Упражнение 08

Командни процедури без и с позиционни параметри. Цикли

Комадни процедури без параметри

comproc1
```
echo LIST /home/KN
ls  /home/KN
```

```
echo
echo LIST  /home/student
ls /home/student
echo
ls LIST curdir
ls
```

Комадни процедури с позиционни параметри

Когато изпълняваме една командна процедура, можем да и дадем и някакви параметри:

```
comproc1 param1 param2 ...
```

Как да ги достъпим в нашата командна процедура?

Командният интерпретатор присвоява тези параметри на специални **системни променливи** 0, 1, 2 ... като всяко число n отговаря на n-тия подаден позиционен параметър.

!!! $0 винаги пази името на командата

Така ако изпълним командната процедура comproc1 по този начин:

```
comproc1 param1 param2 param3
```
То позиционните параметри ще за запазени по този начин:

```
echo $0  # comproc1
echo $1 # param1
echo $2 # param2
echo $3 # param3
```

Даване стойности на позиционните параметри – set

```
set arg1 arg2 arg3
echo $1 $2 $3
set `date`
echo $3 # day
```

Също така можем да "преместваме" стойностите на позиционните параметри наляво

`shift n` - измества стойностите на позиционните параметри с n позиции наляво

Съответно ако:

```
echo $1 $2 $3 # param1 param2 param3
```

и някъде в КП сме изпълнили

```
shift 2
```
то `echo $1 $2 $3 # param3`

В командна процедура можем да променим стойностите на променливите 0, 1, 2 ...

Отново командата е **set** `new_pos_arg1 new_pos_arg2`

Съответно ако сме имали:

```
echo $1 $2 $3 # param1 param2 param3
```

Ако някъде в КП имаме:

```
set new1 new2
```

то

```
echo $1 $2 $3 # new1 new2
```

Запазени стойности на позиционните параметри

- `*` - дава символен низ с параметрите от командния ред
- `@` - дава масив с параметрите от командния ред
- `#` - дава броя на параметрите от командния ред
- `$` - PID на текущия процес
- `?` - код на завършване на последния завършил процес
- `!` - PID на последния завършил процес във фонов режим

Цикли

while цикъл

```
while условие(команда)
do
   команда1
   команда2
   команда3
   ...
done
```

```
read user
until who | grep $user
do
echo "Waiting for user $user to login"
sleep 10
done
```

until цикъл

```
until условие(команда)
do
команда1
команда2
команда3
...
done
```

for цикъл

Синтаксис:

```
for променлива in [списък]
do
команда1
команда2
...
done
```

Примери:

```
for i in 1 2 3 4 5          for i  in 1 2 3 4 5
do                          do
echo  1 $i                  echo 1 $i
                            break/ continue
echo  2 $i                  echo 2 $i
done                        done
```

```
read user1 user2 user3
file=/etc/passwd
for user in $user1 $user2 $user3
write $user < $file
done
```

Изход: изпраща съдържанието на файла /etc/passwd на потребителите, прочетени от стандартния вход

```
x=1
for args in $@
echo "command line argument $x is $args"
x=$(expr $x + 1)
done
```

Изход: принтира на стандартния изход позиционните параметри подадени на командната процедура

break и continue

break [n]

Прекъсва n-тият цикъл отвътре-навън
ако не е зададен n то се прекъсва цикъла, в който е сложен break

continue [n]

skip-ват се всички команди след него и се продължава следващата итерация на n-тия цикъл
ако не е зададен n то се продължава с 1 итерация

Вложени цикли

```
while true
  do echo LOOP1
     until false
        do echo LOOP2
           while true
                 do echo LOOP3
                 done
       done
  done
```