Shell script part 2

1- The $$ variable:

- يحتوي على قيمة pid للبروسيس process الي شغالة حاليا :

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ps -ef
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 14:15 ?        00:00:00 /init
root        14     1  0 14:15 tty1     00:00:00 /init
ahed        15    14  0 14:15 tty1     00:00:00 -bash
ahed        58    15  0 14:17 tty1     00:00:00 ps -ef
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $$
15
```

- تستخدم مع kill اذا بدي انهي ال procees الشغالة
- وتستخدم ايضا مع multiprocessing يعني اذا كل بروسيس بدها تنشأ فايل خاص فيها منقدر نميز كل فايل ب pid الي بقدر اجيبه من خلال $$

2- test command:

- يستخدم لفحص قيمة او اي شرط
- كيف بستخدمها ؟ كالتالي test expression او [expression]
  - **<u>test string :</u>**

```
ahed@DESKTOP-OK5G6FV:~$ my_name="ahed"
ahed@DESKTOP-OK5G6FV:~$ test "$my_name" = "ahed"
ahed@DESKTOP-OK5G6FV:~$ echo $?
0
ahed@DESKTOP-OK5G6FV:~$ test "$my_name" = "sameh"
ahed@DESKTOP-OK5G6FV:~$ echo $?
1
ahed@DESKTOP-OK5G6FV:~$
```

في هذا المثال حطينا في المتغير my_name قيمة معينة "ahed"
وقارنت قيمة هاد المتغير ب ahed من خلال test، في هاي الحالة

القيمة الي في my_name هي نفس القيمة الي قارنت فيها ف لازم يعطيني true
عشان اعرف النتيجة لازم اطبع exit status الي $? وكما هو واضح طبعت 0 يعني true
وفي المقارنة الثانية قارن قيمة المتغير "ahed" ب قيمة "sameh" وطبعا هدول مش متساويات فلازم يعطيني false
كمان مرة بطبع exit status و طبعت 1 يعني false (اي رقم موجب في exit status يمثل false)
مهمممم : لازم احط space قبل وبعد =
مهمممم: الافضل نحط double quotation على الاشياء الي بقارنها الي هم قبل وبعد = ، مثلا حسب المثال حطينا my_name و القيمة الي بعد = في "" —-> "$my_name"

- String operation :

operation	value
string1 = string2	0 if two strings are equal, else return 1
string1 != string2	0 if two strings are not equal, else retu
-n string	0 if string is not empty, else return 1
-z string	0 if string is empty, else return 1

```
ahed@DESKTOP-OK5G6FV: ~/ahed
ahed@DESKTOP-OK5G6FV:~/ahed$
ahed@DESKTOP-OK5G6FV:~/ahed$ name="ahed"
ahed@DESKTOP-OK5G6FV:~/ahed$ [ -n "$name" ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
0
ahed@DESKTOP-OK5G6FV:~/ahed$ [ -z "$name" ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
1
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- **Test integers :**
  - تستخدم لعمل مقارانات على الارقام الصحيحة

rator	Returns TRUE (exit status of 0) if
$_1$ -eq int_2	int_1 is equal to int_2
$_1$ -ge int_2	int_1 is greater than or equal to int_2
$_1$ -gt int_2	int_1 is greater than int_2
$_1$ -le int_2	int_1 is less than or equal to int_2
$_1$ -lt int_2	int_1 is less than int_2
$_1$ -ne int_2	int_1 is not equal to int_2

```
ahed@DESKTOP-OK5G6FV:~/ahed$ num=3
ahed@DESKTOP-OK5G6FV:~/ahed$ [ "$num" -eq 3 ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
0
ahed@DESKTOP-OK5G6FV:~/ahed$ [ "$num" -le 10 ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
0
ahed@DESKTOP-OK5G6FV:~/ahed$ [ "$num" -ge 10 ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
1
ahed@DESKTOP-OK5G6FV:~/ahed$
```

Q) what is the output of the following:

```
x1="005"
x2=" 10"
[ "$x1" = 5 ]
echo $?
[ "$x1" -eq 5 ]
echo $?
[ "$x2" = 10 ]
echo $?
[ "$x2" -eq 10 ]
echo $?
```

- **test files:**
- تستخدم لفحص الملفات مثلا هل هذا الملف ملف عادي وانه مش مجلد او هل هذا الملف موجود او هل يملك صلاحية القراءة او هل حجمه صفر ..... الخ.

Operator	Returns TRUE (exit status of 0) if
`-d file`	`file` is a directory
`-e file`	`file` exists
`-f file`	`file` is an ordinary file
`-r file`	`file` is readable by the process
`-s file`	`file` has nonzero length
`-w file`	`file` is writable by the process
`-x file`	`file` is executable
`-L file`	`file` is a symbolic link

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ls
exp3.txt  file.txtx  files_name.txt  hard  list.sh  main_dir  msg.txt  name
ahed@DESKTOP-OK5G6FV:~/ahed$ [ -d msg.txt ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
1
ahed@DESKTOP-OK5G6FV:~/ahed$ [ -d main_dir ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
0
ahed@DESKTOP-OK5G6FV:~/ahed$ [ -e files_name.txt ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
0
ahed@DESKTOP-OK5G6FV:~/ahed$ [ ! -e files_name.txt ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
1
ahed@DESKTOP-OK5G6FV:~/ahed$
```

note) ! means not

3- The Logical AND Operator -a:

- منستخدمها لما بدنا نفحص اكثر من شرط واحد
- مثلا بدنا نفصحص قيمة رقم معين هل هو بين 0 و 10 يعني اكبر من 0 واقل من 10 :

```
ahed@DESKTOP-OK5G6FV:~/ahed$ num=5
ahed@DESKTOP-OK5G6FV:~/ahed$ [ "$num" -ge 0 -a "$num" -lt 10  ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
0
ahed@DESKTOP-OK5G6FV:~/ahed$ [ "$num" -ge 8 -a "$num" -lt 10  ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
1
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- Q) What is another way to do the same thing????

4- The Logical OR Operator -o:

```
ahed@DESKTOP-OK5G6FV:~/ahed$ num=5
ahed@DESKTOP-OK5G6FV:~/ahed$ [ "$num" -ge 8 -o "$num" -lt 10  ]
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
0
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- اذا استخدمنا a- و o- مع بعض تكون الاولوية ل a-
- اذا اردنا تغير الاولوية نستخد (\ \) :

```
P-OK5G6FV:~/ahed$ [ "$num" -ge 4 -o "$num" -lt 10 -a "$num" -eq 0 ]
P-OK5G6FV:~/ahed$ echo $?

P-OK5G6FV:~/ahed$ [ \( "$num" -ge 4 -o "$num" -lt 10 \) -a "$num" -eq 0 ]
P-OK5G6FV:~/ahed$ echo $?
```

Q) explain the output (why in the first case the output is 0 and in the second case the output is 1)?

5- if else condition:

- تستخدم لعمل شيء معين اذا تحقق شرط معين
- Syntax :

```
if commandt
then
      command
      command
      ...
else
      command
      command
      ...
fi
```

- مهمممممممم: ما تنسى fi ، اذا ما حطيتها رح يعطيك ايرور EOF
- نشوف مثال، خلينا نكتب shell script بتوخذ رقم ك argument وبتفحص هل الرقم اكبر من 5 :

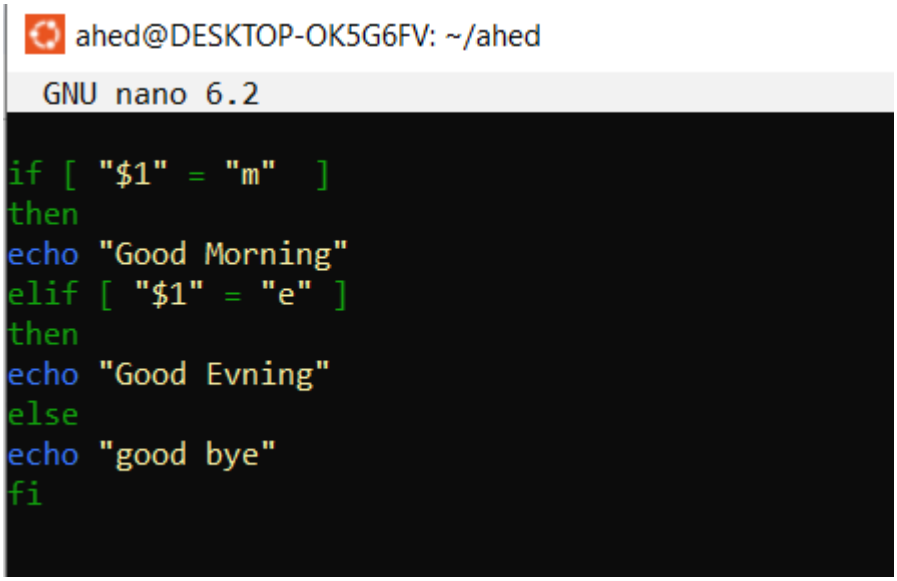
```
ahed@DESKTOP-OK5G6FV: ~/ahed
GNU nano 6.2

if [ "$1" -gt 5 ]
then
    echo "$1 is greater than 5"
else
echo "$1 is less than 5"
fi
```

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./isdigit.sh 2
2 is less than 5
ahed@DESKTOP-OK5G6FV:~/ahed$ ./isdigit.sh 6
6 is greater than 5
ahed@DESKTOP-OK5G6FV:~/ahed$
```

6- if elif :

```
# Start the if block
if [ CONDITION ]; then
   # Commands to execute if CONDITION is true
   COMMANDS
# Optionally, start the elif block
elif [ ANOTHER_CONDITION ]; then
   # Commands to execute if ANOTHER_CONDITION is true
   MORE_COMMANDS
# Optionally, start the else block
else
   # Commands to execute if none of the above conditions are true
   OTHER_COMMANDS
fi
```

- نفس if else if في لغة السي
- خلينا نكتب shell script بتوخذ argument ، اذا كانت arg هي حرف m رح يطبع good morning
- اذا كانت arg هي حرف e رح يطبع good evning
- اي شي ثاني يطبع good bye



```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh m
Good Morning
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh e
Good Evning
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh g
good bye
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh g
good bye
ahed@DESKTOP-OK5G6FV:~/ahed$
```

7- exit n : n is any integer number

- تستخدم للخروج من التيرمنال او من shell script ويمكن تحديد رقم معها ويكون ك exit status ($?)

```
ahed@DESKTOP-OK5G6FV: ~/ahed
GNU nano 6.2

exit 5
```

هسا لو اطبع exit status بعد تشغيل shell script رح يطبعلي 5

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
5
ahed@DESKTOP-OK5G6FV:~/ahed$
```

Q1) write a shell script that takes two numbers as arguments, and check if there are two arguments then print the sum of them else exit from the program.
Q2) write a shell script that takes a string as an argument and print its length. Also check if there is only one argument.
Q3) write a shell script that prints “good morning” if the current hour is less than 12 pm, otherwise print “good evening”.
Q4) write a shell script that check if the specific user is login to the system or not, you must pass the name of the user as argument.

8- case (switch):

- Syntax :

```
case "$variable" in
   pattern1)
      # Commands to execute if $variable matches pattern1
      ;;
   pattern2)
      # Commands to execute if $variable matches pattern2
      ;;
   *)
      # Commands to execute if $variable does not match any pa
      ;;
esac
```

- مهمممممم : ما تنسى esac
- مثال: خلينا نكتب كود بفحص اذا الرقم المدخل ك arg هو بين 1 و 3 واذا دخل اي شي ثاني يطبعله انه لازم يدخل من 1 ل 3:

```
ahed@DESKTOP-OK5G6FV: ~/ahed
NU nano 6.2                                                    test.sh


valuate the input using case statement
e "$1" in
 1)
     echo "You selected option 1."
     ;;
 2)
     echo "You selected option 2."
     ;;
 3)
     echo "You selected option 3."
     ;;
 *)
     echo "Invalid option. Please select a number between 1 and 3."
     ;;
c
```

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 1
You selected option 1.
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 2
You selected option 2.
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 3
You selected option 3.
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 4
Invalid option. Please select a number between 1 and 3.
```

- مهمممممممم: ما تحطو defualt case (*)) اول شي او ثاني ، لازم تكون اخر case عشان يعطيكم حل صح

```
num=$1
case $num in
1) echo "the number is 1" ;;

*) echo "the input muse be between 1 - 4";;

2) echo "the number is 2" ;;

3) echo "the number is 3" ;;

4) echo "the number is 4" ;;


esac
```

- زي الكود الي مكتوب بالصورة غلط يكون هيك لانه رح يعطينا logic غلط

Q) write a shell script that takes a character as an argument, then checks if the character is a digit or small letter or capital letter, otherwise it will print a special character.

9- The Null Command (:) :

- تستخدم لما ما بدي اعمل شي اذا تحقق شرط معين عشان ما اخلي البلوك تاع if او else او حتى loop فاضي
- مثلا بدي افحص اذا arg هي 1 ما يعمل شي غير هيك يطبع thank you:

```
Select ahed@DESKTOP-OK5G6FV: ~/ahed
GNU nano 6.2

if [ "$1" -eq 1 ]
then
   :
else
   echo "thank you :)"
fi
```

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 1
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 2
thank you :)
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 3
thank you :)
ahed@DESKTOP-OK5G6FV:~/ahed$ ./test.sh 45
thank you :)
ahed@DESKTOP-OK5G6FV:~/ahed$
```

10- The && and || Constructs:

- Command1 && command2: معناها اذا كوماند1 تم بنجاح، نفذ كوماند 2 غير هيك ما تنفذ كوماند 2

```
ahed@DESKTOP-OK5G6FV:~/ahed$ echo "hello" && echo "hello" | wc -c
hello
6
ahed@DESKTOP-OK5G6FV:~/ahed$ ls
exp3.txt   files_name.txt  isdigit.sh  main_dir  names.txt  soft     test.t
file.txtx  hard            list.sh     msg.txt   new.txt    test.sh
ahed@DESKTOP-OK5G6FV:~/ahed$ rm asdasdasd && echo "hi"
rm: cannot remove 'asdasdasd': No such file or directory
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- Command1 || command2: نفذ الكوماند 1، إذا ما تم تنفيذ الكوماند 2

```
ahed@DESKTOP-OK5G6FV:~/ahed$ rm asdasdasd || echo "hi"
rm: cannot remove 'asdasdasd': No such file or directory
hi
```

```
ahed@DESKTOP-OK5G6FV:~/ahed$ touch testfile.txt || echo "ok"
ahed@DESKTOP-OK5G6FV:~/ahed$
```

11- Debugging shell scripts with the-x option:

- عشان نعمل debug ل shell script نعمل الكوماند التالي : **Sh -x ./file_name.sh**

```
ahed@DESKTOP-OK5G6FV:~/ahed$ cat test.sh


if [ "$1" -eq 1 ]
then
   :
else
   echo "thank you :)"
fi
ahed@DESKTOP-OK5G6FV:~/ahed$ sh -x ./test.sh 1
+ [ 1 -eq 1 ]
+ :
ahed@DESKTOP-OK5G6FV:~/ahed$ sh -x ./test.sh 2
+ [ 2 -eq 1 ]
+ echo thank you :)
thank you :)
ahed@DESKTOP-OK5G6FV:~/ahed$
```

Tasks:)

1. Write a shell script (calc.sh) that takes 3 args: first and third one are number and the second is operator (+,-,/,*), then do the suitable calculation based on args.

   Exp: ./calc.sh 5 + 6 → output will be "sum of 5 and 6 is 11"

   ./calc.sh 5*6 → output will be "mul of 5 and 6 is 30"

2. Write a shell script that checks take one argument and check if the argument is string (just letter).
3. Write a shell script that check the memory space:

- **If the free memory space is greater than the used memory space, print:**
  "Free space available: [value of free space]"
- **If the free memory space equals the used memory space, print:**
  "No available free space"
- Note: you can free -m command to display the informations about your memory