Sisteme de operare

Curs 2

mv → move

TEST

→ biletele cu username, parolă etc.

→ EXAM.txt → are cod c

cp EXAM.txt a.c

gcc -Wall -g -o a a.c

valgrind ./a

→ screen

screen -ls → (arată ce sesiuni sunt disponibile)

screen -r [ce pune] (restaurează sesiunea)

→ ESC gg=G

---

0 standard → input stdin

1 → output stdout

2 → error stderr (← perror)

cat ./etc/passwd | ./filter lib

FILE *f = fopen("a.txt"

int d = open("a.txt") (incrementează cu 1 la fiecare fișier descris, pornește de la 3)

$..

vi filter.c

```
#include <unistd.h>
#include <stdio.h>
#include <string.h>
int main(int argc, char **argv){
    if (argc>1 && strcmp("lib", argv[1])==0){
        with_clib();
    else if (argc>1 && strcmp("sys", argv[1])==0){
        with_syscall();
    else {
        fputs("unknown input", stderr);
    return 0;
}

void with_syscall(){
    char s[64];
    int i,k;
    while(1){
        k=read(0,s,64);
        if(k<=0) break;
        for (i=0; i<k; i++){
            if(s[i]
            s[i]='*';
        }
        write(1,s,k);
    }
}

void with_clib(){
    char s[64];
    int i;
    while(fgets(s,64,stdin)!=NULL){
        for(i=0;i<strlen(s);i++){
            if(s[i]!='\n')
            s[i]='*';
        }
    fputs(s,stdout);
    }
}
```

Redirectarea inputului și outputului

cat /etc/passwd > a.txt

./filter < /etc/passwd > a.txt

/dev/null → mereu gol (tot ce e trimis acolo este șters, din el nu iese nimic)

./filter asdf < /etc/passwd 2>/dev/null → supress errors

1>/dev/null 2>&1 → trimite erorile unde se trimite stout

true, false - comenzi

true || echo asdf → nu se rulează pentru că LAZY EVALUATION

false || echo asdf → output = asdf

grep -q ":" /etc/passwd || echo nu sunt egaluri

grep -q "a" /etc/passwd || echo nu sunt 'a'-uri

(cmd || cmd) && (! cmd)

test → comandă

==	!=	-n	-z
equal	not equal	not null	null

-lt	-le	-eq	-ne	-ge	-gt
less than	less or equal	equal	not equal		

-f	-d	-r	-w	-x
exista fisier	exista director	putem citi	putem scrie	putem executa

test 1 -eq 2 || echo false → false

variabile: A=5, B=ion

echo $A $B → 5 ion

echo $A ${B}i → 5 ioni

comandă embedded → comandă între `____` (apostrof invers)

echo `grep "aa" /etc/passwd | wc -l`

dacă folosim " → nu se face substituția

echo "$B are `grep 'ar' /etc/passwd | wc -l` case"
ion are 16 case

echo '$B are `grep 'ar' /etc/passwd | wc -l` case'

....

ls -l → detalii (-la și pentru hidden files)

./a.sh → Permission denied

```
100  110  100
rwx  rwx  rwx
```

permisiuni (~~user~~) | grup | others

chmod 700 a.sh => -rwx------

d → director
- → file
~~etc~~

#!/usr/bin/python !!!

file a.sh
a.sh : ASCII text
./a.sh
- are locase

```
#!/bin/bash
echo "$B are `grep "ar" /etc/passwd | wc -l` case"
```
a.sh

$0 - numele comenzii
$1 - $9 - argumentele
$* sau $@ - toate arg. concatenate cu spațiu
$# - nr. arg.
$? - codul de exit al comenzii precedente

```
#!/bin/bash
echo Comanda: $0
echo Argumente: $1 $2 $3 $4
echo Toate arg.: $@
echo Nr arg: $#
true
echo Exit code $?
true
echo Exit code $?
shift 4
echo Argumentele: $1 $2 $3 $4
```

```
#!/bin/bash
for A in a b c d e f g h; do
    echo $A din for
done
for A in a b c d e f g h
do
    echo $A din for
done
for A in $@
do
    echo $A
done
for A
do
    echo $A
done
```

-daca e folder/ dire \for/ nr /altfel

```
#!/bin/bash
for A in $@
do
    if test -f $A; then
        echo $A e fisier
    elif test -d $A; then
        echo $A e director
    elif echo $A | grep -q "^[0-9]\+$"; then
        echo $A e numar
    else
        echo $A Nu stim
    fi
done
```

```
if cond; then
elif cond; then
else
fi
```

```
while cond; do
done
```

---

citim inf. de la tast. până la stop

```
while true; do
    read x
    echo $x
    if test "$x" == "stop"; then
        break
    fi
done
```

---

găsim toate fișierele mai mari de un nr. de octeți

```
#!/bin/bash
for F in $(find . -type f); do
    S=$(ls -l $F | awk '{print $5}')
    if test $S -gt 1024000; then
        ls -l $F
    fi
done
```