta bara.

Héðan í frá verður rætt um skipanir eins og þær eru gefnar í ed, en ljóst ætti að vera hvernig þær yfirfærast í vi.

7.2.3 Sed

Sed (stream editor) er ritþór með skipanir mjög svipaðar og ed, en vinnur á einni textalínu í einu. Þannig getur sed unnið á mjög stórum skrám og er

oft notað í skipaskrám (shell scripts).

Sed er venjulega notað þannig:

```
sed 'skipanir' <skrá1 >skrá2
```

Dæmi:

```
sed 's/a/b/g
3s/xyz/abc/'    <skrá   >file
```

Breytir "a" í "b" í öllum línum og allstaðar í hverri línu. Breytir líka xyz í abc í þriðju línu. Ef xyz kemur tvisvar fyrir í þriðju línu, er aðeins fyrra tilfellinu breytt.

7.3 Strengir og strengjaskipanir

7.3.1 Strengir í ed

Staðsetning tiltekinna lína er aðeins hálf sagan, hinn helmingurinn er staðsetning og gerð strengja innan lína. Hér höfum við geysilega möguleika, t.d. má tiltaka upphaf línu með '^', enda línu með '$', litlir stafir (enskir) eru '[a-z]'. Þannig má rita 's/^a/b/' til að breyta 'a' í 'b', ef a-ið er fremst í línunni, einnig 's/^[a-z]/xxx/', til að breyta litlum staf fremst í línu í 'xxx'.

7.3.2 Staðsetningar í ed

Staðsetning í skrá er teygjanlegt hugtak þegar ed (og vi, sed osfrv) eru annars vegar. Þannig má vísa í tiltekna línu á fjölmarga mismunandi vegu:

tala	vísar nákvæmlega í þá línu
/strengur/	vísar í næstu línu, sem inniheldur streng

til dæmis má gefa skipunina

```
15s/abc/xyz/
```

en þar er fyrirskipað, að breyta skuli abc í xyz í fimmtándu línu. Öllu flóknari er möguleikinn á að vísa í línu með því að tiltaka streng, sem línan á að innihalda. Þannig þýðir

```
/AAA/d
```

að eyða skuli næstu línu, sem inniheldur strenginn 'AAA'.

Sérstaka merkingu hafa punktur ('.'), sem vísar í þá línu, sem verið er að vinna með, og dollar ('$'), sem vísar í síðustu línu skráarinnar.

Til að auka möguleika á að tiltaka staðsetningar, eru gefnir möguleikar á að tilnefna staðsetningar frá og með ákveðinni línu, til og með annarrar línu. Að sjálfsögðu má nota strengi til að lýsa annarri eða báðum línunum.

3	Vísar í þriðju línu.
1,5	Vísar í línur eitt til fimm.
1,$	Öll skráin (fyrsta til síðasta).
.,$	"héðan og niðurúr".
/joe/	Næsta lína sem inniheldur strenginn "joe".
/joe/,$	Frá línu með "joe" niður að enda.
/joe/, /mike/	Frá næstu línu með "joe" að þar næstu með "mike".
/^a/	Lína sem byrjar á a.
/^[a-z]*$/	Lína, sem inniheldur aðeins litla stafi (0 eða fleiri).

7.3.3 Breytingar

Með því að tengja staðsetningalýsingarnar í 10.3.2 við strengjalýsingarnar í 10.3.1, má fá ansi öflugar breytingarskipanir og leyfum við lesandanum að spreyta sig á eftirfarandi:

s/a/b/	Breyta fyrsta a í b í línunni.
s/a/b/g	Breyta a í b alls staðar í línunni.
1,5 s/a/b/g	Breyta a í b alls staðar í línum 1-5
/^[a-z]*$/s/^\([^a]*a*\).*/\1/	Í línum, sem innihalda litla stafi frá upphafi til enda skal taka allt frá og með streng sem byrjar á runu af a og breyta þannig að a-runan standi eftir.

7.4 Dæmi um notkun sed og grep.

7.4.1 Rúnna í 5 cm flokka.

Ef inntaksskrá inniheldur marga dálka, aðskilda með tabbi, þá rúnnar eftirfarandi skipun fyrsta dálkinn í 5cm flokka.

sed 's/^\([0-9]*\)[0-4] /\12 / s/^\([0-9]*\)[5-9] /\17 /'

Inntak :

133 3
21 34
36 21
34 543

Úttak :

132 3
22 34
37 21
32 543

Þar sem þetta er hugsanlega ekki alveg augljóst, má skýra málið. Lengdarmælingar fiska eru yfirleitt þannig að tölur á bilinu 39.5-40.5 eru skráðar 40 og mælingarnar 40-44 tákna því að upphaflegu gögnin voru á bilinu 39.5-44.5, og miðpunktur þess bils er 42. Því skal breyta tölum sem enda á 0,1,2,3 eða 4 í tilsvarandi tölu, sem endar á 2, en það er gert í fyrri línunni í sed-skipuninni. Strengurinn /^[0-9]*[0-4]/ táknar tölu í byrjun línu, sem endar í 0-4. Fyrri part tölunnar þarf að geyma : /^\([0-9]*\)[0-4] / og sá fyrri partur er aðgengilegur sem \1. Þarmeð er fyrri línan tilbúin.

7.4.2 Símaskrárkerfi með grep.

Hér á eftir fylgja skipanaskrár, sem má nota til að halda utan um mjög einfalt símaskárkerfi.

Fyrst er búið til vinnusvæðið TELNOS undir heimasvæðinu ($HOME). Skipunin addtel bætir inn í skrána og skipunin tel flettir upp.

addtel :

```
cd ; cd TELNOS
echo $* » telnos
```

tel :

```
cd ; cd TELNOS
case $* in
"") cat telnos
break
;;
*)  grep "$*" telnos
break
;;
```

```
esac
```

Dæmigerð notkun gæti verið

```
addtel "Bjorn bjorn Björn Ævarr Steinarsson Hafrannsókn
20240"
```

þannig er Birni bætt inn undir helstu nöfnum, sem hugsanlegt er að honum yrði flett upp með síðar.

Síðan má nota

```
tel bjorn
```

til að fletta upp.

Til að fá bara alla skrána má rita

```
tel ""
```

.bp

8 Awk

9 Hvað er awk?

Til að fyrirbyggja misskilning, þá er nafnið myndað með upphafstöfum höfundanna, Aho, Weinberger, Kernighan og hefur ekkert með notkun forritsins að gera.

Í stuttu máli getur awk unnið á skrá: útreikninga á dálkum; leitað að og breytt strengjum; valið línur eða dálka ofl. Þannig getur awk flest það sem venjuleg gagnagrunnkerfi geta, á einni skrá.

Awk er venjulega keyrt þannig:

awk 'skipanir' <gagnaskrá

þar sem "skipanir" eru safn skipana á forminu

pattern action

"Pattern" er skilyrði, sem lína þarf að uppfylla til þess að "action" sé framkvæmt. Ef "pattern" er sleppt, er aðgerðin framkvæmd á öllum línum.

Ef "action" er sleppt, er framkvæmd aðgerðin "print", þ.e. þær línur eru prentaðar, sem uppfylla skilyrðið.

Venjulega eru inntaksskrár á dálkaformi, þar sem eyður og/eða "több" aðskilja dálka. Vísað er í einstaka dálka sem $1, $2,.... og í alla línuna sem $0. Fjöldi dálka er geymdur sem NF (number of fields) og númer inntakslínu er NR. Að auki eru ýmsar breytur og föll, sem munu útskýrð með dæmum. Reglur um strengi í "pattern" eru eins og í egrep, en að auki má nota $1, $2...., NF, NR ásamt reiknisetningum.

Awk núllstillir allar breytur sjálfkrafa. Breytur og vektorar verða til um leið og vísað er til þeirra. Skilgreininga er aldrei þörf.

BEGIN má nota sem "pattern" og vísar til hluta, sem á að gera áður en innlestur gagna hefst. Á tilsvarandi hátt er skipunin

END action

lýsing á aðgerðum, sem á að framkvæma eftir að innlestri er lokið.

9.1 Nokkur einföld dæmi.

9.1.1 Prenta línur, sem innihalda joe:

awk '/joe/' <skrá

Athugið, að skipunin þýðir í raun

awk '/joe/ print' <skrá

eða

awk '/joe/'print $0' <skrá

Streng-lýsingin milli skástrikanna má vera eins flókin og verkast vill, sbr kaflann um strengi.

9.1.2 Prenta dálk nr. 1 í inntaki:

awk 'print $1' <skrá

Þannig er mjög einfalt að velja tiltekna dálka úr inntakinu. Gert er ráð fyrir að eyður eða "több" sé milli dálkanna.

9.1.3 Prenta dálka 1,2 og summuna af dálkum 3 og 4:

awk 'print $1,$2,$3+$4' <skrá

9.1.4 Reikna og prenta summu fyrsta dálks:

awk 'sum + = $1 END print sum' <skrá

Hér er nokkura athugasemda þörf:

- "sum + = $1" er styttri útgáfa af "sum = sum + $1" (og getur verið fljótvirkari í framkvæmd).

- . Sum er sjálfkrafa skilgreint og núllstillt.

- Ef þessi skipun er gefin beint í csh þarf \ aftast í fyrri línu.

- Aðgerðin sum + = $1 er framkvæmd fyrir allar línur í inntaksskrá, því "pattern" er sleppt. Á eftir síðustu línu er svo "print sum" framkvæmt.

9.2 Forleikur með awk og nokkrum öðrum Unix forritum

Hér á eftir verður sýnt, hvernig má á einfaldan hátt setja upp gagnagrunnskerfi með Unix tólum, án verulegrar forritunarvinnu. Áður hefur verið minnst á helstu gagnavinnslutól í Unix, og í eftirfarandi töflu er sýnt, hvaða Unix forrit má nota til að gera það sem forleiksforritin gera:

Forleikur	Unix tól
project	cut (eða awk)
select	awk (eða grep)
compute	awk (eða sed)
union	cat
sorttable	sort
jointable	join

9.2.1 project með Unix tólum

Project á að pilla dálka úr skrá, skv nafni dálksins. Þetta er mjög svipað t.d.

```
awk 'print $1,$3,...',
```

ellegar

```
cut -fdálknúmer
```

sem pillar út dálka skv stöðu þeirra (númeri). Til að nota awk eða cut þarf aðeins að umsnúa dálknöfnum í númer.

Project með cut getur orðið þannig :

```
# Implementing the project command with Unix cut.
opt="-f "
outcols=""
read inpcols
for i in $*
do
colnr=1
for j in $inpcols
do
if [ X$i  = X$j ]
then
outcols="$outcols,$colnr"
fi
colnr=‘expr $colnr + 1‘
done
done
echo "$*" | sed 's/  */ /g'
cut $opt$outcols
```

en ef notað er awk í staðinn má nota :

```
# Implementing the project command with awk.
outcols=""
read inpcols
for i in $*
do
colnr=1
for j in $inpcols
do
if [ X$i  = X$j ]
then
outcols="$outcols $$colnr"
fi
colnr=‘expr $colnr + 1‘
done
done
echo "$*" | sed 's/  */ /g'
```

```
awklist=‘echo $outcols | sed ’s/  */,/g’‘
awk ’BEGINFS=" ";OFS=" "
print ’"$awklist"’’
```

Þar sem project er mjög mikið notað, er það endurskrifað í C og ofangreindar útgáfur ekki notaðar. Munurinn á ofangreindu og útgáfunni í C felst í tvennu : C útgáfan hefur þann galla að geta aðeins ráðið við takmarkaðan fjölda dálka í úttaki (takmarkaður fjöldi á skipanalínu. Á hinn bóginn geta ofangreindar tvær útgáfur aðeins pillað út dálka sem eru í skránni – C forritið smíðar nýjan dálk, ef beðið er um dálk, sem er ekki í skránni.

9.2.2 Math

Skipunin 'math $<$ skrá' skilar fjölda lína, meðaltali, staðalfráviki ofl fyrir hvern dálk í inntaki.

Til að sjá, hvernig skal byggja upp slíka skipun, er rétt að minna á hvað þarf að koma út :

$$\text{Mean} := \bar{x} := \frac{\sum x}{n}$$

$$\text{Std. dev.} := \sigma := \sqrt{\frac{\sum x^2 - \sum x^2/n}{n-1}}$$

Við þurfum því að reikna fyrst $\sum x$ og $\sum xsup2$ fyrir hvern dálk. Alls eru NF dálkar og math verður því bara eftirfarandi awk-skipun :

```
awk ’BEGIN FS="        ";OFS=" "
NR==1      print $0"      Type";totf=NF
NR==2      print $0"      --";
for(i=1;i<=NF;i++)
max[i]= -1e64;
min[i]=1e64;
NR>2       for(i=1;i<=NF;i++)
sum[i]+=$(i);
sum2[i]+=$(i)*$(i);
freq[i]++;
if($(i)>max[i])
max[i]=$(i);
if($(i)<min[i])
```

```
min[i]=$(i);
END       for(i=1;i<=totf;i++)
printf("%f\t",freq[i]);
printf("Freq\n");
for(i=1;i<=totf;i++)
printf("%f\t",sum[i]);
printf("Sum\n");
for(i=1;i<=totf;i++)
printf("%f\t",sum[i]/freq[i]);
printf("Mean\n");
for(i=1;i<=totf;i++)
printf("%f\t",sqrt((sum2[i]-sum[i]*sum[i]/freq[i])/(freq[i]-1)));
printf("Stddev\n");
for(i=1;i<=totf;i++)
printf("%f\t",max[i]);
printf("Maximum\n");
for(i=1;i<=totf;i++)
printf("%f\t",min[i]);
printf("Minimum\n");
'
```

Skráin junk inniheldur

Hlutur	Thyngd	Fjoldi	Aths
1	43.2	10	Nóg til
10	21	15	Gott ástand
2	9	5	Þarf að panta
3	10	0	Vantar
5	34	800	Þarf að selja

Skipunin 'project Hlutur Thyngd Fjoldi < junk | math' gefur þá eftirfarandi niðurstöðu á skjáinn :

Hlutur	Thyngd	Fjoldi	Type
5.000000	5.000000	5.000000	Freq
21.000000	117.200000	830.000000	Sum
4.200000	23.440000	166.000000	Mean
3.563706	14.975580	354.460858	Stddev
10.000000	43.200000	800.000000	Maximum
1.000000	9.000000	0.000000	Minimum

9.2.3 Jointable

Jointable setur saman tvær skrár skv sameiginlegum dálki. Þannig gegnir jointable sama hlutverki og Unix join, en vinnur á Forleiks-gögnum.

```
trap "rm tmp$$ haus$$" 2 14 15
opt=""
case $1 in
'-n'|'-a1'|'-a2')
opt=$1
shift
;;
esac
sed 1q < $1 | tr '\012' '      '  >haus$$
sed 's/^[^      ]*      //
1q' < $2 »haus$$
sed 'p
s/[^    ]/-/g' < haus$$
sed '1,2d' < $2 >tmp$$
sed '1,2d' < $1 | join $opt -t" " - tmp$$
rm tmp$$ haus$$
```

Ath. Valmöguleikinn '-n' virkar ekki.

Takið eftir, að jointable kallar bara á Unix join skipunina, eftir að vera búin að ganga frá hausunum í báðum skránum.

9.2.4 union

Union setur allar skrárnar saman í eina, eins og cat, nema hvað hausinn er aðeins tekinn úr fyrstu skránni. Því nægir að gera 'cat' við fyrstu skrána, til að fá hausinn með, en síðan 'tail +3' við afganginn. Þetta er einmitt gert :

```
cat $1
shift
for i
do
tail +3 <$i
done
```

9.2.5 sorttable

Sorttable raðar Forleiks-skrá. Einföld útgáfa er bara svona :

```
read head
read next
echo $head
echo $next
sort
```

Hér eru gefnar tvær lestrarskipanir í skelinni, til að ná í hausinn. Síðan er afgangnum raðað.

Ef þetta er sett í skrána 'sorttable', þá gert 'chmod +x sorttable', þá er unnt að rita 'sorttable < skrá' til að raða skrá með stafrófsröðun. Athuga ber, að echo-skipanirnar eyðileggja többin í inntakinu og því kemur ekki lögleg Forleiksskrá út úr þessari einföldu sorttable-skipun.

Nú má fínpússa sorttable aðeins, og bæta inn möguleikum á :

1. Raða tölulega með '-n'. Ef 'sorttable -n' er notað á bara að bæta '-n' við á sort-línunni.

2. Raða á tiltekna dálka inni í miðri skrá, en ekki bara á fyrsta dálk. Hér þarf aðeins að hugsa, því Unix sort kann á dálka skv númeri, en við viljum nota nöfn. Því þarf að bera saman nafn á skipanalínu við nafn í skránni.

Útkoman verður eftirfarandi sorttable :

```
trap "rm tmp$$ tmp1$$" 15 2 1
opt="-t "
col=""
sortcol=""
for i
do
case $i in
'-n')
opt="$opt -n"
;;
*)
```

```
col="$col $i"
;;
esac
done
sed -n "1,2p
3,\$w tmp$$"  | tee tmp1$$
read columns < tmp1$$
for i in $col
do
colnr=0
for j in $columns
do
if [ X$i  = X$j ]
then
sortcol="$sortcol +$colnr"
fi
colnr=‘expr $colnr + 1‘
done
done
sort $opt $sortcol tmp$$
rm tmp$$ tmp1$$
```

Skipunin "read columns < tmp1$$" lætur skel-breytuna "columns" innihalda öll dálkanöfn í inntaksskránni. Í fyrstu for-skipuninni er "col" látið innihalda öll dálkanöfnin á skipanalínunni.

Síðan eru notaðar tvær for-skipanir til að finna, hvaða dálkar í inntaki tilsvara dálkunum á skipanalínunni. Númer þeirra inntaksdálka er síðan afhent Unix sort-skipuninni til að það forrit geti raðað fyrir okkur.

9.2.6 Check.

Check er forrit, sem kannar hvort Forleiks-skrá sé í innbyrðis samræmi, þ.e. hvort sami fjöldi dálka sé allsstaðar í skránni.

Notkunin er einfaldlega

check skrár

Til dæmis, ef test inniheldur :

Hlutur	Thyngd
1	43.2
10	21
2	9
3	10
5	34

þá gefur 'check test' niðurstöðuna
Table test is ok
Check er bara örstutt :

```
   for i
   do
   sed 's/         /       .              /g
   s/         /       .              /g' < $i |
   awk 'BEGIN FS="        "
   NR==1reclen=NF;error=0
   NR==2if(NF!=reclen)print "Dashes do not match column
heads";error=1
   NR>2 if(NF!=reclen)print "Error in line number ",NR,"fields
",NF," headers ",reclen;error=1
   END if(error==0)print "Table '$i' is ok"'
   done
```

Fyrsta awk-línan segir aðeins að táknið, sem aðskilur dálka er tab.

Næsta lína byrjar á NR==1, sem segir að skipunina skuli framkvæma á fyrstu línu í inntaki. Skipunin er tvíþætt, fyrst "reclen=NF", sem segir að breytan reclen eigi að geyma fjölda dálka í fyrstu línu. Þennan fjölda á síðan að bera saman við fjölda dálka í síðari línum. Næsti hluti skipunarinnar er "error=0", en ef finnst villa neðar í gagnaskránni verður error sett jafnt og 1.

Í awk-línu 3 er vísað í gagnalínu 2 (NR==2) og prentuð villumelding ef dálkafjöldinn passar ekki. Það sama er gert fyrir allar síðari gagnalínur.

Að lokum er prentuð athugasemd um að taflan (gagnaskráin) sé í lagi, ef enn gildir að error-breytan er jöfn núlli.

9.2.7 Dbdict.

Dbdict gefur upplýsingar um dálkabreidd í skrá. Notkun :

```
dbdict < skrá
```

Til dæmis, með 'test' eins og í 11.3.8, þá má rita 'dbdict test' og fá

Atridi	Lengd
Hlutur	6
Thyngd	6

á skjáinn. Dbdict reiknar lengd hvers sviðs og tekur þar með lengd nafnanna líka.

```
awk '   BEGINFS="      "
NR==1for(i=1;i<=NF;i++)
ind[i]=$(i)
lengd[i]=length($(i))
svidafj=NF;
NR>2for(i=1;i<=NF;i++)
if(length($(i))>lengd[i])
lengd[i]=length($(i));
END    print "Atridi   Lengd"
print "---   ---"
for(i=1;i<=svidafj;i++)
print ind[i]"   "lengd[i]
'
```

Hér vantar ýmislegt, t.d. hvort dálkur sé talnadálkur eða textadálkur osfrv, til að dbdict geri allt það sem æskilegt er að fá í gagnagrunnsyfirliti.

9.3 Límmiðar með awk

Eftirfarandi er notað til að skrifa út ICEUUG límmiða hjá Hafrannsókn.

```
   # ********************************************************
   # Umbrotssforrit fyrir limmida.
   # Hafrannsoknastofnunin 1987.  Konráð Konráðsson
   # ********************************************************
   # Tekur inntak a eftirfarandi formi af "standard input"
:
   # fyrirtæki:nafn:gata:borg
```

```
#
# Setur prentara (HP LaserJet) a "manual feed"
awk  '  BEGIN
print ".cf /u1/konni/Priv/man"
print ".po 0"
print ".na"
print ".nf"
print ".ta .5c 7.5c 14.5c"
print ".pl 65"
print ".sp 4"
FS=":"
nafn=nafn"      "$2;
fyrirt = fyrirt"        "$1;
gata = gata"    "$4;
borg = borg"    "$5;
NR % 3 == 0
print nafn ; nafn = ""
print fyrirt ; fyrirt = ""
print gata ; gata = ""
print borg ; borg = ""
print ".sp 5"
END
print nafn
print fyrirt
print gata
print borg
print ".cf /u1/konni/Priv/auto"
' |
preeroff |
eroff -rN4 -p |
posteroff |
lp -or
```

10 Allt í steik.

Ætlunin hér er að lýsa aðeins helstu vandræðum, sem byrjendur lenda í, og lausnum á þeim vandamálum, ellegar benda á, hvar má finna upplýsingar.

10.1 Skjár í Steik.

10.2 Alveg dauður - - ekkert svar.

1. Prófið:
   BREAK
   ^C (control og C saman)
   <delete>
   ^Q
   ^J

   -nokkrum sinnum. Líka ^G ef emacs er í gangi.

2. Ef ekkert gengur í (1), er eitthvað alvarlegt að. Lítið á skjásnúrurnar. Á einsnotanda vél gæti þurft að ræsa aftur. Á fjölnotanda vél má prófa stty sane </dev/tæki

10.3 ^C gefur % frá skel, en ekkert annað gengur.

Prófið :

1. ^C ^J stty sane ^J

10.4 ^D gengur ekki til að logga af.

Prófið að rita "logout", etv. með ^J á undan og eftir.
Sjá nánar 16.1.

10.5 Emacs í steik

10.6 Frosinn

Prófið

1. ^G nokkrum sinnum ("abort")
2. ^L Refresh screen.
3. ^Q

10.6.1 Kemst ekki út úr emacs

Prófið

1. ^C
2. ^X^C
3. <esc> X exit
4. <esc> X exit-emacs
5. <esc> X describe-bindings

10.7 Vi í steik

Prófið <esc> nokkrum sinnum og svo :wq eða :q!. Ath. Ekki gera :wq ef þið hafið eyðilagt skrána. Þessi skipun byrjar á að skrifa út breytingarnar (og skemma þarmeð eintakið á diskinum).

10.8 Skjárinn höktir.

Ef línur koma ein og ein eða hálf í einu getur verið um hp-samskiptavandamál að ræða og er þá reynandi að gefa skipunina

stty -ienqak

10.9 cu eða kermit bilað.

Ekki stöðva cu eða kermit afbrigðilega. Notið <return> . <return> til að sleppa út úr cu, og ^\-c eða ^-c (þ.e. control-backslash c eða hattur c).

Ef cu eða kermit eru stöðvuð afbrigðilega, læsist skjárinn (sjá 15.1 og 16.1) og tengið sem hringt var út um læsist líka. Lítið á kafla 13.2 ef tengingarnar eru í steik.

Í góðum kermit er unnt að fikta sig áfram með því að nota '?' og escape-takkann.

10.10 Forrit í steik

Ef skjárinn virðist dauður og/eða forrit gerir ekki neitt, getur verið að það sé bara að bíða eftir inntaki. Ef t.d. gleymist að skrifa skráarnafnið í skipuninni "cat skrá, þ.e. ritað er "cat", þá mun forritið cat einfaldlega sitja og bíða eftir að notanda þóknast að pikka inn það sem cat á að senda á skjáinn.

Til að stöðva innlesturinn má rita ^D (end-of-file), ellegar stöðva forritið með "interrupt", sem getur verið t.d. ^C, <delete>, <break>

Prófið líka ^Q, því vera kann að einhver hafi óvart ýtt á ^S.

11 Ýmis kerfisforrit.

11.1 Stty.

11.1.1 Inngangur - notkun.

Forritið stty sér um að stilla ýmis atriði (í "driver") varðandi samskipti tölvu og skjás.

Notkun:

stty	Gefur núverandi stillingar á þessum skjá.
stty stillingar	Setur "stillingar".
stty < /dev/ttya1	Gefur núverandi stillingar á ttya1.
stty stillingar < /dev/ttya1	Setur "stillingar" á ttya1.

11.1.2 Nokkur dæmi

stty sane	Gerir tenginguna sæmilega heilbrigða. sbr "stty raw", "stty cbreak".
stty echo	Lætur vél sýna stafi, sem eru slegnir inn. Sbr. "stty -echo".
stty echoe	Lætur vélina þurrka rétt út stafi á skjánum, þegar ýtt er á <delete> (eða <backspace>, eftir því hvað er notað).
stty erase ^?	Lætur <delete> takkann sjá um að eyða stöfum. "^" er hatt-takkinn.
stty erase ^H	Gerir <backspace> (control-H) að "erase" skipun.
stty kill ^U	Lætur control<ce eyða innsleginni línu. Í sumum Unix kerfum gerir @ -merkið þetta, en því þarf að breyta fyrir tölvupóst.
stty intr ^C	Lætur control-C vera táknið sem stöðvar forrit.

11.2 Lp kerfið

Búið er að lýsa, hvernig **lp** er notað til að prenta skrár :

lp skrá	Venjuleg prentun
lp -ol skrá	Prentun á hlið
lp -or skrá	"Hrá" prentun. Notað t.d. fyrir úttak úr troff.

Forritið **lpstat** er notað til að kanna, hvernig ástand prentarabiðraðar er:

lpstat Gefur lista yfir eigin prentverkefni.
lpstat -o Gefur lista yfir öll verkefni.
lpstat -t Gefur lista yfir öll verkefni og biðraðir.

Síðasta notendaforritið er svo **cancel**, sem stöðvar útprentun á einu verki eða fleirum.

Forritið /usr/lib/lpsched (sem enginn sér) stjórnar prentunum, sér til þess að biðraðir gangi rétt osfrv.

Kallað er á síu (filter) sem eru geymd á /usr/spool/lp/interface, til að sjá um sjálfa prentunina. Ef **lp** er notað eitt sér, er kallað á einhverja sjálfgefna síu. Ef notað er t.d. **lp -dplt** er kallað á síu sem heitir plt (þ.e. /usr/spool/lp/interface/plt).

Prentun tengist líka tæki, sem er táknað með einhverju nafni í /dev. T.d. er á "hafro" til /dev/lj, sem er laser prentarinn, /dev/plt7550, sem er teiknari osfrv.

Þegar notuð er skipunin

```
lp -dplt skrá
```

sér lpsched um að setja /usr/spool/lp/interface/plt af stað, með úttak tengt við /dev/plt7550.

Ef tæki stöðvast, ellegar er tekið úr sambandi, þarf að stoppa prentun. Það er gert með skipuninni **disable**, t.d.

```
disable lj
```

Til að setja prentun af stað aftur er notað **enable** :

```
enable lj
```

Forritið /usr/lib/lpshut er notað til að stöðva prentarakerfið (taka niður lpsched). Einstöku sinnum geta biðraðir farið alveg í klessu og getur þá þurftað 'logga sig inn' sem notandinn lp og gefa eftirfarandi skipanir :

```
disable lj
/usr/lib/lpshut
/usr/lib/lpsched
enable lj
```

(prentkerfið er tekið alveg niður og sett alveg upp aftur).

11.3 Inittab, getty, login ofl.

Þegar tölva bíður eftir að notandi 'loggi sig inn', er forritið **getty** tengt viðkomandi tæki (porti, t.d. /dev/ttya4). Það forrit sér um að senda fyrsta "login:" -merkið á skjáinn.

Þegar notandi ritar verknúmer sitt, er **login** sett af stað. Það sér um að lesa lykilorð notandans og setja af stað skel fyrir hann (ef lykilorðið reynist rétt).

Í skránni /etc/inittab eru línur af gerðinni :

```
a1:2:off:/etc/getty ttya1 H
a6:2:respawn:/etc/getty ttya6 H
```

Sú fyrri lýsir því, að ekki sé neitt getty tengt /dev/ttya1, en sú síðari segir, að getty sé tengt /dev/ttya6. Respawn merkir að nýtt getty sé sett af stað ef ekkert annað forrit (t.d. skel) er tengt viðkomandi skjá.

Stafurinn H er tilvísun í skrána /etc/gettydefs, en þar eru línur af gerðinni :

```
   9600# B9600 HUPCL PARENB CS7 # B9600 SANE PARENB CS7
ISTRIP IXANY TAB3 #login:  #9600
   H# B9600 CLOCAL # B9600 CLOCAL SANE IXANY IENQAK ECHOE
#login:  #H
   U# B1200 # B1200 SANE IXANY ISTRIP TAB3 #login:  #U
```

Hér er um að ræða hvernig stillingar eru settar fyrir línurnar. T.d. er H-línan dæmigerð stilling fyrir HP-skjá, en 9600-línan gengur fyrir vt100-skjá.

11.4 Uppsetning fyrir nýja notendur

Á sumum vélum (t.d. Professional með Venix) er til forritið **nu**, sem er keyrt til að setja inn nýja notendur.

Á öðrum vélum þarf að gera eftirfarandi í höndunum :

1. Bæta notandanum inn í /etc/passwd.
2. Búa til vinnusvæði, t.d. "mkdir /users/jon".

3. Gera notandann að eiganda að eigin vinnusvæði, t.d. :

```
chown jon /users/jon
chgrp other /users/jon
```

4. Smíða .login og .cshrc (eða bara .profile) skrá.

11.5 Uppsetning uucp tenginga

Setja þarf upp skrárnar
L.sys
L-devices
Muna að setja upp 'login' á annarri vélinni - **EKKI BÁÐUM**.
Samræma hraðann í L.sys og L-devices í vélinni, sem kallar við gettydefs í vélinni, sem er með login.
Hugsanlega þarf einnig að breyta
.nf USERFILE .fi
Ef enginn áhugi er á verndunum má setja línuna
.nf , / .fi
þar inn. (Hver sem er má gera hvað sem er).