MINISTERUL EDUCAȚIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA

Universitatea Tehnică a Moldovei

RAPORT

Lucrare de laborator nr. 5

la cursul ***„Programarea de sistem și rețea”***

A efectuat: **St. gr. CR-221FR Serba Cristina**

A verificat: **conf.univ. Victor Moraru**

Chișinău 2025

Mersul lucrării:

1. Creați un script care afișează numărul de argumente ce i-au fost transmise și le afi șează în continuare unul câte unul. În cazul când nu i-au fost transmise argumente, ea emite mesajul: "Nici un argument transmis". Modificați scriptul pentru a afișa argumentele în ordine inversa

```
#!/bin/bash

if [ $# -eq 0 ]; then
    echo "Nici un argument transmis"
else
    echo "Numărul de argumente: $#"
    for ((i=$#; i>0; i--)); do
        echo "${!i}"
    done
fi
```

2. Analizați scriptul următor:

#!/bin/bash

((test $1 -lt $2) && (echo '$1 < $2')) || (echo '$2 < $1')

Ce face el ? După testarea scriptului înlocuiți ghilimelele simple cu ghilimele duble. Lansați din nou scriptul. Ce ați constatat? Explicați. Studiați codul returnat după executarea acestui script. Explicați. Rescrieți scriptul utilizând structura if then else fi

Răspuns: scriptul compară numeric primii 2 parametri, și afișează exact mesajul $1 < $2 corespunzător dacă al doilea argument e mai mare decât primul, și mesajul $2 < $1 dacă e invers.

Varianta cu if then else if

```
#!/bin/bash

if [ $1 -lt $2 ]
then
    echo "$1 < $2"
else
    echo "$2 < $1"
fi
```

3. Când executați un script, un shell nou demară pentru a executa instrucțiunile conținute în script. Sa testam aceasta afirmație cu un exemplu. Creați un script cu următorul conținut: echo PID-ul meu este $$ Variabila specială $$ ne da PID-ul procesului. Executați scriptul de mai multe ori și
comparați rezultatele. Ce constatați? Vi se pare normal? Explicați.

```
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$ ./script1.sh
PID-ul meu este 10129
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$ ./script1.sh
PID-ul meu este 10130
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$ ./script1.sh
PID-ul meu este 10131
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$ ./script1.sh
PID-ul meu este 10132
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$ ./script1.sh
PID-ul meu este 10133
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$ ./script1.sh
PID-ul meu este 10134
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$ ./script1.sh
PID-ul meu este 10135
kiu@pop-os:~/Documents/uni/ANUL4/sem1/psr/Lab5$
```

Răspuns: La fiecare execuție, scriptul rulează într-un shell nou, care are un alt PID, iar după terminarea execuției scriptului, procesul copil dispare.

1. Modificați scriptul BunaZiua.sh redenumindu-l în BunaZiuaNume.sh după cum
urmează:

a) dacă îi transmiteți doua argumente, primul conținând numele iar al doilea prenumele, el trebuie să le afișeze după salutare (de ex. Buna ziua, Nume Prenume)

b) dacă nu-i transmiteți nici un argument, el trebuie sa va afișeze după salutare numele de conectare al utilizatorului (LOGNAME)

```
#!/bin/bash
if [ $# -eq 2 ]; then
    echo "Buna ziua, $1 $2"
elif [ $# -eq 0 ]; then
    echo "Buna ziua, $LOGNAME"
else
    echo "Utilizare: $0 [Nume Prenume]"
fi
```

2. Realizați un script shell UNIX care permite "curățarea" ecranului iar apoi afi șează structura de fișiere și directoare a directorului pe care l-ați furnizat ca argument. Indicație: comanda care "curăță" ecranul este

clear. Studiați pagina de manual a acestei comenzi ! În cazul când scriptul este lansat fără de argument el va afișa structura de fișiere și directoare a dosarului personal al utilizatorului.

```
#!/bin/bash
clear
if [ $# -eq 0 ]; then
    ls -l ~
else
    ls -l "$1"
fi
```

3. Realizați un script shell UNIX care cere utilizatorului sa introducă doua șiruri de caractere și apoi afișează un mesaj de informare dacă cele doua șiruri sunt sau nu egale. Mai exact, dacă cele doua șiruri de caractere sunt egale se afișează la terminal un mesaj de tipul “Cele doua șiruri de caracter sunt egale”, iar dacă sunt diferite - un mesaj de tipul “Cele doua șiruri de caractere sunt diferite”.

```
#!/bin/bash
echo "Introdu primul sir: "
read s1
echo "Introdu al doilea sir: "
read s2

if [ "$s1" = "$s2" ]; then
    echo "Cele doua siruri de caractere sunt egale"
else
    echo "Cele doua siruri de caractere sunt diferite"
fi
```

4. Realizați un script care primește drept argument calea către un director, verifica dacă acesta este un director și în caz afirmativ afișează conținutul acestuia și al subdirectoarelor care fac parte din el.

```
#!/bin/bash
if [ -d "$1" ]; then
    ls -R "$1"
else
    echo "$1 nu este un director"
fi
```

5. Realizați un script care permite copierea unui director specificat de către utilizator ca argument, cu întreg conținutul sau de fișiere și directoare într-un director numit copie. Dacă fișierul specificat nu exista, scriptul va anunța despre asta și se închide.

```
#!/bin/bash
if [ -d "$1" ]; then
    cp -r "$1" copie
    echo "Directorul $1 a fost copiat in copie/"
else
    echo "Directorul $1 nu exista"
fi
```

6. Realizați un script shell UNIX care calculează suma primelor cinci numere pare, utilizând un ciclu while.

```
#!/bin/bash
i=2
s=0
c=0
while [ $c -lt 5 ]
do
    s=$((s + i))
    i=$((i + 2))
    c=$((c + 1))
done
echo "Suma primelor 5 numere pare este: $s"
```

7. Realizați un script care permite afișarea denumirii unei luni, în situația în care utilizatorul specifica numărul lunii. De exemplu pentru 3 afișează “martie”.

```
#!/bin/bash
case $1 in
    1) echo "ianuarie" ;;
    2) echo "februarie" ;;
    3) echo "martie" ;;
    4) echo "aprilie" ;;
    5) echo "mai" ;;
    6) echo "iunie" ;;
    7) echo "iulie" ;;
    8) echo "august" ;;
```

```
        9) echo "septembrie" ;;
        10) echo "octombrie" ;;
        11) echo "noiembrie" ;;
        12) echo "decembrie" ;;
        *) echo "Numar invalid (1-12)" ;;
    esac
```

8. Analog, creați un script shell UNIX care nu afișează decât subdirectoarele dintr-un anumit director furnizat ca argument.

```
    #!/bin/bash
    if [ -d "$1" ]; then
        for f in "$1"/*; do
            [ -d "$f" ] && echo "$f"
        done
    else
        echo "$1 nu este un director"
    fi
```

9. Realizați un script shell UNIX care permite afișarea numărului de fișiere și de subdirectoare dintr-un director furnizat drept argument (verificați 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)

```
    #!/bin/bash
    if [ -d "$1" ]; then
        echo "Cautare superficiala:"
        echo "Fisiere: $(find "$1" -maxdepth 1 -type f | wc -l)"
        echo "Directoare: $(find "$1" -maxdepth 1 -type d | wc -l)"

        echo "Cautare in profunzime:"
        echo "Fisiere: $(find "$1" -type f | wc -l)"
        echo "Directoare: $(find "$1" -type d | wc -l)"
    else
        echo "$1 nu exista"
    fi
```

10. Scrieți un script care afișează numerele de la 1 până la valoarea N transmisa scriptului în calitate de parametru:

a) Folosiți în acest scop o buclă while do done

b) Rescrieți script-ul folosind structura until do done.

c) Schimbați script-ul pentru a calcula media tuturor valorilor afișate anterior.

```
#!/bin/bash
i=1
sum=0
while [ $i -le $1 ]; do
    echo $i
    sum=$((sum+i))
    i=$((i+1))
done
echo "Media = $((sum/$1))"
```

```
#!/bin/bash
i=1
sum=0
until [ $i -gt $1 ]; do
    echo $i
    sum=$((sum+i))
    i=$((i+1))
done
echo "Media = $((sum/$1))"
```

11. Scrieți 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, și vă afișează în continuare:

a) spațiul sumar pe disc al acestui dosar în format convenabil (Gocteți, Mocteți, etc.)

b) spațiul 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.

Testați funcționară scriptului aplicându-l dosarului /var

```
#!/bin/bash
dir=${1:-$HOME}
```

```
    echo "Spatiul total al directorului $dir:"
    du -sh "$dir"

    echo
    echo "Spatiul fiecarui subdirector sortat descrescator:"
    du -sh "$dir"/* 2>/dev/null | sort -hr
```

Concluzii:

În cadrul acestor exerciții am realizat scripturi shell care folosesc argumente, variabile de mediu, structuri condiționale și bucle pentru a automatiza diferite operații în Linux: afișarea de mesaje personalizate, compararea șirurilor, lucrul cu directoare și fișiere, copierea și numărarea acestora, calculul unor valori numerice sau analiza spațiului pe disc. Activitatea a evidențiat importanța comenzilor de bază (ls, cp, find, du etc.) și modul în care pot fi combinate cu structuri de control pentru a rezolva probleme practice. În concluzie, exercițiile au oferit o imagine generală și aplicată asupra modului de lucru cu scripturi shell, consolidând competențe esențiale pentru administrarea și automatizarea sarcinilor într-un sistem Linux.