Geografické analýzy a aplikácie v GIS

Hana Stanková, Tomáš Schmidt

Skriptovanie v Bash

Lekcia 3

Náplň lekcie

1. Bash skript
2. Spúšťanie skriptov
3. Premenné
4. Formátovanie reťazcov
5. Podmienky
6. Cykly

Skriptovanie v Bash

shell - časť OS, ktorá definuje správanie terminálu (vykonávanie príkazov atď.)

bash - druh shell-u (Bourne Again SHell)

skript - textový súbor obsahujúci množinu príkazov, ktoré sa normálne dajú spustiť aj v príkazovom riadku (musia byť udelené práva na spúšťanie)

- shell skript má koncovku .sh

```
#!/bin/bash
# ukazka skriptu
echo 'Dobre rano!'
```

Spúšťanie skriptov

```
./mojskript.sh
```

- ak sa zadá iba názov príkazu/skriptu, Bash ho hľadá na miestach, ktoré sú definované v premennej $PATH

```
echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:
```

- ak je skript uložený inde (napr. v domovskom adresári), zadávame absolútnu alebo relatívnu cestu k skriptu

```
/home/student/mojskript.sh
```

Shebang (#!)

```
#!/bin/bash
```

- za kombináciou znakov **#!** sa zadá cesta (väčšinou absolútna) k Bash interpreteru
- musí byť napísaný v prvom riadku súboru bez medzier

Komentovanie

```
# skript na prevzorkovanie
```

- všetko za znakom **#** berie Bash ako komentár a neinterpretuje to ako príkaz
- pri skriptovaní/programovaní je dobrým zvykom písať vysvetľujúce komentáre

Premenné

- miesta na dočasné ukladanie informácií

- môžeme do nich vkladať hodnoty

- môžeme z nich čítať hodnoty

```
#!/bin/bash
$meno='Hana'
echo "Dobre rano, $meno!"
```

```
./pozdrav.sh
Dobre rano, Hana!
```

- dvojité úvodzovky umožňujú nahradenie

Premenné

- pri odkaze na premennú alebo pri čítaní z premennej v Bash skriptoch dávame pred názov premennej znak **$** (pri vkladaní hodnôt do premennej sa nedáva)
- hodnotu premennej môžeme načítať aj zo vstupu od užívateľa ako parameter:

```
#!/bin/bash
# skript na vypis informacii o rastrovej vrstve
gdalinfo $1
```

```
./mojvypis.sh b02.tif
```

- vstupných premenných môže byť aj viac, číslujeme ich podľa poradia ($1, $2, ... , $9)

Cvičenia

Vytvorte skript **resample.sh**, ktorý prevzorkuje vstupný raster **b2.tif** podľa zadaného rozlíšenia (premenná 1) a uloží ho pod zadaným názvom (premenná 2). Použite nástroj **gdalwarp** a metódu prevzorkovania **bilinear**. Spustite skript so zadaným rozlíšením 5 m a 20 m.

Manipulácia s reťazcami

- výsek z reťazca:

`${string:position}` - výsek od pozície danej indexom (začína od 0)

`${string:position:length}` - výsek od pozície s určenou dĺžkou

```
str='Dobre rano!'
echo ${str:6}    #rano!
echo ${str:6:4}     #rano
```

Manipulácia s reťazcami

- odstraňovanie výseku z reťazca:

`${string#substring}` - odstránenie výseku zo začiatku reťazca

`${string%substring}` - odstránenie výseku z konca reťazca

- odstránenie koncovky z názvu súboru:

```
filename='b02.tif'
echo ${filename%.tif}       #b02
echo ${filename%.*}         #b02
```

Manipulácia s reťazcami

- skladanie reťazcov (string concatenation):

```
str1='Dobre'
str2='rano'
pozdrav="$str1 $str2"
echo $pozdrav       #Dobre rano
pozdrav="$str1 $str2!"
echo $pozdrav       #Dobre rano!
pozdrav=$str1"_"$str2
echo $pozdrav       #Dobre_rano
```

Cvičenia

1. Zmeňte skript **resample.sh** tak, aby nebolo nutné zadávať názov výstupného súboru, ale tento sa vygeneruje automaticky na základe názvu vstupného rastra a zadaného rozlíšenia.
2. Zmeňte skript **resample.sh** tak, aby sa dal zadať názov vstupného súboru.

Podmienky

- vetvenie behu programu (vykonajú sa rôzne príkazy na základe podmienok)

- kontrola vstupov

```
if [ <some test> ]
then
    <commands>
fi
```

```
if [ $str1 = $str2 ]
then
    echo 'Reťazce sa rovnajú.'
fi
```

- ak je podmienka splnená, vykoná sa všetko medzi **then** a **fi**

- hranaté zátvorky odkazujú na program **test**, ktorý vyhodnocuje podmienky

- odsadenie (indent) - nie je povinné, ale sprehľadňuje kód skriptu

Podmienky

- operátory porovnania:

= rovná sa (porovnanie reťazcov)

!= nerovná sa (porovnanie reťazcov)

-eq rovná sa (numerické porovnanie)

-gt je väčšie ako (numerické porovnanie)

-lt je menšie ako (numerické porovnanie)

!EXPRESSION výraz je nepravdivý (negovanie podmienok)

-n string dĺžka reťazca je väčšia ako nula

-z string dĺžka reťazca je nula (prázdny reťazec)

Podmienky

- operátory kontroly súborov:

-e *file* súbor existuje

-d *file* súbor existuje a je to adresár

-r *file* súbor existuje a je udelené právo *read*

-w *file* súbor existuje a je udelené právo *write*

-x *file* súbor existuje a je udelené právo *execute*

-s *file* súbor existuje a jeho veľkosť je väčšia ako nula (nie je prázdny)

Cvičenia

Doplňte do skriptu **resample.sh** kontrolu, či vstupný súbor existuje. Otestujte skript s existujúcim aj s neexistujúcim súborom.

Else

- umožňuje vykonanie iných príkazov, ak podmienka nie je splnená

```
if [ <some test> ]
then
    <commands>
else
    <commands>
fi
```

```
if [ $str1 = $str2 ]
then
    echo 'Reťazce sa rovnajú.'
then
    echo 'Reťazce sa nerovnajú.'
fi
```

Elif

- umožňuje vetvenie na viac ako dve možnosti (elif = else if)

```
if [ <some test> ]
then
    <commands>
elif [ <some test> ]
then
    <commands>
else
    <commands>
fi
```

```
if [ $num1 -gt $num2 ]
then
    echo 'num1 je väčšie ako num2.’
elif [ $num1 -lt $num2 ]
then
    echo 'num1 je menšie ako num2.'
else
    echo 'num1 sa rovná num2.'
fi
```

Vnorené podmienky

- umožňujú vytvárať hierarchickú štruktúru podmienok

```
if [ <some test> ]
then
    if [ <some test> ]
    then
        <commands>
    else
        <commands>
    fi
else
    <commands>
fi
```

```
if [ $num1 -eq 1 ]
then
    if [ $num1 = 1 ]
    then
        echo 'num1 je presne 1.'
    else
        echo 'num1 sa numericky = 1.'
    fi
else
    echo 'num1 sa nerovná 1.'
fi
```

Cvičenia

1. Doplňte do skriptu **resample.sh** upozornenie pre používateľa, ak zadá názov neexistujúceho súboru. Otestujte skript s neexistujúcim súborom.
2. Doplňte do skriptu **resample.sh** kontrolu, či je vstupný súbor vo formáte **.tif**. Ak nie je, upozornite používateľa. Otestujte skript so súborom vo formáte .tif a .jp2.

Spájanie podmienok

- umožňuje otestovať viacero podmienok

- buď musia platiť podmienky súčasne - operátor AND: **&&**

```
if [ $num1 -gt 0 ] && [ $num1 -lt 2 ]
then
    echo 'num1 je kladné a menšie ako 2.'
fi
```

alebo musí platiť aspoň jedna z nich - operátor OR: ||

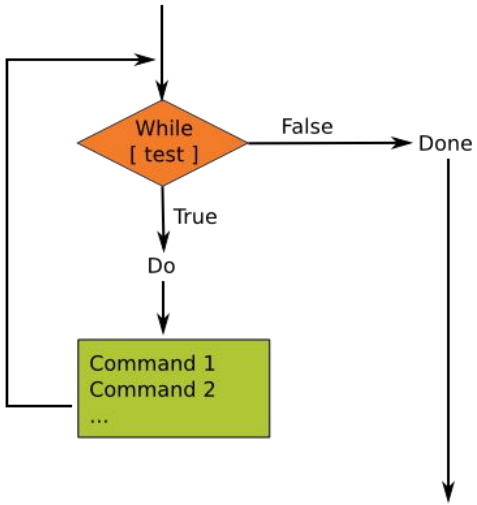
```
if [ $num1 = 1 ] || [ $num1 = 2 ]
then
    echo 'num1 sa rovná 1 alebo 2.'
fi
```

Cvičenia

Doplňte kontrolu formátu v skripte **resample.sh** o formát **.jp2**. Ak vstupný súbor nie vo formáte .tif alebo .jp2, upozornite používateľa. Otestujte skript s existujúcim súborom v inom formáte (napr. shp).

Cykly

- vykonávanie série príkazov dovtedy, kým nie je splnená podmienka
- tri druhy cyklov v Bash: **while**, **until** a **for**



While

- príkazy sa vykonávajú dovtedy, kým platí (je splnená) podmienka

```
while [ <some test> ]
do
    <commands>
done
```

```
i=1
while [ $i -lt 11 ]
do
    ((i++))
done
```

- počítadlo od 1 do 10

Until

- príkazy sa vykonávajú dovtedy, kým nie je splnená podmienka

```
until [ <some test> ]
do
    <commands>
done
```

```
i=1
until [ $i -gt 10 ]
do
    ((i++))
done
```

- môžeme si vybrať medzi cyklom while a until podľa toho, ktorý dáva väčší zmysel v našom kóde (aby bol kód čo najviac zrozumiteľný)

Čítanie riadkov súboru

- cyklus while sa dá použiť na prechádzanie po riadkoch súboru

- čítanie riadkov:

```
while read line
do
    echo $line
done < list.txt
```

Čítanie súradníc zo súboru

- ak sú v súbore uložené súradnice vo formáte x y alebo iné atribúty oddelené medzerou, môžeme ich načítať do samostatných premenných:

```
while read x y
do
    echo $x $y
done < xy.txt
```

Cvičenia

1. Uložte výpis obsahu pracovného adresára do textového súboru a potom vypíšte jednotlivé riadky na obrazovku pomocou cyklu while.
2. Pomocou textového editora nano vytvorte súbor so súradnicami aspoň troch fiktívnych bodov a vypíšte jednotlivé súradnice na obrazovku pomocou cyklu while.

For

- umožňuje listovanie v zozname
- príkazy sa vykonávajú pre každú položku v zozname

```
for var in <list>
do
    <commands>
done
```

```
fruits='apple plum apricot'
for fruit in $fruits
do
    echo $fruit
done
```

- názov položky v zozname (var) je voliteľný

For

- počítadlo so zadaným rozsahom (prípadne aj rozostupom) hodnôt:

```
for value in {range}
do
    <commands>
done
```

```
for value in {1..10}
    do
        echo $value
    done
```

- rozsahom sa zadáva v tvare **{start..stop}** alebo **{start..stop..step}**

- pomocou cyklu for môžeme tiež listovať v súboroch v adresári:

```
for file in *.jpg
do ...etc.
```

Cvičenia

Vytvorte skript clip.sh, ktorý oreže všetky rastre vo formáte **.tif** v pracovnom adresári podľa vektorovej vrstvy **BA_okresy.shp** a uloží ich pod pôvodným názvom s príponou **_clip**.