Oblig 1, INF3331

Aina Johannessen

September 19, 2014

Abstract

Oppsummerer arbeidet jeg har gjort i oblig1 i inf3331. Hovedsaklig dreide oppgavene seg om filhåndtering. Obligen er delt i to deler. Første del er i Bash, andre del i Python

1 Del 1: BASH

1.1 List opp alle filer som er modifisert de siste n dagene.

Løsningen er i filen: "list_new_files.sh". Jeg brukte koden:

```
find "$1" -type f -mtime -$2 \! -name ".*" | xargs ls -sSr1k
```

`"$1"` Path settes som første parameter til programmet, og sendes vidre til find, som søker rekursivt under denne
`\! -name ".*"` ekskluderer alle filer som begynner på "." (dot). Dette er nødvendig da Mac OS X lager skjulte filer jeg ikke vil ha med i resultatet
`ls -sSr1k`, skriver ut filene funnet i "find" kommandoen. En rad av gangen(r1), sortert etter størrelse(S), skriver ut størrelsen(s) i kilobyte(k)
`-type f`, begrenser resultatet til kun filer
`-mtime -$2` Filer som er modifisert for `$2` dager siden. Antall dager settes som den andre input argumentet når programmet kjøres
`xargs` lagrer alle filene `find` finner i en streng, splitter det opp og sender det som input argument til ls. Dette lar meg få lov til å "pipe" i mellom `find` og `ls`. Uten xargs ville ikke dette gått

```
-------test result 1:------
$ ./list_new_files.sh file_tree 100
 36 file_tree/Pkvye/htZiVgRE
 48 file_tree/Kq0Wv/5RYWI5kQ
```

```
104 file_tree/KqOWv/MH/7GvTL2y
184 file_tree/zg/grYxji7
564 file_tree/zg/Hu/dNmOlK
644 file_tree/KqOWv/MH/XhdhBbk
```

1.2 Find alle filer som inneholder et gitt ord.

Løsningen er i filen: "find_word.sh". Jeg brukte koden:

```
$(grep -r "$2" $1/*)
```

Grep søker på innholdet i filene. Den leter rekursivt(-r) i alle under mappene i den oppgitte "pathen" (`$1`). "Pathen"(`$1`) settes som det første input argumentet når programmet kjøres. Ordet vi søker etter (`$2`) settes som det andre input argumentet når programmet kjøres.

```
-----test result 1-------
$ ./find_word.sh file_tree what
file_tree/Pkvye/vlfN/ZLbGhCmj:NDa6gmZswhat77iTUFuoNiG23Yn

----test result 2------
$ ./find_word.sh file_tree help
No files containing "help" found.
```

1.3 Slett alle filer med størrelse større enn en gitt verdi

Løsningen er i filen: "sized_delete.sh". Brukte koden:

```
deleted_files=‘find "$1" -type f \! -name ".*" -size +$2k -print -delete‘
```

Her brukte jeg såkalte "back-ticks" for å lagre alle resultatet fra find i variabelen `deleted_files`. Tilsvarende som `deleted_files=$(find...)`
`-size +$2k` Finner filer med størrelse `$2` eller over(+), oppgitt i kilobyte(`k`) Størrelsen(`$2`) settes som andre input argumentet når programmet kjøres i kilobyte
`-print` printer ut linje for linje
`-delete` sletter alle filene som er funnet

```
----test run 1-------
$ ./sized_delete.sh file_tree 750
```

```
Deleting...
file_tree/Kq0Wv/MH/Z9kP8NB
file_tree/Kq0Wv/MH/zWG/8puxfjS
file_tree/Pkvye/vlfN/ZLbGhCmj
file_tree/zg/Hu/vv/2KKnyIt5

-----test run 2-------
$ ./sized_delete.sh file_tree 750
No file of size 750 kilobytes or larger found.
```

1.4 Sorter linjene i en fil og lagre dem i en ny fil.

Løsningen er i filen: "sort_file.sh". Koden jeg brukte var:

```
sort $1 > $2
```

`sort` sorterer innholdet i en fil alfabetisk, og skriver det ut
`>` skriver output fra venstre til en fil på høyre
Koden sorterer altså det som er i filen `$1` og sender det til filen `"$2"`

```
------test run 1------
$ ./sort_file.sh unsorted_fruits sorted_fruits
$ cat sorted_fruits
apple
grape
orange
pear
pineapple
```

2 Del 2: Python

Jeg bruker oppgavemalen, som definerer tre hovedfunksjoner med hver sin oppgave.

`random_string` skal generere en tilfeldig streng med en fastsatt lengde.
`generate_tree` skal generere et tilfeldig mappe struktur
`populate_tree` skal generere tilfeldige filer i en struktur, med tilfeldig størrlese, aksesert tid og modifisert tid.
Løsningen er i filen: "my_generate_file_tree.py".

2.1 Hvordan jeg har løst oppgaven

For å enklere holde styr på den genererte strukturen bygges den under en hovedmappe, `file_tree_aina`. Jeg bruker `shutil.rmtree()` for å slette denne mappa for hver gang programmet kjører.

Implementasjon av random_string: Bruker `radom.choice(legal_char)` for å hente et tilfeldig tegn fra strengen `legal_char`. Disse tegnene, blir satt sammen til en egen streng i en for-løkke. Input argumentet `length` bestemmer hvor mange ganger løkka går, og for hver iterasjon blir et nytt tegn lagt til i den tilfeldige strengen.

Implementasjon av generate_tree: Jeg endret litt på input parameterne til `generate_tree` slik at den inkluderer navn på folderen jeg vil lage. Dette skaper et mer klart skille mellom "target" (der vi skal lage nye foldere) og navnet på selve folderen som skal genereres.

I funksjonen `generate_tree` har jeg laget to andre funksjoner; `folder_output` og `generate_tree_recursion`.

`folder_output` brukes kun til å skrive ut en pen output til terminalen når `verbose` er satt, `generate_tree_recursion` Rekursiv funksjon som genrerer mappestrukturen

I `generate_tree_recursion` bruker jeg:

```
random.choice( range( 1, rec_depth + 1 ) ) > depth_level
```

Denne styrer dybden i mappestrukturen. I koden er `rec_depth` maksimal tillatt dybde og `depth_level` dybden vi har nå. Koden genererer en tilfeldig ønsket dybde mindre en eller lik `rec_depth`, og dersom denne er mindre eller lik dybden vi har nå, avbrytes rekursjonen, og vi lager ikke flere mapper under mappa vi er på.

Implementasjon av populate_tree: Jeg bruker her to lokale funksjoner, `file_output` og `walk_function`. `file_output` brukes kun til å skrive ut en pen output til terminalen når `verbose` er satt, `walk_function` er callback funksjonen som sendes til `os.path.walk`

`os.path.walk` kjører `walk_function` for hver folder i treet og generer et tilfeldig antall filer for hver gang. Størrelse, aksessering tid(atime) og modifikasjons tid(mtime) blir tilfeldig valgt. Jeg har lagt til linjeskift som et lovlig tegn når jeg skriver til filen. Ved bruk av os.utime(); endrer jeg aksessert og modifisert tid.
Jeg hadde litt utfordringer med atime på filene, da det virker som om OS X automatisk satte den når skriptet avsluttet. Filene har rett atime, helt frem til skriptet avsluttes. Jeg mener skriptet er riktig.

2.2 Resultat

```
 ----------test result 1--------
$ ./my_generate_file_tree.py "./" 3 2 300 4 1388534400 1406851200 4 1

MAKING DIRECTORIE TREE::
------------------------
- file_tree_aina
   - yjez5Y
      - Hrkgn6
         - YmtNU1
            - MQFlDf
         - 2Is5K3
            - ALBktZ
            - cNrHDv
         - mAmNrw

CREATING FILES:
----------------
- "J4gdoW": size 100[kB], atime: 11 May 2014 13:19, mtime: 20 Feb 2014 08:58,
in folder: ./file_tree_aina
- "aNwfef": size  21[kB], atime: 18 Feb 2014 13:42, mtime: 06 Jun 2014 23:34,
in folder: ./file_tree_aina/yjez5Y
- "u2xruL": size 253[kB], atime: 25 May 2014 05:42, mtime: 29 May 2014 01:21,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6
- "PcV3jr": size 193[kB], atime: 05 Mar 2014 12:42, mtime: 15 Jan 2014 20:26,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/2Is5K3
- "StUQ9_": size 263[kB], atime: 27 May 2014 19:52, mtime: 09 Feb 2014 06:54,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/2Is5K3/ALBktZ
- "zynN9J": size 136[kB], atime: 12 Jun 2014 01:49, mtime: 30 Jun 2014 07:14,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/2Is5K3/ALBktZ
- "zGh8Pv": size  78[kB], atime: 16 Mar 2014 04:41, mtime: 31 Jul 2014 18:56,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/2Is5K3/cNrHDv
- "2NKrqn": size 186[kB], atime: 28 Jul 2014 22:38, mtime: 05 Jun 2014 16:37,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/mAmNrw
- "y7QXMa": size 189[kB], atime: 12 Mar 2014 00:11, mtime: 17 Apr 2014 16:28,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/mAmNrw
- "6y63ax": size 293[kB], atime: 05 Jan 2014 13:02, mtime: 18 Feb 2014 02:45,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/YmtNU1
- "RhTSA1": size 271[kB], atime: 16 May 2014 02:33, mtime: 24 Jan 2014 21:50,
in folder: ./file_tree_aina/yjez5Y/Hrkgn6/YmtNU1/MQFlDf

----------Displaying the folder tree------------
Ainas-MacBook-Pro:Finished assignment aina$ ls -R
```

```
file_tree_aina my_generate_file_tree.py

./file_tree_aina:
J4gdoW yjez5Y

./file_tree_aina/yjez5Y:
Hrkgn6 aNwfef

./file_tree_aina/yjez5Y/Hrkgn6:
2Is5K3 YmtNU1 mAmNrw u2xruL

./file_tree_aina/yjez5Y/Hrkgn6/2Is5K3:
ALBktZ PcV3jr cNrHDv

./file_tree_aina/yjez5Y/Hrkgn6/2Is5K3/ALBktZ:
StUQ9_ zynN9J

./file_tree_aina/yjez5Y/Hrkgn6/2Is5K3/cNrHDv:
zGh8Pv

./file_tree_aina/yjez5Y/Hrkgn6/YmtNU1:
6y63ax MQFlDf

./file_tree_aina/yjez5Y/Hrkgn6/YmtNU1/MQFlDf:
RhTSA1

./file_tree_aina/yjez5Y/Hrkgn6/mAmNrw:
2NKrqn y7QXMa
```