Obligatorisk innlevering 1 høsten 2014, INF3331

Arselan Sultani <arselans@ulrik.uio.no>

September 19, 2014

1 Oppgave 1.1

Oppgaven løst ved bruk av følgende komandoer:
find: for å finne de filene vi leter etter
-type: type fil vi leter etter
-mtime: alder på filen vi leter etter
xargs: for å vise filresultatene
sort: Sort resultatet

1.1 Kjøring

```
./list_new_files.sh file_tree/ 100
34      file_tree/Pkvye/htZiVgRE
45      file_tree/Kq0Wv/5RYWI5kQ
184     file_tree/zg/grYxji7
564     file_tree/zg/Hu/dNmOlK
644     file_tree/Kq0Wv/MH/XhdhBbk
```

Alle file i mappen blir skrevet ut dersom ingen av filene passer kriteriene som er gitt av brukeren.

2 Oppgave 1.2

Oppgaven løst ved bruk av følgende komandoer:
find: for å finne de filene vi leter etter
grep: er ordet som vi får som vi skal lete etter

Kjøring

```
./find_word.sh file_tree/ what
file_tree/Pkvye/vlfN/ZLbGhCmj:NDa6gmZswhat77iTUFuoNiG23Y

./find_word.sh file_tree/ hello
```

Når den ikke finner noe, blir ingenting skrevet ut.

Oppgave 1.3

Oppgaven løst ved bruk av følgende komandoer:
find: for å finne de filene vi leter etter
-size: størrelsen som er større enn gitt
-exec rm -rf \; Fjerner alle filer som oppfyller kriteriene. Dersom vi hadde hatt -ok i stedet for -exec hadde den spurt oss for hver fil om vi vil slette eller ikke. Mens -exec bare utfører.

Kjøring

```
./sized_delete.sh file_tree/ 750
file_tree/Kq0Wv/MH/Z9kP8NB
file_tree/Kq0Wv/MH/zWG/8puxfjS
file_tree/Pkvye/vlfN/ZLbGhCmj
file_tree/zg/Hu/vv/2KKnyIt5
```

Oppgave 1.4

Oppgaven løst ved bruk av følgende komandoer:
sort: Sortere filen som man får i første argumentet
-o: Skrive til fil som man får som andre argument

Kjøring

```
./sort_file.sh unsorted_fruits sorted_fruits

cat sorted_fruits
apple
grape
orange
pear
pineapple
```

3 Oppgave 2

Den starter ved a sjekke argumenter, og hvis det er mindre enn 4, så skriver den ut en feilmelding. Hvis ikke, så skriver den ut først target, så files osv. Så kaller vi på metoden generate_tree.

3.1 generate_tree() og mkDirs()

Jeg har antatt at vi skulle finne random dypde fra root. Og det skal være mellom 2 og det er som gitt. Så skal den finne random tall mellom 2 til dir fordi jeg har antall at også dette skal være random. Så kaller den på mkDirs metoden som lager mappene. Så sier at så lenge dypde er større eller lik 0, så skal den gå videre. Så lenge x er mellom 1 og en random tall for antall mapper i mappen, lager jeg en ny path i new_dir2. Den tar den pathen som den allerede har og legger en \i mellom og så en tilfeldig string fra random_string() som blir navnet på den nye mappen. os.makedirs lager mappen. Deretter blir det sjekket om verbose er True eller False, for dersom den er True, så skal den skrive pathen til den nye mappen. Så kalles populate_tree(), som fyller mappene med filer. Og den vil gå i rekursjon. Den vil eventuelt bli stoppet av for-løkka og dir_deep som minsker hver gang vi lager en mappe inn i en mappe fordi da er vi lenger unna root.

3.2 populate_tree()

Den vil først lage en last_modified og last_accessed. last_modified skal være mellom start_time og end_time, som vi får fra argumentene og last_accessed skal være mellom last_modified og end_time, fordi jeg antar at du kan få tilgang til filen uten å ha endret på den. Deretter lages det filer i mappen. Filnavnet blir opprettet og vi spør om vi kan åpne filen. Og dersom filen eksisterer, så kan skal vi åpne den og overskrive. Men dersom filen ikke eksisterer, så oppretter vi den. Dersom verbose er True, så skriver vi ut pathen til filen som vi nettopp lagde. Deretter finner vi størrelsen til filen og fyller den etter til det blir så mye som den vil ha Og til slutt closer vi den.