$ cx hello teen F täidetavaks
$ hello proovin uuesti
Tere, tere! OK!
$ rm hello pole enam vaja
aga kui oleks vaja mitu argumenti?
chmod +x $1 $2 $3 ... $9 hästi ei sobi
nn. positsioonilised parameetrid
NB! mida tähendab $10
S-s on veel 2 erimuutujat: $0 ja $# (0-parameeter - P nimi,
parameetrite arv käsus, pos. par. arv)
chmod +x $* $* tähendab interpretaatorile 'kõik arg-d'

Tingimuslikud operaatorid

primitiivid - S-i sisseehitatud operatsioonid (U-s utiliidid)
iga P-i täitmine lõpeb mingi koodi väljatöötamisega (0 või 1)
konveieri lõpukood - tema viimase P-i lõpukood
1) && 0 (true)
$ test -d bin && echo edukas!
edukas!
2) || 1 (false)
$ test -d mat || echo ei kolba!
ei kolba!
3) list
konveier - lihtkäsk või lihtkäskude jada (| või ^)
list - konveierite jada
eraldajaks võib olla
• ; või reavahetus
•&
•&&
• ||
list on U põhistruktuur

Sümboli spetsiaalse tähenduse muutmine 06.03.97 2.

probleem - enamikku SS kasutavad teised P-d

$ ls | wc -l | - konveier

$ expr $m1 \| $m2 | - disjunktsioon

3 meetodit:

1. sümboli ette \
2. kõikide sümbolite spetsiaalne tähendus on muudetud, kui nad on jadas, mis on apostroofide vahel (‘jada‘)
3. kõikide sümbolite spetsiaalne tähendus, v.a. \, `, “, $, on muudetud, kui nad on jadas, mis on jutumärkide vahel (“jada“)

1 sümboli spets. tähenduse muutmiseks - soovitav \

sümbolite grupi korral - soovitav (‘jada‘), kui ei ole vaja nendest ühtegi interpreteerida

3. kasutada siis, kui soovitakse kasutada käske ja asendusi ($,..)

```
$ echo '$HOME'
$ echo \$HOME
$HOME
$  echo "$HOME"
/export/home/user6/aavo
```

Käsu tulemuse asendus

```
$ echo tana on `date`
tana on Wed Mar 5 18:09:17 EET 1997
$ echo "tana on `date`."
tana on Wed Mar  5 18:14:41 EET 1997.
$ set m=`date`
$ echo $m
Wed Mar 5 18:16:45 EET 1997
$ expr 5 + 13                18
m=10
m=`expr $m + 1`
echo $m                      11
```

if

if if-list
then then-list
elif elif-list
then then-list
else else-list
fi

elif-de arv ei ole piiratud
list'id viitavad U-i käskude listidele
elif …then ja ***else*** võivad puududa

if if-list
then then-list
fi

F 'veafail'
date
if test -r veafail
  then
    cat veavail
    rm veafail
  else
    echo Sellel tunnil polnud vigu!
fi

olgu P, millel 4 pos. par-t, mida väljastab vastupidises järjekorras
rev
if test $# = 4
    then echo $4 $3 $2 $1
else echo $0 kasutamine: p1 p2 p3 p4
fi

if talv **06.03.97** **3.**
  then
    lumekoristus
    purikad
elif kevad
then
    aed_kevad
    muru
elif suvi
then
    aed_suvi
    muru
    ehitus
else
    echo See on ju sügis!
fi

while ja until while test -r junk **06.03.97** **4.**

while while-list do sleep 10; echo alles!
do do-list done
done

while test -r junk ; do sleep 10 ; echo alles! ; done

until until-list vastupidi!
do do-list
done

ootame mingi F-i tekkimist

while test ! -r raha ! - argument
do sleep 10; echo veel pole!
done

until test -r raha
do sleep 10; echo veel pole!
done

for

for sona ***in*** s1 s2 …
do do-list
done

muutujale $sona omistatakse väärtused s1, s2, …

for tudeng in aavo ants tom juta sirje
do echo $tudeng on tudeng
done

võib ka

for sona do-list täidetakse 1* iga pos. par. jaoks
do do-list NB! lihtsaim võimalus do-list täitmiseks
done argumentide jada jaoks

list=”” **06.03.97 5.**
for arg
  do list=”$arg $list”
done
echo $list
mida teeb? katseta!

case

case sona ***in***
  var1***)*** var1-list ***;;***
  var2***)*** var2-list ***;;***
  …
esac
sona võrreldakse kõikide vari-dega
on lubatud shell’i metasümbolid (*, ?, […])
ühele vari-list’ile võib vastada mitu väärtust, eraldaja - | (disjunktsioon)

break ja continue

kasutatakse while, until ja for töö juhtimiseks
break katkestab selle tsükli töö, milles ta asub
continue paneb tsükli tööle algusest (tsükli 1.-st operaatorist)

case'i näide

```
for kuu
  do case $kuu in
        1|3|5|7|8|10|12) echo $kuu.-s on 31 päeva ;;
        4|6|9|11) echo $kuu.-s on 30 päeva ;;
        2) echo $kuu.-s on 28 päeva ;;
        *) echo sellise numbriga kuud ei ole ! ;;
     esac
  done
```

```
$ katse 1                    1.-s on 31 päeva
$ katse 2                    2.-s on 28 päeva
$ katse 13                   sellise numbriga kuud ei ole !
$ katse 1 2 14
1.-s on 31 päeva
2.-s on 28 päeva
sellise numbriga kuud ei ole !
```

naide

```
loendi=$#
kask=echo
while test $loendi -gt 0
  do
    kask="$kask \$$loendi"
    loendi=`expr $loendi - 1`
  done
eval $kask
```

mida teeb?

eval evaluate

eval kindlustab argumentide asendamise, $kask tõlgendamist käsuna (sõna echo loetakse käsuks) ja selle täitmise

```
loendi=par arv                  4
kask=‘echo‘                     echo
while loendi > 0 do
        kask=kask & $loendi     echo $4
        loendi=loendi-1
                done
```