Modul 3

Arbeta med filer

Sebastian Bengtegård, Johan Holmberg

22 Januari 2025

Innehållsförteckning

1 Förord

I denna modul kommer vi att introduceras till tre nya kommandon/program:

- `cat` (*concatenate files and print on the standard output*)
- `touch` (*change file timestamps*)
- `nano` (*Nano's ANOther editor*)

Vi kommer även lära oss att omdirigera input och output (*input/output redirection*) från ett kommando med hjälp av notationen `>`, `>>` och `|`.

Genom dessa nya kommandon kommer vi att kunna arbeta med filer, det vill säga att skapa nya filer och redigera innehållet i filer. Den nya notationen kommer bland annat hjälpa oss med att kunna kombinera kommandon, vilket kommer att öppna upp nya möjligheter för vad vi kan göra i vårt skal.

Modulen avslutas sedan med en övning.

2 Standard Output

Som vi har sett så skriver många kommandon ut sina resultat till vår skärm (i terminalen), så som `ls` och `pwd`. Vi har dock möjligheten att, med hjälp av en speciell notation, kunna bestämma vart vi vill att ett kommando ska skriva ut sitt resultat, till exempel i en fil eller till ett annat kommando.

De flesta kommandon skickar sina resultat till en plats som kallas för *standard output*, som i sin tur dirigerar detta till vår skärm. Skulle vi istället vilja omdirigera detta till en annan plats, till exempel en fil, så använder vi notationen `>`.

```
sebastian@mindator:~$ ls > lista.txt
```

I exemplet ovan, så väljer vi att omdirigera resultatet av kommandot `ls` till filen `lista.txt`. Eftersom vi väljer att skriva ut resultatet till en fil så får vi heller ingen utskrift på vår skärm.

Om vi upprepar detta kommando, så kommer filen `lista.txt` att **skrivas över** med samma resultat. Skulle vi däremot vilja **lägga till** resultatet i samma fil, för varje gång vi upprepar kommandot, använder vi istället notationen `>>`.

```
sebastian@mindator:~$ ls >> lista.txt
```

När vi lägger till ett resultat så sker det i slutet av filen, vilket samtidigt innebär att filen kommer att bli längre varje gång vi upprepar kommandot.

Om en fil **inte existerar** när vi använder notationen `>` eller `>>`, så kommer den att skapas.

3 `cat`: Skriv ut innehållet i en fil

Kommandot `cat` används för att skriva ut innehållet från en eller flera filer till *standard output*.

För att demonstrera detta kommer vi att skriva ut resultatet av kommandot `ls` i filen `lista.txt`. Sedan använder vi oss av kommandot `cat` för att skriva ut innehållet från samma fil till *standard output*.

```
sebastian@mindator:~$ ls
Desktop  Documents  dog.jpg  Downloads  example.txt
sebastian@mindator:~$ ls > lista.txt
sebastian@mindator:~$ ls
Desktop  Documents  dog.jpg  Downloads  example.txt  lista.txt
sebastian@mindator:~$ cat lista.txt
Desktop
Documents
dog.jpg
Downloads
example.txt
```

Här kan vi observera att när kommandot `ls` skriver ut sitt resultat till en fil så blir det en fil *per rad*.

Eftersom kommandot `cat` kan skriva ut innehållet från flera filer, så hade vi till exempel kunnat slå samman innehållet i filerna `example.txt` och `lista.txt` till en ny fil, till exempel `sammanslagna.txt`.

```
sebastian@mindator:~$ cat example.txt lista.txt > sammanslagna.txt
```

4 `touch`: Skapa tomma filer

Ett enkelt sätt att skapa en ny tom fil är att använda kommandot `touch` följt av ett filnamn.

```
sebastian@mindator:~$ touch ny_fil.txt
sebastian@mindator:~$ ls
Desktop  Documents  dog.jpg  Downloads  example.txt  ny_fil.txt
```

Syftet med kommandot `touch` är egentligen att uppdatera datum och tid för när en fil ändrades (*modification time*). Men, om filen vars datum och tid vi vill ändra på inte existerar, så skapas den.

5 Pipelines

För att dirigera om resultatet från ett kommando till ett annat används notationen `|`, en så kallad *pipe* (rör). I sin enklaste form kan vi skicka resultatet från ett kommando till ett annat.

```
sebastian@mindator:~$ ls -l | less
```

I exemplet ovan, så kommer utskriften av `ls -l` att skickas via en *pipe* till programmet `less`. Eftersom vi kommer att läsa listan av filer (men inte deras innehåll) med hjälp av `less`, så kan vi även till exempel söka efter ett filnamn.

Genom att kombinera flera kommandon skapar vi därför en *pipeline* (rörledning). Till exempel, säg att vi har en osorterad fil av användarnamn (`usernames.txt`), och vi hade velat läsa denna i bokstavsordning via `less`. Med hjälp av en *pipeline* är detta väldigt enkelt.

```
sebastian@mindator:~$ cat usernames.txt | sort | less
```

Kommandot `sort`, och några andra, kommer vi att presentera i "Modul 04: Sortera och filtrera".

6 Textredigeraren `nano`

Programmet `nano` används för att skapa och redigera textfiler, med andra ord är `nano` en *textredigerare*. Du har förmodligen redan använt dig av en textredigare tidigare, kanske den klassiska "Notepad" (Anteckningar) på Windows, eller en modern variant som Visual Studio Code.

Varför skulle vi vilja använda oss av en textredigerare i vår terminal när dessa finns som grafiska gränssnitt? Om vi redan befinner oss i terminalen och arbetar, så är det ett smidigt sätt att göra snabba ändringar i en fil.

Precis som med programmet `less`, så behöver vi veta hur vi kan kontrollera `nano`. Till skillnad från `less` så använder vi `nano` för att även skriva text, så vi kan till exempel inte trycka "q" för att stänga ner programmet, det hade istället skrivit ut bokstaven "q" i vår textfil.

Istället används konventionen av att trycka `CTRL` eller `ALT` tillsammans med ett tecken för att kontrollera `nano`. Till exempel `CTRL-x` för att stänga ner programmet.

Dessa brukar noteras som `^` för `CTRL` och `M` (*meta*) för `ALT`. Det vill säga att `^X` är detsamma som `CTRL-x` och `M-D` är detsamma som `ALT-d`.

På MacOS heter ALT-knappen OPTION, däremot fungerar den inte utan en viss inställning i din terminal, se nedan.

Eftersom det finns så många kortkommandon att hålla reda på visar `nano` oss ett urval av de vanligaste i botten av vår skärm. För att läsa mer om vilka kortkommandon som finns trycker vi på `^G` (`CTRL-g`). För att navigera runt i hjälptexten använder vi pil-tangenterna (upp och ner) som vanligt.

6.1 Vår första fil

För att skapa en ny fil, eller redigera en befintlig, skriver vi `nano` följt av ett filnamn.

```
sebastian@mindator:~$ nano test.txt
```

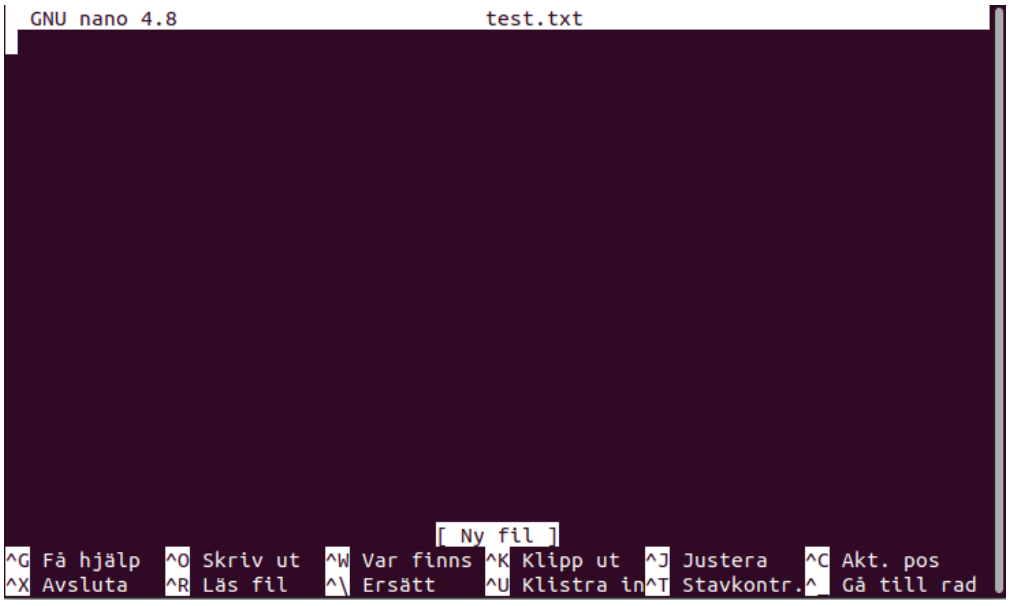

Figure 1: Textredigeraren Nano

Centrerat i överkanten kan vi se att det står “test.txt”, det vill säga den fil vi nu öppnat i `nano`, och i botten ser vi ett urval av kortkommandon.

Nu kan vi börja skriva text, till exempel “Hej från nano!”. Skulle vi sedan vilja spara våra ändringar så trycker vi på `^O` (observera `^O Write Out` i botten), det vill säga `CTRL-o`.

Därefter visar `nano` oss meddelandet "File Name to Write: test.txt", här har vi möjlighet att ändra filnamnet om vi vill, om inte så trycker vi på Enter för att spara.

För att stänga ner vår fil trycker vi sedan på `^X`.

Nu kan vi använda kommandot `cat` för att snabbt se så att vår fil finns och att den innehåller texten vi skrev, till exempel "Hej från nano!".

```
sebastian@mindator:~$ cat test.txt
Hej från nano!
```

Ett rekommenderat tillval till nano är `-m` (*enable mouse support*) som ger oss möjlighet att navigera och klicka med vår mus (vi behöver ju inte göra det svårt för oss i onödan). Det vill säga:

```
sebastian@mindator:~$ nano -m test.txt
```

Ett bra kortkommando att komma ihåg är `^C` (`CTRL-c`) som visar oss vilken rad vi befinner oss på.

6.2 nano + pipes

Till skillnad från många andra kommandon, så använder `nano` en speciell notation för att kunna ta emot resultatet av ett annat kommando, nämligen specialtecknet `-` (annars tror `nano` att den ska skapa en ny tom fil, utan ett namn).

```
sebastian@mindator:~$ ls -l | nano -
```

6.3 MacOS: Alt-Meta

För att ställa in så att OPTION-knappen i MacOS fungerar som Meta/ALT öppnar du inställningarna i terminalen och klickar i följande (se Appendix A, i slutet av detta dokument).

7 Övning

Börja med att öppna en ny terminal och navigera till den plats där du placerat övningsmappen (*exercise-module-03*).

- Omdirigera resultatet av kommandot `pwd` till filen `info.txt`. Antecka ner kommandot du skrev som **A1**. Lägg sedan till resultatet av kommandot `ls` till samma fil. Antecka ner kommandot du skrev som **A2**.

- Använd kommandot `cat` för att slå samman `a.txt` och `b.txt` till `ab.txt`. Antecka ner kommandot du skrev som **A3**. Lägg sedan till innehållet av `ab.txt` till `info.txt`.
- Använd `nano` för att redigera filen `info.txt`, skriv ut rubrikerna "A1", "A2", och "A3" på raden ovanför respektive bit text från frågorna ovan. Det vill säga "A1" ovanför den första raden (från `pwd`), "A2" ovanför resultatet av `ls` och "A3" ovanför innehållet från filen `ab.txt`. Detta är en bra övning i att navigera och redigera med nano, inget som behöver antecknas, men se till att faktiskt göra det (om din dator inte har `nano` kan du hoppa över denna punkten).
- Öppna filen `printk/sysctl.c` i `nano` och ta reda på vad som står på rad 71. Anteckna ner det som **A4**.

8 Appendix

8.1 Appendix A

Figure 2: Use Option as Meta key