Rapport

Introduktion till Linux och små nätverk, 7,5 hp
VT-2025

Projektuppgift

Laboration 2

av

Sixten Peterson (050402-XXXX)
Korrigerad: 2025-03-21

Akademin för teknik och miljö
Avdelningen för industriell utveckling, IT och samhällsbyggnad

Högskolan i Gävle
S-801 76 Gävle, Sweden

Datorpost:

education@snicon.rip

Innehållsförteckning

1. Inledning

Under laboration två används skalprogrammering för att skapa kataloger såväl som textfiler och skript utefter given uppgiftsbeskrvning vid exekvering av ett skript. Syftet är att ge en grundläggande förståelse för skalprogrammerig och hur det kan appliceras praktiskt i filstrukturen av en linuxmiljö.

1.1. Bakgrund

Uppgiften är den andra laborationen i kursen ”Introduktion till Linux och små nätverk” (DVG001) och bygger på att studenten med hjälp av skalprogrammering skapar filer och kataloger vid körning av ett *Bash*-skript. Skalprogrammeringen utförs genom en *SSH*-anslutning till Linuxmiljön som installerats i den föregående laborationen. Kraven för uppgiften är att skapa kataloger och filer, för utförlig redovisning av uppgiftskraven se Tabell 1.

Tabell 1: Redovisning av uppgiftskrav.

Relativ sökväg	**Beskrivning/Krav**
lab2.sh	Innehåll: Bash-skriptet som genererar alla övriga filer och kataloger.
filett.txt	Innehåll: ”fil ett”.
laboration2	Katalog.
laboration2/filtvaa.txt	Innehåll: ”fil två”.
laboration2/filtree.txt	Innehåll: ”fil tre”.
laboration2/katalog-1	Katalog.
laboration2/katalog-1/pgm.sh	Innheåll: Bash-skript som räknar antal rader i datafil.txt i samtliga kataloger.
laboration2/katalog-1/datafil.txt	Innehåll: Godtycklig text, fritt att hitta på.
laboration2/katalog-tva	Katalog.
laboration2/katalog-tva/datafil.txt	Innehåll: Godtycklig text, fritt att hitta på men ej samma som laboration2/katalog-1/datafil.txt.
laboration2/katalog-tva/filfyra.txt	Innehåll: ”fil fyra”.

1.2. Syfte

Laborationen syftar till att ge en grundläggande kunskap om skalprogrammering. Ytterligare syftar laborationen till att bilda en förståelse för hur det går till att skapa kataloger och filer, detta med hjälp av Bash-skript. Dessa kunskaper är en viktig del för att kunna utföra automatisering på datorer.

2. Planering och genomförande

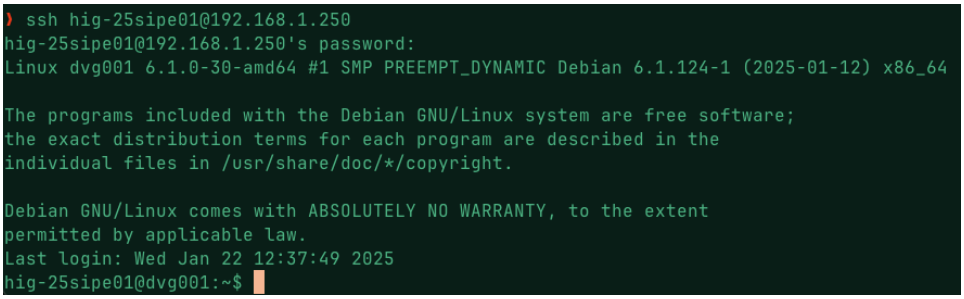
Laborationen planeras utföras med hjälp av *SSH*, *GNU Nano*, referenslitteratur och löpande dokumentation. Under genomförandet skapas även en egen katalog för laborationen för en god struktur på filerna. Så snart skript-filen är skapad och filen fått korrekta filrättigheter så påbörjas utvecklingen som varvas med test i terminalen och programmering i *Nano* tills önskat resultat är upnpått. Detta med undantag för ett kortare sidospår där ett mindre skript författas för att rensa katalogen efter testkörningar av skriptet som utvecklas för laborationen.

2.1. Planering

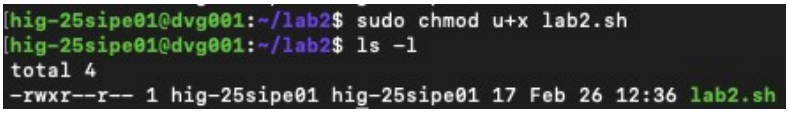
För att utföra laborationen kommer *SSH* brukas som verktyg för att kommunicera med Linuxmiljön över det lokala nätverket från en Macbook Pro (Nov, 2024) med *MacOS Sequoia 15.3.1*. Yttermera nyttjas *GNU Nano* för skalprogrammeringen då det är den textredigeraren som jag är van vid från Linuxkursen på Valla Folkhögskola. Genom hela processen testas alla kommandon först i terminalen innan de därefter skrivs in i skriptet. Vid eventuella problem används i första hand kursboken och/eller The Linux Command Line 2:ed av William Shotts som referens för skalprogrammeringen. Löpande dokumentation sker under hela laborationen för att återspegla genomförandet så ackurat som möjlig.

2.2. Genomförande

Först öppnades en anslutning till Linuxmiljön och jag loggade in, se Figur 1. Därefter skapades en ny katalog med namn ”lab2” i hemkatalogen med `mkdir lab2` för att lagra alla filer på ett strukturerat vis. Efter det brukades `cd lab2` för att navigera in till den nya katalogen och med hjälp av `touch lab2.sh` skapades en tom fil. För att göra filen exekverbar för mitt användarkonto så kördes `chmod u+x lab2.sh`, se Figur 2.

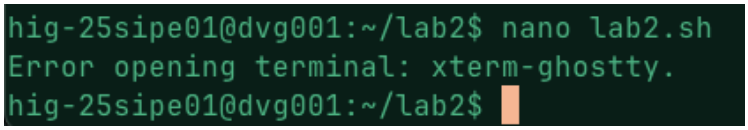


Figur 1: SSH-anslutning öppnas, och jag loggar in på datorn.

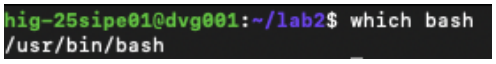


Figur 2: Filrättigheterna för lab2.sh justeras för att tillåta exekvering av användaren.

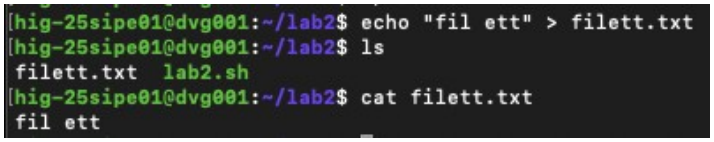
Sedan gjordes ett försök att redigera skript-filen, men terminalen som brukades (Ghostty) resulterade i ett felmeddelande, se Figur 3, vilket ledde till ett terminalbyte. Istället användes den medföljande terminalen i *MacOS Sequoia*, som kunde öppna *nano* utan problem genom `nano lab2.sh`. Väl inne i textredigeraren författas först en *shebang* som talar om vilket skal som ska användas vid körning av skriptet. Genom att köra `which bash` (se figur 4) återfanns bash-binärens filsökväg som placerades i *shebangen*. Därefter författades kommentarer som beskriver önskad funktionalitet uppdelat i flera delar. Under vardera kommentar skrevs det kod som utför det som beskrivits i kommentaren. Utanför textredigeraren testades kommandot `echo ”fil ett” > filett.txt` för att se så att det utförde det som förväntades, vilket den gjorde – se figur 5. Detta blev den första riktiga koden som placerades i skriptet. Därpå skapades filstrukturen under respektive kommentar med hjälp av `mkdir(1)`.



Figur 3: Felmeddelande vid användning av nano genom Ghostty.



Figur 4: Tar reda på var bash-binären ligger genom kommandot which.



Figur 5: Testkörning av kommandot echo, samt verifikation av resultatet.

Strax därefter utfördes den första testkörningen, utan några vidare problem – se figur 6. Då alla kataloger skapades som förväntat fortlöpte arbetet med att skapa de övriga filerna, en efter en. Vid detta laget blev det uppenbar att det kommer behövas raderas filer och kataloger varje gång skriptet ska testas. Därmed skapades ett *radera-skript* som raderar *filett.txt* och katalogen *laboration2* rekursivt, det vill säga inklusive innehållet i katalogen. Hur rekursiv radering går till finns att läsa i man, `rm(1)`. Skriptet går att hitta i Bilaga 1. Slutligen utvecklades *pgm.sh* i *lab2.sh* med hjälp av `HERE`-dokument och `cat(1)`. Centralt för *pgm.sh* är användningen av variabler i skalprogrammeringen, dessa bygger på förklaringar från William Shotts bok på sida 332-338[1], annars är det inget märkvärdigt i skriptet. Med hjälp av `man wc` gick det snabbt att hitta flaggan ”-l” för att räkna antalet rader i en fil med hjälp av `wc(1)`.

Figur 6: Första testkörningen av skriptet innan slutgiltig implementation. Katalogen laboration2 skapas som förväntat.

När själva funktionaliteten var helt klar modifierades skriptet för att visa information om vad skriptet gör för användaren med hjälp av `mkdir(1) -v`, som berättar att katalogen skapats. Ytterligare användes `echo` i kombination med `cat` för att informera användaren om att en fil skapats och vad den innehåller. Efter en testkörnings lades även tomma rader in i skriptet med hjälp av echo för att göra det lättare att läsa all information. För att se *bash-skriptet* i sin helhet se Bilaga 2, för att se hur en programkörning ser ut se bilaga 3.

3. Beskrivning av slutresultat

Resultatet är ett program som uppfyller kraven för uppgiften, ytterligare finns ett extra skript som underlättat testningen av programmet genom att radera filerna som själva huvudskriptet genererar. Innan varje testning körs extraskriptet och städar upp så att det vanliga skriptet kan göra sitt jobb i en ren miljö. Inget av skripten är på något vis perfekta eller följer några ”best practices”, däremot är skripten simplistiska och lättförstådda av erfarna såväl som oerfarna skalprogrammerare.

4. Diskussion

Om jag skulle jämföra mina skript som utförts i denna laboration med de jag genomförde under min Linuxkurs på Valla Folkhögskola så var dessa skript lite enklare i utformningen. Min upplevelse är att jag under båda kurserna mest skrapat på ytan av vad skalprogrammering kan åstadkomma, särskilt om jag jämför med skript jag stött på i andra sammanhang. Det vore kul att djupdyka lite mer i vad som går att göra, en vacker dag när jag har tid ska jag se till att läsa igenom Shotts bok från pärm till pärm för att få bättre koll på helheten.

Ska jag jämföra upplevelsen med skalprogrammering med programmering i programspråk som Java, C#, Python, JavaScript och Go så känns bash-skript väldigt kraftfulla i all sin enkelhet och framförallt smidiga. Eftersom jag kan komma åt alla kommandon som går att köra i terminalen så är det en enorm frihet för automatisering och allmän interaktion med linuxmiljön. Däremot är jag inte riktigt överens med syntaxen då den känns väldigt annorlunda från de nämnda programspråken - det är väl dock en vanesak.

I och med mina förkunskaper från kursen på Valla så har jag fått en bra repetition på grunderna i området, den största lärdomen jag tar med mig från laborationen är användningen av *man*. Tidigare har jag upplevt att *man* är kryptiskt och svåranvänt men i samband med denna uppgiften känner jag mig självsäkrare på man och har inte behövt använda internet i samma utsträckning för att förstå kommandon. Hade jag gjort om uppgiften på nytt så hade jag nog försökt ge mig på mer komplexa programmeringskoncept såsom funktioner och loopar för att underlätta utvecklandet och minska på upprepad kod. Annars känner jag mig nöjd med min insats, kraven är uppnådda och koden är välkommenterad.

5. Slutsatser

Precis enligt syftet har har ett skript skapats för att generera filer och kataloger. Under processen har variabler och diverse kommandon använts, skriptet är högst simplistiskt men gör det den ska. Yttermera kommuniceras det skriptet gör till användaren i terminalen. Utöver den grundläggande skalprogrammeringen har kommandot *chmod* använts för att justera filrättigheter. Trots problem med att öppna *nano* i *Ghostty* har uppgiften för övrigt flutit på utan besvär. Slutligen är bilagorna bra exempel på grundläggande skalprogrammering i *Bash*.

6. Referenser

Litteraturförteckning

[1]: W. Shotts, The Linux Command Line, A complete introduction, 2019

I. Bilaga 1: Radera.sh

```
#! /usr/bin/bash

# Radera filer för att kunna testa i en "ren" miljö
rm -r laboration2
rm filett.txt
```

II. Bilaga 2: lab2.sh

```
#! /usr/bin/bash

echo "Kör lab2.sh av Sixten Peterson (25sipe01)..."
echo ""

# Skriver "fil ett" till filett.txt
echo "fil ett" > filett.txt
echo "filett.txt skapad, se innehåll nedan:"
cat filett.txt
echo ""

# Skapar katalogen laboration2
mkdir -v laboration2
echo ""

# Skapar filerna som ska vara i laboration2
echo "fil två" > laboration2/filtvaa.txt
echo "filtvaa.txt skapad, se innehåll nedan:"
cat laboration2/filtvaa.txt
echo ""

echo "fil tre" > laboration2/filtree.txt
echo "filtree.txt skapad, se innehåll nedan:"
cat laboration2/filtree.txt
echo ""

# Skapar katlog-1
mkdir -v laboration2/katalog-1
echo ""

# Skapar filer i katalog-1
cat > laboration2/katalog-1/datafil.txt <<EOF
Kod: A12B3
Kategori: Elektronik
Pris: 599 kr
Lagerstatus: I lager
EOF
echo "datafil.txt skapad, se innehåll nedan:"
cat laboration2/katalog-1/datafil.txt
echo ""

# Skapar katalog-tva
mkdir -v laboration2/katalog-tva
echo ""

# Skapar filer i katalog-tva
cat > laboration2/katalog-tva/datafil.txt <<EOF
Sverige, Svenska
Brasil, Portugês
```

```
France, Français
Italia, Italiano
EOF
echo "datafil.txt skapad, se innehåll nedan:"
cat laboration2/katalog-tva/datafil.txt
echo ""

echo "fil fyra" > laboration2/katalog-tva/filfyra.txt
echo "filfyra.txt skapa, se innehåll nedan:"
cat laboration2/katalog-tva/filfyra.txt
echo ""

# Skapar pgm.sh
cat > laboration2/katalog-1/pgm.sh <<'EOF'
#!/usr/bin/bash

# Lagrar antalet rader i variabler
rader_datafil1=$(cat datafil.txt | wc -l)
rader_datafil2=$(cat ../katalog-tva/datafil.txt | wc -l)

# Skriver ut antalet rader i standard output
echo "datafil.txt i katalog-1 har $rader_datafil1 rader."
echo "datafil.txt i katalog-tva har $rader_datafil2 rader."
EOF
echo "pgm.sh skapad, se innehåll nedan:"
cat laboration2/katalog-1/pgm.sh
echo ""

# Justerar filrättigheter
chmod +x laboration2/katalog-1/pgm.sh
echo "Filrättigheter ändrade för pgm.sh, programmet går nu att
exekveras fritt (+x)."
```

III. Bilaga 3: Programkörning lab2.sh

```
hig-25sipe01@dvg001:~/lab2$ ls
filett.txt  lab2.sh  laboration2  radera.sh
hig-25sipe01@dvg001:~/lab2$ ./radera.sh
hig-25sipe01@dvg001:~/lab2$ ls
lab2.sh  radera.sh
hig-25sipe01@dvg001:~/lab2$ ./lab2.sh
Kör lab2.sh av Sixten Peterson (25sipe01)...

filett.txt skapad, se innehåll nedan:
fil ett

mkdir: created directory 'laboration2'

filtvaa.txt skapad, se innehåll nedan:
fil två

filtree.txt skapad, se innehåll nedan:
fil tre

mkdir: created directory 'laboration2/katalog-1'

datafil.txt skapad, se innehåll nedan:
Kod: A12B3
Kategori: Elektronik
Pris: 599 kr
Lagerstatus: I lager

mkdir: created directory 'laboration2/katalog-tva'

datafil.txt skapad, se innehåll nedan:
Sverige, Svenska
Brasil, Portugês
France, Français
Italia, Italiano

filfyra.txt skapa, se innehåll nedan:
fil fyra

pgm.sh skapad, se innehåll nedan:
#!/usr/bin/bash

# Lagrar antalet rader i variabler
rader_datafil1=$(cat datafil.txt | wc -l)
rader_datafil2=$(cat ../katalog-tva/datafil.txt | wc -l)

# Skriver ut antalet rader i standard output
echo "datafil.txt i katalog-1 har $rader_datafil1 rader."
echo "datafil.txt i katalog-tva har $rader_datafil2 rader."

Filrättigheter ändrade för pgm.sh, programmet går nu att exekveras fritt (+x).
hig-25sipe01@dvg001:~/lab2$
```