Командни процедури во UNIX

Командните интерпретери препознаваат т.н. команден јазик во кој можат да се пишуваат командни процедури. Командните интерпретери можат да извршуваат вакви програми. Командните процедури често се користат како административни алатки. Многу апликации се исто така напишани во вид на командни процедури.

Интерпретерите дозволуваат команди кои се протегаат повеќе од ширината на терминалот. Команда може да се продолжи во следна линија со додавање "\" на крајот на линијата. Командата може да се прошири и ако интерпретерот заклучи дека командата не е завршена, на пример отворени се наводници кои не се затворени. Во тој случај интерпретерот прикажува нов одзивен знак ">". Всушност обликот на тој одзивен знак се дефинира со системската променлива PS2. Командите кои се протегаат на повеќе линии интерпретерот ги третира како една команда.

Во случаи кога е потребно резултатот од извршување на команда да се смести во системска променлива може да се користи *grave* операторот (` команда ` каде наводниците се оние под копчето Escape). Тој предизвикува извршување на командата и пренасочување на стандардниот излез во променливата. Во следниот пример, содржината на тековниот именик се сместува во променливата листа:

```
student@os:~$ lista=`ls`
student@os:~$ echo "Sodrzinata na tekovniot imenik e: $lista"
```

додека со:

```
student@os:~$ echo "Ima `ls $HOME | wc -l` datoteki vo $HOME"
```

се прикажува бројот на датотеки во именикот $HOME. (Напомена уште еднаш: да се внимава на наводниците кај grave операторот!)

Командните процедури може да се сместуваат во датотеки. Еден начин да се извршат е да се пренесе името на датотеката во која е сместена процедурата како аргумент на команден интерпретер. На пример ако креираме командна процедура broi.sh во која ќе ги сместиме претходните команди, тогаш на следниов начин се извршуваат командите во датотеката broi:

```
student@os:~$ bash broi.sh
```

или пак со:

```
student@os:~$ chmod u+x broi.sh
student@os:~$ ./broi.sh
```

Често е погодно дефинирање на системски променливи кои ќе постојат само за време на извршувањето на процедурата.

```
#!/bin/bash
DIR=`pwd`
echo $DIR
```

Отако ќе се изврши процедурата `dir.sh` и притоа пробате:

```
student@os:~$ echo $DIR
```

ќе забележите дека нема да се испечати ништо. Значи, променливата `DIR` постои само за време на извршувањето на скриптата `dir.sh`. Слично и други команди кои се извршуваат во рамки на една скрипта не влијаат на надворешната сесија.

Повеќето команди враќаат нумеричка вредност која е достапна во системската променлива ?. Наредбата touch служи за креирање нова датотека (празна) или за измена на датум на пристап на веќе постоечка датотека.

```
#!/bin/bash
touch tmpdar
rm tmpdat
echo "Exit code of ok rm is $?"
rm tmpdat
echo "Exit code of failed rm is $?"
```

Со командата `read` се врши читање на влезна низа до ENTER. Првиот збор се доделува на првата променлива, вториот на втората итн. Ако има повеќе зборови отколку променливи, остатокот од зборови се доделува на последната променлива.

```
#!/bin/bash
echo "Vnesete ime, prezime i indeks"
read ime prez ind
echo "Vaseto ime e: $ime"
echo "Vaseto prezime e: $prez"
echo "Vasiot indeks e: $ind"
```

За на час

1. Направете скрипта која од корисникот чита број на индекс. Потоа печати колку пати тој студент е најавен на системот. Доколку не е најавен ниту еднаш, да се испечати соодветна порака.

```
#!/bin/bash
users="$(users)"
i=0
for user in $users
 do
  if [ $user = $1 ]; then
   i=$(($i+1))
  fi
 done
if [ $i -gt 0 ] ;then
 echo "$1 e najaven $i pati"
else
 echo "$1 ne e najaven"
fi
```

2. Направете скрипта која на секои 10 секунди ќе проверува дали корисникот "XXXXX" е најавен на системот, се додека тој не се најави (со користење на наредбата sleep n, каде n е бројот на секунди).

```
#!/bin/bash
while [ "0" = "0" ]
 do
  users=$(users)
  for user in $users
   do
    if [ $user = $1 ] ;then
    echo "$1 e najaven"
    exit 1
    fi
   done
  echo "$1 seuste ne e najaven"
  sleep 10
 done
```

3. Направете скрипта `backup` која врши копирање на сите датотеки со наставка `.txt` во именик со име `backup`. Доколку не постои именикот, прво да се креира.

```
#!/bin/bash
ls_list=$(ls)
tmp=0
for b in $ls_list
 do
  if [ $b = "backup" ] ;then
   echo "Hurray"
   tmp=1
   break
  fi
 done
if [ $tmp = 0 ] ;then
 mkdir backup
fi
cp *.txt backup
```

4. Направете скрипта која од низа од внесени три броја ги подредува по големина.

```
#!/bin/bash
niza=("$1 $2 $3")
sorted=( $(
for i in $niza ; do
 echo $i
done | sort) )
echo ${sorted[@]}
```

5. Да се избројат сите датотеки во домашниот именик чие што име започнува и завршува на број и во својата содржина го имаат зборот "if".

```
#!/bin/bash
dir=$(ls | grep '\b[0-9].*[0-9]\b')
count=0
for i in $dir; do
 tmp=$(grep 'if' $i | wc -l)
 count=$(($count+$tmp))
done
echo $count
```

6. Направете скрипта на која и се задаваат два аргументи на командната линија. Првиот е име на датотека или именик, а вториот може да има вредност 1 или 2. Ако првиот аргумент е 1, тогаш датотеката/именикот се брише, а ако е 2, се печати порака "Vnesi novo ime" и датотеката/именикот се преименува со ново име кое се внесува од тастатура.

```
#!/bin/bash
if [ $2 = 1 ] ; then
 rm -rf $1
elif [ $2 = 2 ] ; then
 echo "Vnesi novo ime"
 read tmp
 mv $1 $tmp
fi
```

7. Со користење на командата `select` да се напише командна процедура која ќе печати листа за избор на една од операциите: sobiranje, odzemanje, mnozenje и delenje. Во листата да има и посебен избор kraj за излез од процедурата. Листата за избор да се прикажува постојано се додека корисникот не избере крај. Процедурата потоа треба да го пресмета соодветниот резултат за два агрументи внесени од командна линија и да го прикаже на екран.

```
#!/bin/bash
select i in 'sobiranje' 'odzemanje' 'mnozenje' 'delenje' 'kraj' ; do
 case $i in
  sobiranje) read o1; read o2; echo "$o1 + $o2 = $(($o1+$o2))";;
  odzemanje) read o1; read o2; echo "$o1 - $o2 = $(($o1-$o2))";;
  mnozenje) read o1; read o2; echo "$o1 * $o2 = $(($o1*$o2))";;
  delenje) read o1; read o2; echo "$o1 / $o2 = $(($o1/$o2))";;
  kraj) break ;;
  *) echo "sad face " ;;
 esac
done
```

8. Направете скрипта која ќе ја определи вкупната големина на датотеките во тековниот директориум и сите негови поддиректориуми рекурзивно.

```
#!/bin/bash
du -sh
```