#lec2

- cli → command line interface, it is the command that takes arguments & options.
- Some commands in OS are found in bash scripts, and some are built in.
- Some arguments are not obligatory, they are sent with $@

$0 → the script name
$1 → first argum.
$2 → 2nd argum. } there is upto $9 arguments.

→ ./mycli ahmed hamdy
($0 = ./mycli, $1 = ahmed, $2 = hamdy)

vim →

```
echo "the script name $0"
echo "The 1st arg $1"
 "     "    "    "  $2
 "     "    "    "  $3   → will print blank space [as no arg 3]
 "     "    "    "  $@   → will print all arguments.
 "     "    "    "  $#   → print the number of arguments
```

```
for filename in ${@}
do
    echo $filename
done
```

→ $1 $2 $3 دي لو وكتها آكش عملت
وبعمل loop على element element فيهم

→ Switch case

```
myoption = $1      → 1st arg
my filename = $2   → 2nd arg

case $myoption in
"L")  we can put it or not.
      ls -l myfilename
      ;;  → break

"a")
      ls -a myfilename
      ;;

*)    → default behaviour.
      ls myfilename

esac
```

→ ./myscript a .
↗ to list current directory

→ ./myscript l Hello

→ ./myscript . → هنا انا بيعتبر
→ ./myscript d . ال . هو ال arg
فيعمل ls
لل current dir
لأن لو جات ls
لوحده هيقراها
ls → ls .
عشان انا مش باخد
اسم/ file
ls ▽

(d →) ده هيكون ال default

→ function

```
myhello_func() {
      echo "Hello"
}

myhello_func → func call
```

if i want to send parameters

```
myhello_func "ahmed" "hamdy"
   ↳ func call

myhello_func() {
    echo "$"
}
```

↳ will print 1st arg → ahmed

my hello_func $1 → terminal ... param

my hello_func $0 → This will print script name.

- declaration of any variable is global by default

if i want to make local scope through func scope:

```
filename = /etc/passwd
my func () {
    local filename = "/etc/shadow"
        echo $filename
}
my func
echo $filename ①
```

* local ... scope ... func ... filename ... /etc/sh ... /etc/pass ... $filename ... local ... /etc/shadow ... 1 ... override

Meta characters (Regex)

* → zero or more

? → zero or one = {0,1} = {,1}

. → one character

+ → at least one or more = {1,}

{N,M} → ex {2,4} from 2 → 4 charac.

{N,} → 2 → ∞

{,M} ex→ {,5} from zero to 5

{M} → a{3} → a is found 3 times.

[a-z] → from a to z

^ → begin of the line

^$ → empty line.

[^a-d] → neglect anything inside this range allowed

/<cat/> → لو عايزين بداية كلمة cat في ال line او بعد ال space وده اسم cat بس

/<cat → تنفع لو مثلاً categories

Example

→ categories bcat catcat 2cat2

/<cat → categories (only) catcat

cat/> → bcat catcat ≅ *cat/>

/<cat/> → nothing will appear.

↳ ≡

▽cat▽
^cat▽
▽cat$
^cat$

} دول حالة خاصة من /<cat/>

grep → is used for pattern matching.

grep -o root /etc/passwd

↳ بيطبع كل ال suggest. اللي كلمة root طالعت فيها، وكل مرة بتطبع in newline

-c → بيطبع ظهرت في كام line

-n → ايه ال lines اللي طلعت فيها كلمة root

grep -i ps. /etc/passwd
↳ case insensitive

grep -v history → neglect o/p that has word history.

grep -w cat ≡ grep /<cat/>
↳ word

grep [yf] /etc/passwd → f,y او كلمة فيها f,y كده يطلع

to use regex, put inside " _ "

character classes

[[:alpha:]] = [a-z]

[[:punct:]]

[[:digit:]]

[[:word:]] → words يطلع

grep "a---d" /etc/passwd → كده يطلع الكلمة اللي فيها اول حرف a واخر حرف d

grep "*" يطلع * و backslash بتمنع ال def action بتاع ال * اي حاجة فيها

grep '\$' ① must be single with $ quote