Programarea de sistem și de rețea

Lucrare de laborator nr. 5

Scripturi shell UNIX

Acest laborator are drept scop familiarizarea cu procedurile de scrire şi rulare a scripturilor shell-urilor Unix.

Un script UNIX este un program care poate fi interpretat de către shell-ul UNIX al calculatorului dumneavoastră. El este compus din :

- comenzi UNIX;
- instrucţiuni de control;
- comentarii.

Creaţi un director LAB4 în care veţi depune fişierele corespunzătoare scripturilor UNIX realizate în acest laborator.

Noțiuni generale

Variabile de mediu

Variabilele de mediu pot sa conţină ca valoare un sir de caractere. Atribuirea de valori se face astfel:

variabila=valoare

În UNIX exista câteva variabile predefinite.

- variabile *read-only*, actualizate de interpretor:
    - **$?** - codul returnat de ultima comanda executata
    - **$$** - identificatorul de proces al interpretorului de comenzi
    - **$!** - identificatorul ultimului proces lansat în paralel
    - **$#** - numărul de argumente cu care a fost apelat fişierul de comenzi curent
    - **$0** - conţine numele comenzii executate de interpretor
    - **$1, $2** ... - argumentele cu care a fost apelat fişierul de comenzi care se afla în execuţie

- variabile iniţializate la intrarea în sesiune:
    - **$HOME** - numele directorului "home" afectat utilizatorului;
    - **$PATH** - căile de căutare a programelor;
    - **$PS1** - prompter-ul pe care îl afişează interpretorul atunci când aşteaptă o comanda;
    - **$PS2** - al doilea prompter;
    - **$TERM** - tipul terminalului pe care se lucrează.

Directive de control (informaţii succinte)

Directivele de control ale interpretorului bash (sau sh**)** sunt structurile de limbaj care pot fi utilizate în scrierea de programe. În continuare vor fi prezentate câteva din cele mai folosite structuri de control.

Pentru detalii consultaţi pagina de manual a interpretorului de comenzi bash (şi/sau sh), analizaţi cu

atenţie directivele şi facilităţile pe care acesta le pune la dispoziţie.

Instrucţiuni de decizie

- *Instrucţiunea **if***

if *lista1*
then *lista2*
else *lista3*
fi

if *lista1*
then *lista2*
elif *lista3*
then *lista4*
else *lista5*
fi

O comanda returnează o valoare la terminarea ei. În general, dacă o comanda s-a terminat cu succes ea va returna 0, altfel va returna un cod de eroare nenul.

În prima forma a comenzii **if**, se executa *lista1*, iar dacă *si* ultima instrucţiune din lista returnează codul 0 (succes) se executa *lista2*, altfel se executa *lista3*.

În a doua forma se pot testa mai multe condiţii: când *lista1* se termina cu succes, se va executa *lista2*, altfel se executa *lista3*. Dacă aceasta se termina cu succes se executa *lista4*, altfel se executa *lista5*.

- *Instrucţiunea **case***

case *cuvant* **in**
tipar1) *lista1*;;
tipar2) *lista2*;;
...
esac

Aceasta instrucţiune implementează decizia multipla. Şablonul *tipar* este o construcţie care poate conţine simbolurile ? si *, similara celor folosite la specificarea generica a numelor de fişiere. Comanda expandează (evaluează) şirul *cuvant* şi încearcă sa îl potrivească pe unul din tipare. Va fi executata lista de comenzi pentru care aceasta potrivire poate fi făcută.

Instrucţiuni de ciclare

- *Instrucţiunea **while***

while *lista1*
do *lista2*
done

Se executa comenzile din *lista2* în mod repetat, cat timp lista de comenzi *lista1* se încheie cu cod de succes.

- *Instrucţiunea **until***

until *lista1*
do *lista2*

done

Se executa comenzile din *lista2* in mod repetat, pana când lista de comenzi *lista1* se încheie cu cod de succes.

- *Instrucțiunea **for***

for *variabila* [**in** *val1 val2 ...*]
do *lista*
done

Se executa lista de comenzi în mod repetat, variabila luând pe rand valorile *val1, val2, ...* Dacă lipseşte cuvântul cheie **in**, valorile pe care le va lua pe rand *variabila* vor fi parametrii din linia de comanda pe care i-a primit fişierul de comenzi atunci când a fost lansat în execuție.

Alte comenzi

- **break** - permite ieşirea din ciclu înainte de îndeplinirea condiției;
- **continue** - permite reluarea ciclului cu următoarea iteratie, inainte de terminarea iteratiei curente;
- **exec** *cmd* - comenzile specificate ca argumente sunt executate de interpretorul de comenzi în loc sa se creeze procese separate de execuție; dacă se dorește rularea comenzilor în procese separate ele se scriu direct, asa cum se scriu si in linia de comanda
- **shift** - realizează deplasarea argumentelor cu o poziție la stânga ($2$1, $3$2 etc);
- **wait** [*pid*] - permite sincronizarea unui proces cu sfârşitul procesului cu pid-ul indicat sau cu sfarsitul tuturor proceselor "fii";
- **expr** *expresie* - permite evaluarea unei expresii.

Mersul lucrării

Primul script “Hello World”

Orice script trebuie sa înceapă cu linia
`#!/bin/bash`

Pentru a afla adresa bash exactă de pe calculatorul Dumneavoastră utilizați instrucțiunea which:

`which bash`

Această linie este utilizată pentru a specifica faptul ca scriptul dumneavoastră trebuie interpretat de către shell-ul bash care se găseşte la adresa indicată.

Scriptul care permite afişarea unui mesaj de întâmpinare “Buna ziua !” este expus mai jos :

```
#!/bin/bash
echo Buna ziua !
```

Mai departe este expus procesul prin care se editează scriptul shell şi se rulează :

```
$ cat > BunaZiua.sh
#!/bin/bash

echo Buna ziua !

Ctrl+C
chmod +x BunaZiua.sh
./BunaZiua.sh
Buna ziua !
$
```

Orice script shell trebuie salvat într-un fişier cu un nume NumeScript.sh , extensia este opţională. Fişierul trebuie să fie cu drept de executare. Execuţia scriptului se realizează cu ajutorul comenzii.

```
$bash calea-catre-scriptul-dumneavoastra
```
sau
```
$./nume-script
```

Scripturi simple

1. Creaţi un script care afişează numărul de argumente ce i-au fost transmise şi le afişează în continuare unul câte unul. În cazul când nu i-au fost transmise argumente, ea emite mesajul: "Nici un argument transmis". Modificaţi scriptul pentru a afişa argumentele în ordine inversa
2. Analizaţi scriptul următor:

```
#!/bin/bash
((test $1 -lt $2) && (echo '$1 < $2')) || (echo '$2 < $1')
```

   Ce face el ? După testarea scriptului înlocuiţi ghilimelele simple cu ghilimele duble. Lansaţi din nou scriptul. Ce aţi constatat? Explicaţi.
   Studiaţi codul returnat după executarea acestui script. Explicaţi.
   Rescrieţi scriptul utilizând structura *if then else fi*

3. Când executaţi un script, un shell nou demară pentru a executa instrucţiunile conţinute în script. Sa testam aceasta afirmaţie cu un exemplu. Creaţi un script cu următorul conţinut:

   ```
   echo PID-ul meu este $$
   ```

   Variabila specială $$ ne da PID-ul procesului. Executaţi scriptul de mai multe ori şi comparaţi rezultatele. Ce constataţi? Vi se pare normal? Explicaţi.

Exerciţii

1. Modificaţi scriptul **`BunaZiua.sh`** redenumindu-l în **`BunaZiuaNume.sh`** după cum urmează:
   a) dacă îi transmiteţi doua argumente, primul conţinând numele iar al doilea prenumele, el trebuie să le afişeze după salutare (de ex. Buna ziua, Nume Prenume)
   b) dacă nu-i transmiteţi nici un argument, el trebuie sa va afişeze după salutare numele de conectare al utilizatorului (LOGNAME)
2. Realizaţi un script shell UNIX care permite "curăţarea" ecranului iar apoi afişează structura de fişiere şi directoare a directorului pe care l-aţi furnizat ca argument. *Indicaţie*: comanda care "curăţă" ecranul este *clear*. Studiaţi pagina de manual a acestei comenzi ! În cazul când scriptul este lansat fără de argument el va afişa structura de fişiere şi directoare a dosarului personal al utilizatorului.
3. Realizaţi un script shell UNIX care cere utilizatorului sa introducă doua şiruri de caractere şi apoi afişează un mesaj de informare dacă cele doua şiruri sunt sau nu egale. Mai exact, dacă

cele doua şiruri de caractere sunt egale se afişează la terminal un mesaj de tipul “Cele doua şiruri de caracter sunt egale”, iar dacă sunt diferite - un mesaj de tipul “Cele doua şiruri de caractere sunt diferite”.

4. Realizaţi un script care primeşte drept argument calea către un director, verifica dacă acesta este un director şi în caz afirmativ afişează conţinutul acestuia şi al subdirectoarelor care fac parte din el.
5. Realizaţi un script care permite copierea unui director specificat de către utilizator ca argument, cu întreg conţinutul sau de fişiere şi directoare într-un director numit *copie*. Dacă fişierul specificat nu exista, scriptul va anunţa despre asta şi se închide.
6. Realizaţi un script shell UNIX care calculează suma primelor cinci numere pare, utilizând un ciclu *while*.
7. Realizaţi un script care permite afişarea denumirii unei luni, în situaţia în care utilizatorul specifica numărul lunii. De exemplu pentru 3 afişează “martie”.
8. Analog, creaţi un script shell UNIX care nu afişează decât subdirectoarele dintr-un anumit director furnizat ca argument.
9. Realizaţi un script shell UNIX care permite afişarea numărului de fişiere şi de subdirectoare dintr-un director furnizat drept argument (verificaţi pentru început dacă directorul exista). Se vor lua în considerare doua cazuri :
   a) căutare superficiala (limitată la directorul curent fără a va preocupa de subdirectoare);
   b) căutare în profunzime (căutare în toate sub-directoarele din dosarul de baza)
10. Scrieţi un script care afişează numerele de la 1 până la valoarea *N* transmisa scriptului în calitate de parametru:
    a) Folosiţi în acest scop o buclă *while do done*
    b) Rescrieţi script-ul folosind structura *until do done.*
    c) *S*chimbaţi script-ul pentru a calcula media tuturor valorilor afişate anterior.
11. Scrieţi un script care vă cere numele unui dosar (îl vom numi dosar rădăcină) pentru care calculeaza, bazându-se pe rezultatele comenzii *ls -l* si pe calculele respective, şi vă afişează în continuare:
    a) spaţiul sumar pe disc al acestui dosar în format convenabil (Gocteţi, Mocteţi, etc.)
    b) spaţiul sumar pe disc al fiecărui dosar care face parte din dosarul rădăcină sortate în ordine descrescândă
    c) În cazul când nu se indică nici un dosar se va utiliza dosarul personal al utilizatorului.

    Testaţi funcţionară scriptului aplicându-l dosarului /var

    Utilizaţi comanda *du* cu opţiunile potrivite (vedeţi *man du* pentru informaţie) pentru a verifica corectitudinea calculelor realizate de scriptul Dumneavoastră.

Conţinutul raportului

Faceţi un raport prezentând mersul exerciţiilor şi prezentaţi-l pentru validare la sfârşitul lucrării de laborator sau în decurs de doua săptămâni după efectuarea lucrării. Includeţi în fiecare răspuns textul scriptului respectiv.

Referinţe

- Pagina de manual bash
- Machtelt Garrels "Bash Guide for Beginners" http://tldp.org/LDP/Bash-Beginners-Guide/html/index.html
- Linux Documentation Project Guides: http://www.tldp.org/guides.html
- Diverse surse pe Internet