Operacijski sistemi

vaje 5

Programiranje v BASH-u

- BASH - **B**ourne **A**gain **Sh**ell
- lupina za ukaze
- skriptni jezik
  - omogoča v celoti uporabiti zmožnosti lupine
  - avtomatizacija opravil (veliko programov v Linuxu so skriptni programi → `/etc/`)

- prevedeni programi ↔ skripte
  - hitrost, prenosljivost, prevajanje, …

Pisanje skript

- pozivna vrstica
  - interaktivni način
- skripta
  - neinteraktivni način (+ možnost interaktivnega načina)

pisanje skript

- poganjanje
- zaključek skripte (`exit`)
- komentarji (`#`)
- spremenljivke
  - deklaracija:

    ```
    ime_spremenljivke=vrednost
    ```

  - uporaba:

    ```
    ${ime_spremenljivke}
    $ime_spremenljivke
    ```

spremenljivke

```
${ime_spremenljivke}
$ime_spremenljivke
${ime_spremenljivke:-vrednost}
${ime_spremenljivke:=vrednost}
${#ime_spremenljivke}          … dolžina niza
${niz:poz}                     … podniz z leve
${niz:poz:dolžina}             … podniz z leve
${niz:(-poz)}                  … podniz z desne
${niz:(-poz):dolžina}          … podniz z desne
```

spremenljivke (odstranitev podniza)

- najkrajše ujemanje
  `${niz#podniz}`
- najdaljše ujemanje
  `${niz##podniz}`
- najkrajše ujemanje z zadnje strani
  `${niz%podniz}`
- najdaljše ujemanje z zadnje strani
  `${niz%%podniz}`

Kako delujejo naslednje operacije?

```
x="To je test, 123, 123."
echo ${x#* }
echo ${x##* }
echo ${x% *}
echo ${x%% *}
```

spremenljivke (zamenjava podniza)

- zamenja prvo ujemanje
  `${niz/podniz/zamenjava}`
- zamenja vsa ujemanja
  `${niz//podniz/zamenjava}`
- zamenja začetek niza
  `${niz/#podniz/zamenjava}`
- zamenja konec niza
  `${niz/%podniz/zamenjava}`

Kontrolne strukture

- primer programa: kopiraj.sh

```
#!/bin/bash
cp /etc/blabla .
echo "Končano."
```

Kakšna pomanjkljivost programa?

struktura: if … then … else … elif … fi

```
#!/bin/bash
if test -f /etc/blabla
then
  # datoteka obstaja, kopiraj, izpiši, da je uspešno
  cp /etc/blabla .
  echo "Končano."
else
  # datoteka ne obstaja, izpiši
  echo "Datoteka ne obstaja"
fi
```

Znamo napisati še kako drugače?

naloga

1. Napišite skripto v BASH-u, ki naredi trdo povezavo na datoteko, ki jo podamo kot prvi argument ob klicu. Povezavo naredi v trenutnem imeniku in naj se imenuje enako kot izvorna datoteka, če pa izvorna datoteka ne obstaja, pa naj se ustvari prazna datoteka z istim imenom v trenutnem imeniku.

```
$./1naloga.sh /etc/passwd
Končano
```

(ustvari trdo povezavo z imenom passwd)

```
$./1naloga.sh /etc/passwd2
Datoteka ustvarjena
```

(ustvari novo datoteko z imenom passwd2)

testiranje

- `if test -f "/etc/blabla"; then`
- `if [ -f "/etc/blabla" ]; then`
- `if [[ -f /etc/blabla ]]; then`

- `if test "$ime" -eq 5; then`
- `if ["$ime" -eq 5 ]; then`
- `if [[ $ime -eq 5 ]]; then`

ustvari.sh

```
#!/bin/bash
if test -f "$1"; then
   ln "$1" $(basename "$1")
   echo "Končano."
else
   touch $(basename "$1")
   echo "Datoteka ustvarjena."
fi
```

AND in OR

- pogoj1 && pogoj2
- pogoj1 || pogoj2

```
#!/bin/bash
x=5
y=10
if [ "$x" -eq 5 ] && [ "$y" -eq 10 ]; then
  echo "Oba pogoja sta resnična."
else
  echo "Pogoja nista resnična."
fi
```

AND in OR

- pogoj1 && pogoj2
- pogoj1 || pogoj2

```
#!/bin/bash
x=3
y=2
if [ "$x" -eq 5 ] || [ "$y" -eq 2 ]; then
  echo "En od pogojev je resnicen."
else
  echo "Noben pogoj ni resnicen."
fi
```

case … in … esac

```
#!/bin/bash
x=5 # prednastavimo x na 5
# preveri vrednost x:
case $x in
  0) echo "Vrednost x je 0."
     ;;
  5) echo "Vrednost x je 5."
     ;;
  9) echo "Vrednost x je 9."
     ;;
  *) echo "Nepoznana vrednost."
esac
```

case ↔ if

```
#!/bin/bash
x=5 # prednstavimo x na 5
if [ "$x" -eq 0 ]; then
  echo "Vrednost x je 0."
elif [ "$x" -eq 5 ]; then
  echo "Vrednost x je 5."
elif [ "$x" -eq 9 ]; then
  echo "Vrednost x je 9."
else
  echo "Nepoznana vrednost."
fi
```

aritmetika

- ukaz `expr`
- primer uporabe: `expr 1 + 2`
- katere operacije?

- vgrajenost?
- `$((...))`

aritmetika

```
#!/bin/bash
x=8
y=4
z=$(expr $x + $y)
echo "Vsota števil $x + $y je $z"
```

aritmetika

```
#!/bin/bash
x=8
y=4
z=$(($x + $y))
echo "Vsota števil $x + $y je $z"
```

aritmetika

```
#!/bin/bash
x=5 # prenastavimo x na 5
y=3 # prenastavimo y na 3
add=$(($x + $y))
sub=$(($x - $y))
mul=$(($x * $y))
div=$(($x / $y))
mod=$(($x % $y))
# izpišemo rezultate:
echo "vsota: $add"
echo "razlika: $sub"
echo "produkt: $mul"
echo "kolicnik: $div"
echo "ostanek: $mod"
```

while … do … done

```
#!/bin/bash
while true; do
   echo "Za izhod pritisni CTRL-C."
done
```

- vgrajen?

while … do … done

```
#!/bin/bash
while :; do
   echo "Za izhod pritisni CTRL-C."
done
```

- vgrajen?

while … do … done

```
#!/bin/bash
x=0;  #prenastavimo x na 0
while [ "$x" -le 10 ]; do
  echo "Trenutna vrednost spremenljivke x: $x"
  x=$x+1     # povečamo vrednost x?
  sleep 1
done
```

- Kaj dela ta koda?

while … do … done

```
#!/bin/bash
x=0;  #prenastavimo x na 0
while [ "$x" -le 10 ]; do
   echo "Trenutna vrednost spremenljivke x: $x"
   x=$x+1     # povečamo vrednost x?
   sleep 1
done
```

- Kako bi popravili?

while … do … done

```
#!/bin/bash
x=0;  #prenastavimo x na 0
while [ "$x" -le 10 ]; do
  echo "Trenutna vrednost spremenljivke x: $x"
  x=$(expr $x + 1)     #povečamo vrednost x
  sleep 1
done
```

until … do … done

```
#!/bin/bash
x=0
until [ "$x" -gt 10 ]; do
   echo "trenutna vrednost x: $x"
   x=$(expr $x + 1)
   sleep 1
done
```

for … in … do … done

```
#!/bin/bash
for x in papir svincnik pero; do
  echo "Vrednost spremenljivke x je: $x"
  sleep 1
done
```

for … in … do … done

```
#!/bin/bash
echo -n "Kontrola sistema za napake"
for dots in 1 2 3 4 5 6 7 8 9 10; do
  echo -n "."
  sleep 1
done
echo "Sistem je pregledan."
```

for … in … do … done

```
#!/bin/bash
for datoteka in *; do
  echo "Dodaj koncnico .html datoteki $datoteka..."
  mv $datoteka $datoteka.html
  sleep 1
done
```

naloge

2. S pomočjo (a) zanke for in (b) zanke while napišite skriptni program, ki bo štel od 1 do 500 v enosekundnih intervalih.
3. Napišite skripto, ki bo združila vse navadne datoteke iz trenutnega imenika v pakete **`tar`**. V posameznem paketu naj bodo vse datoteke, ki imajo prvih n črk imena istih. Ime paketa naj bo sestavljeno iz prvih skupnih n črk in končnice **`tar`**. Število črk n podamo kot argument skripte.
4. Napišite skripto, ki bo združila vse navadne datoteke iz trenutnega imenika v pakete **`tar`**. V posameznem paketu naj bodo vse datoteke, ki imajo isto končnico (torej imajo isti niz za zadnjo piko). Ime paketa naj bo ime končnice. Če končnice ni, napišite `null`.

zanka for

```
for var in spisek ; do
  ukazi
done
```

- od BASH 2.04 naprej:

```
for (( inicializacija ; pogoj ; inkrementiranje )); do
  ukazi
done
```

zanka for

```
for i in `seq 24 42`; do
  echo -n "$i "
done
```

```
for ((i=24; i<=42; i++)); do
  echo -n "$i "
done
```

Uporaba tabel

```
tabela=(rdeča oranžna rumena zelena modra vijolična)
dolzina=${#tabela[*]}
echo "Mavrica ima $dolzina barv:"
i=0
while [ $i -lt $dolzina ]; do
        echo "$((i+1)): ${tabela[$i]}"
        let i++
done
```

Kaj naredi naslednja skripta?

```
y=1
while [ $y -le 12 ]; do
      x=1
      while [ $x -le 12 ]; do
            printf "% 4d" $(( $x * $y ))
            let x++
      done
      echo ""
      let y++
done
```

branje vhoda: ukaz read

- interakcija z uporabnikom

```
#!/bin/bash
# prebere ime uporabnika in izpiše dobrodošlico
echo -n "Vpisi svoje ime: "
read user_name
echo "Pozdravljen $user_name!"
```

branje vhoda: ukaz read

```
#!/bin/bash
# prebere ime uporabnika in izpiše dobrodošlico
echo -n "Vpisi svoje ime: "
read user_name
# ce uporabnik ne vnese nic:
if [ -z "$user_name" ]; then
  echo "Nisi mi povedal svojega imena!"
  exit 10
fi
echo "Pozdravljen $user_name"
exit 0
```

branje vhoda: ukaz read

```
#!/bin/bash
# prebere ime uporabnika in izpiše dobrodošlico
echo -n "Vpisi svoje ime: "
read user_name
# ce uporabnik ne vnese nic:
if [ -z "$user_name" ]; then
  echo "Nisi mi povedal svojega imena!"
  exit 10
fi
echo "Pozdravljen $user_name"
exit 0
```

- Naloga: Kako bi spremenili program, da bi spraševal uporabnika po imenu dokler ga le-ta ne vpiše?

branje vhoda: ukaz read

```
#!/bin/bash
while [ -z $user_name ]; do
  echo -n "Vpisi svoje ime: "
  read user_name
  if [ -n "$user_name" ]; then
    echo "Pozdravljen $user_name"
    exit 0
  fi
  echo "Nisi mi povedal svojega imena!"
done
exit 1
```

naloga

4. Napišite skripto, ki bo v podanem imeniku izpisala vse mehke povezave, ki ne kažejo na neko končno datoteko ali imenik (v verigi povezav manjka kakšen člen ali pa ciljna datoteka ne obstaja).

   Za vsako takšno povezavo naj skripta izpiše verigo povezav do člena, ki manjka.

   Imenik skripta prejme tako, da uporabniku izpiše poziv in čaka, da uporabnik vpiše imenik.

   Ko skripta obdela podani imenik in izpiše problematične mehke povezave, ponovno izpiše uporabniku poziv za vpis novega imenika itd.

   Če uporabnik ne vpiše imenika v petih sekundah, se skripta ustavi.

branje iz datoteke

```
datoteka=$1
while read vrstica; do
    echo $vrstica
done < $datoteka
```

```
datoteka=$1
cat $datoteka | while read vrstica; do
    echo $vrstica
done
```