Pracownia Technologii Informacyjnej

Zajęcia 9, skrypty powłoki, cd. (instrukcje sterujące)

Grzegorz Grzelak

Zakład Cząstek i Oddziaływań Fundamentalnych
IFD UW

e-mail: grzelak@fuw.edu.pl
WWW: http://www.fuw.edu.pl/~grzelak/PTI

10 grudzień 2018

Skrypty, instrukcje sterujące

Instrukcje warunkowe

- `if ( [` *wyrażenie testujące* `] ) then`
  polecenie_1
  ...
  `else`
  polecenie_2
  ...
  `fi`
- `if [` *wyrażenie testujące* `]`
  `then`
  polecenie_1
  ...
  `else`
  polecenie_2
  ...
  `fi`

Spacje są ważne !

Testy dla plików, testy dla zmiennych

```
if [ -e config.txt ]
then
echo Plik config.txt istnieje
fi
```

Inne testy:

`-e plik`	plik istnieje
`! -e plik`	plik nie istnieje
`-d plik`	plik jest katalogiem
`-f plik`	plik jest zwykłym plikiem
`-s plik`	plik ma niezerowy rozmiar
`-r plik`	plik można odczytać
`-w plik`	plik można modyfikować
`-x plik`	plik wykonywalny
`-v zmienna`	zmienna istnieje

Instrukcja warunkowa w `bash` - porównywanie ciągów tekstowych

Wyrażenia do porównywania ciągów tekstowych:

`"$c1" = "$c2"`	ciągi są jednakowe
`"$c1" != "$c2"`	ciągi nie są jednakowe
`"$c1"`	ciąg jest niepusty
`-z "$c1"`	długość ciągu jest zerowa
`-n "$c1"`	długość ciągu jest niezerowa

Cudzysłowy ważne, jeżeli ciąg ze spacjami.

Przykład:

```
napis="Ala ma kota"
if [ "$napis" = "Ala ma kota" ] then
     echo "Napisy są takie same"
else
     echo "Napisy są różne"
fi
```

Pętle w bash

- `for` *zmienna* `in` *lista*
  `do`
  polecenie_1
  polecenie_2
  ...
  polecenie_n
  `done`
- `for` *nazwa_pliku* `in` *wzorzec*
  `do`
  polecenie_1
  polecenie_2
  ...
  polecenie_n
  `done`

Pętle w bash - przykłady

- ```
  for i in 1 3 5 7 9
  do
        echo $i
  done
  ```
- ```
  for plik in *
  do
        echo $plik
  done
  ```

Wczytywanie danych z klawiatury

- `read NAZWA`
  Przykład:
  echo ”Podaj swoje Imie i Nazwisko”
  read NAZWISKO
  echo ”Witaj uzytkowniku” $NAZWISKO
  Wynik na ekranie:

```
Podaj swoje Imie i Nazwisko
Jan Kowalski
Witaj uzytkowniku Jan Kowalski
```

Instrukcje warunkowe w `bash`

- `if` oczekuje wywołania polecenia
- Powodzenie wykonywanej operacji: kod powrotu `0` (`true`)
- Niepowodzenie wykonywanej operacji: kod powrotu różny od `0` (`false`)
- `if (`*warunek*`) then`
  polecenie_1
  polecenie_2
  ...
  polecenie_n
  `fi`
- ```
  if (make>/dev/null  2>/dev/null ) then
        echo makefile OK
  else
        echo nie ma pliku makefile ?
  fi
  ```


  ```

Instrukcja warunkowe (`bash`)

- `if (`*warunek1*`) then`
  polecenie_1
  ...
  `else`
  polecenie_2
  ...
  `fi`
- `if (`*warunek1*`) then`
  polecenie_1
  ...
  `elif (`*warunek2*`) then`
  polecenie_2
  ...
  `else`
  polecenie_3
  `fi`

Instrukcja warunkowa w `bash` - porównywanie liczb

- Standardowo instrukcja `if` w `bash` służy do testowania wyników poleceń
- Do prównywania liczb, ciągów tekstowych, testowania plików służy polecenie `test`
- Przykład:

```
x=15
y=25
if(test $x -eq $y) then
      echo $x=$y
else
      echo $x!=$y
fi
```

- Wyrażenia do porównywania liczb: `-eq`, `-ne`, `-gt`, `-ge`, `-lt`, `-le`

Pętle w bash

- Pętla w stylu języka C (w `bash`, nie `tcsh`) `max=`*górna_granica*
  `for((i=`*wartość_początkowa*`;i<=max;`*krok*`))`
  `do`
  polecenie_1
  polecenie_2
  . . .
  polecenie_n
  `done`

Pętle w bash - przykłady

- ```
  for i in 1 3 5 7 9
  do
        echo $i
  done
  ```
- ```
  for plik in *
  do
        echo $plik
  done
  ```
- ```
  max=10
  for((i=1;i<=max;i=i+2))
  do
        echo "Wartosc maksymalna= " $max
        echo "i= " $i
  done
  ```


  ```

Inne pętle

- bash
  while [*wyrażenie testujące*]
  do
  polecenie_1
  ...
  done

Operatory logiczne

- `-a` operacja logiczna AND
- `-o` operacja logiczna OR
- `!` negacja

Użycie np. `if [ -r $x -eq $y -a $x -eq $z ]...`