5 C Shell (csh)

5.1 Ominaisuuksia ja alustustiedostot

C shellin ominaisuuksin kuuluvat mm: komentorivihistoria, aliakset, aritmetiikka, tiedostonimitäydennys ja töidenhallinta.

Mikäli csh käynnistetään login shellinä suoritetaan aluksi systeemin laajuinen alustustiedosto `/etc/csh.login` (mikäli tiedosto on olemassa). Seuraavaksi etsitään käyttäjän kotihakemistosta tiedostoa nimeltä `.cshrc` ja suoritetaan siellä olevat komennot. Viimeisenä suoritetaan käyttäjän kotihakemistossa oleva `.login`, jos sellainen löytyy.

Kotihakemistossa oleva `.cshrc` tiedosto suoritetaan myös kaikille myöhemmin käynnistettäville C shelleille.

5.2 Muuttujat

5.2.1 Lokaalitmuuttujat (shell muuttujat)

Lokaalit muuttujat ovat voimassa (tunnettuja) vain siinä shellissä, jossa ne ovat määritelty. Normaali tapa on nimetä lokaalit muuttujat käyttäen pieniä kirjaimia. Lokaaleja muuttujia voidaan asettaa tai poistaa käyttäen `set` ja `unset` komentoja.

C shellissä on käytössä joukko ns. shell muuttujia, joilla voidaan määritellä shellin käyttäytymistä. Osa näistä muuttujista on "on/off" (boolean) tyyppisiä ja osa vaatii määrittelyn asetettaessa. Yleensä nämä muuttujat määritellään `.cshrc` tiedostossa, jolloin ne saadaan käyttöön myös alishelleissä.

Näitä muuttujia ovat mm.

argv
Taulukko komennon argumenteista (ks. luku 5.4).

cwd
Absoluuttinen polku nykyiseen työhakemistoon. Muuttuja päivitetään automaattisesti aina kun hakemistoa vaihdetaan `cd` komennolla.

```
echo $cwd
```

history
Komennolla otetaan käyttöön komentohistoria ja asetetaan sille koko. Mikäli muuttujaa ei ole asetettu komentohistoriaa ei ylläpidetä eivätkä komentohistoriakorvaukset ole mahdollisia.

home
Absoluuttinen polku käyttäjän kotihakemistoon.

ignoreeof
Muuttuja, joka määrittelee sallitaanko shellistä poistuminen `ctrl-d` komennolla.

```
set ignoreeof
```

noclobber
Asettamalla *noclobber* muuttuja estetään jo olemassa olevan tiedoston päällekirjoitus esimerkiksi `cat > file` komennolla.

```
set noclobber
```

noglob
Kun *noglob* muuttuja on asetettu tiedostonimikorvausta ei suoritetaan, vaan tiedostonimikorvauksen metamerkit tulkitaan sellaisenaan.

```
set noglob
echo * ?? [] ~
```

notify
Jos muuttuja on asetettu ilmoitetaan taustalla suoritettujen töiden päättymisestä välittömästi stdout:iin.

path
Määrittelee ne hakemistot, joista käyttäjän antamia komentoja etsitään.

prompt
Määrittelee komentokehotteen. Oletuskehote on % merkki.

shell
Sisältää `csh` tiedostonimen polkuineen.

status
Viimeksi suoritetun komennon palutusarvo. Mikäli komennon suoritus on päättynyt epänormaalisti lisätään 0200 status muuttujan arvoon. Sisäiset komennot, jotka päättyvät epänormaalisti palauttavat arvon 1 ja kaikki muut sisäiset komennot asettavat arvoksi 0.

```
echo $status
```

5.2.2 Ympäristömuuttujat

Ympäristömuuttujat ovat globaaleja muuttujia ja ne ovat voimassa myös kaikissa (ali)prosesseissa, jotka käynnistetään shellistä, jossa ympäristömuuttuja on määritelty. Ympäristömuuttujia voidaan asettaa `setenv` komennolla ja niitä voidaan poistaa komennolla `unsetenv`. Ympäristömuuttujat voidaan listata käyttäen komentoja `setenv` tai riippuen järjestelmästä myös komennoilla `printenv` (UCB) tai `env` (SVR4).

5.2.3 Muita erityisasemassa olevia muuttujia

$?muut
Palauttaa 1 jos muuttuja *muut* on määritelty. Jos muuttujaa *muut* ei ole määritelty palautetaan 0.

```
echo $?ignoreeof
set ignoreeof
echo $?ignoreeof
```

$#taulu
Tulostaa taulukon *taulu* alkioiden lukumäärän.

```
echo $#argv
```

$$
Tulostaa nykyisen shellin PID:n.

$<
Käytetään luettaessa syötettä käyttäjältä.

```
set nimi=$<      echo $nimi
```

5.3 Syötteen ja tulostuksen uudelleensuuntaus

Kuten Bourne ja Korn shelleissä myös C shellissä on käytössä syötteen ja tulostuksen uudelleensuuntaukseen <, <<, >, >> merkit. Lisäksi csh:ssa on seuraavat tavat syötteen tai tulostuksen uudelleensuuntaukseen.

- **> &** Ohjaa sekä tulostuksen että virheen annettuun tiedostoon.
  `cc prog.c -o prog >& outfile`
- **>>&** Ohjaa sekä tulostuksen että virheen annetun tiedoston loppuun.
  `find . -name file >>& foundit`
- **|&** Ohjaa edellä olevan komennon tulostuksen ja virheen seuraavan komennon syötteeksi.
  `ls -la | sort | more`
  `ls -la |& sort | more`
- **>!** Jos `noclobber` muuttuja on asetettu kumotaan sen vaikutus tämän komennon suorituksessa. Annettu tiedosto joko luodaan tai sen sisältö korvataan.
  `ls -la >! file`
- **>>!** Kumotaan `noclobber` muuttujan vaikutus komennon suorituksessa. Annettu tiedosto joko luodaan tai komennon tulostus ohjataan olemassaolevan tiedoston jatkeeksi.
  `cat file >>! file2`

\>>&! Kumotaan `noclobber` muuttujan vaikutus ja ohjataan komennon tulostus ja virhe annetun tiedoston jatkeeksi.
`find . name file >>&! foundit`

5.4 Paikkasidonnaiset parametrit ja argv taulukko

Komentojonoille välitettyihin parametreihin voidaan viitata samoin kuten Bourne ja Korn shelleissä käyttäen merkintää $1 $2 ... ${10} jne. $0 tarkoittaa komentojonon nimeä. Toinen tapa on käyttää hyväksi shellin sisäistä taulukkoa `argv`.

$argv[1]	Tarkoittaa ensimmäistä argumenttia.
	`set argv=(a s d gf kh)`
	`echo $1`
	`echo $argv[1]`

$argv[n]	Tarkoittaa argumenttia *n*. `echo $argv[3]`
$argv[*]	Tarkoittaa kaikkia argumentteja. `echo $argv[*]`
$argv	Tarkoittaa kaikkia argumentteja. `echo $argv`
$#argv	Tarkoittaa argumenttien määrää. `echo $#argv`
$argv[$#argv]	Tarkoittaa viimeistä argumenttia. `$argv[$#argv]`

5.5 Taulukot

Kuten Korn shellissä myös C shellissä voidaan määritellä yksiulotteisia taulukoita. Taulukon alkioiden indeksointi alkaa ykkösestä. Taulukko voidaan asettaa `set` komennolla tai komentokorvauksella.

```
set viikko=( ma ti ke to pe la su )
```
Asettaa *viikko* nimisen taulukon alkioiksi *ma ti ke to pe la su*.

```
echo $viikko[*]
```
Tulostaa taulukon *viikko* kaikki alkiot.

```
echo $viikko[4]
echo $#viikko
echo $viikko[$#viikko]
set viikko[6]=lauantai
```

Taulukon alustaminen käyttäen komentokorvausta.

```
set paivays=`date`
echo $paivays[*]
echo $paivays[5]
```

5.6 C shellin sisäisiä komentoja

Seuraavassa on lueteltu muutamia C shellin sisäisiä komentoja.

logout
Kirjoittautuminen ulos shellistä.

nohup [komento]
Suoritetaan komennot välittämättä HUP signaalista (hangup). Ilman argumentteja koskee loppuja komentojonon komentoja.

notify [%job]
Shell ilmoittaa käyttäjälle asykronisesti mikäli nykyisen tai parametrina annetun työn tila muuttuu. Ilmoitus on automaattinen jos shell muuttuja `notify` on asetettu.

onintr [-] [label]
Kontrolloi shellin käyttäytymistä keskeytyksissä. Ilman argumenttejä palautetaan oletukset. Jos argumentti on viiva [-] kaikki keskeytykset jätetään huomiotta. Mikäli argumenttina on *label* suoritetaan "hyppy" *label*in määräämään paikkaan, jos keskeytys otetaan vastaan tai jos lapsiprosessi päättyy koska se on keskeytetty.

repeat n komento
Toisetaan komentoa *komento* *n* kertaa.

```
repeat 10 echo kala
```

set [muut[=arvo]]
Ilman argumentteja `set` komento tulostaa kaikki nykyisen shellin lokaalit muuttujat. Komento `set` *muut arvo* asettaa muuttujan *muut* arvoksi *arvo*.

setenv [MUUTTUJA [sana]]
Ympäristömuuttujien asettaminen.

source [-h] file
Luetaan komennot tiedostosta *file*. Käytetään yleensä `.login` ja `.cshrc` tiedostojen uudelleensuoritukseen (vrt. sh:n ja ksh . komentoon). Optio `-h` aiheuttaa sen, että komennot tiedostosta *file* laitetaan komentohistoriaan suorittamatta komentoja.

```
source .login
```

5.7 Ehto- ja silmukka komennot

5.7.1 If komento

if (testi1 **) then**

komento1

else if (testi2 **) then**

komento2

else

komento3

endif

Mikäli *testi1* palauttaa toden suoritetaan *komento1*. Muuten suoritetaan *testi2* jne. Jos mikään testi ei ole tosi suoritetaan `else` osan jälkeinen komento. `else if` osia voi olla useampia tai ne voivat puuttua kokonaan, myös `else` osa voi puuttua. Suoritettavia komentoja voi olla myös useampia.

5.7.2 Switch komento

switch (muuttuja)

case vakio1:

komento1

breaksw

case vakio2:

komento2

breaksw

default:

komento3

breaksw

endsw

`if` komennon sijasta voidaan käyttää `switch` komentoa. Parametrin *muuttuja* arvoa verrataan `case` komennon perässä olevaan *vakio*n arvoon ja jos vastaavuus löytyy suoritetaan perässä olevat komennot kunnes tulee vastaa `breaksw`. *vakio* voi sisältää myös "jokeri"merkkejä. Mikäli jokin `breaksw` puuttuu verrataan *muuttuja*n arvoa seuraaviin *vakio*n arvoihin kunnes vastaan tulee joko `breaksw` tai `endsw`. `default:` kohta suoritetaan mikäli *muutt-*

ja ei vastaa mitään *vakio*ta. `default:` kohta voi myös puuttua.

5.7.3 Foreach silmukka

foreach muuttuja (sanalista)

  komentoja

end

`Foreach` komennon perässä olevan muuttujan (*muuttuja*) arvoksi laitetaan *sanalistan* sana ja sen jälkeen suoritetaan ennen `end` osaa olevat komennot. Silmukkaa suoritetaan niin kauan kun *sanalista*ssa on sanoja. Silmukan suoritus voidaan lopettaa `break` komennolla ja `continue` komennolla voidaan siirtyä seuraavalle kierrokselle.

5.7.4 While silmukka

while (testi)

  komentoja

end

Silmukkaa suoritetaan niin kauan kun *testi* on tosi (eri kuin nolla) Silmukan suoritus voidaan lopettaa `break` komennolla ja `continue` komennolla voidaan siirtyä seuraavalle kierrokselle.

5.7.5 Goto

C shellissä on myös käytössä `goto` komento, jolla voidaan suorittaa "hyppyjä" komentojonossa.

:label

  komentoja

  goto label

Seuraavassa on lueteltu muutamia eroja shellien syntaksissa.

C shell	Bourne shell	Korn Shell
Lokaalien muuttujien alustaminen:		
`set x=5`	`x=5`	`x=5`
Ymp. muuttujien alustaminen:		
`setenv NAME Unix`	`NAME=Unix` `export NAME`	`export NAME=Unix`
Palautusarvo:		
`$status`	`$?`	`$?`
Taulukot:		
`set x = ( a b c )`	`N/A`	`y[0]=a; y[1]=b; y[2]=c;` `set -A y a b c`
`echo $x[0] $x[1]`	`N/A`	`print ${y[0]} ${y[1]}`
`echo $x[*] tai $x[*]`	`N/A`	`print ${y[*]}`
`echo $#x`	`N/A`	`echo ${#y[*]}`
Aritmeettiset operaatiot		
`@ var = 1 + 2`	`var='expr 1 + 2'`	`(( var=1 + 2 ))` `let var=1+2`
Alustustiedosto login shellille		
`.login`	`.profile`	`.profile`
Alustustiedosto muille shelleille		
`.cshrc`	`N/A`	`ENV=file`