Shell scripts

- هيك منكون خلصنا اول 3 مختبرات (33.3% من المختبر-الجزء الاول) والي تعلمنا فيهم بعض الكوماندز الكثير مهمة ومستخدمة في اللينوكس.
- وحاليا رح ندخل بالجزء الثاني من المختبر والي هو عبارة أيضا عن 3 تجارب او مختبرات والي رح نحكي فيهم عن shell script
- الجزء الثاني من المختبر يعتمد كثيييير!!!!!! على الجزء الاول

- ايش هو shell script ؟

  في البداية حكينا في الجزء الاول عن الكوماندز وكنا نكتب هاي الكوماندز في التيرمنال عشان ننفذ شيء معين، طيب ايش فهم النظام (kernel) ايش هاي الكوماندز وشو بتعمل.

  المسؤول عن هاي الوظيفة برنامج اسمه shell، هاد shell بوخذ الكوماند الي بكتبه ال user وبحوله لاشي بفهمه kernel ليتم تنفيذه او يعني تنفيذ service معينة في system.

  طيب لفرضا بدك تعمل مجموعة كوماندز معينه عشان تبحث عن شيء معين او تحصل على داتا معينه او انك تعدل على داتا او تفحص هل الداتا صحيحة او بدك توصل لمحتوي ملف سطر سطر ، ف ببطل انه نعتمد ع التيرمنال اشي عملي .

  هل منطقي وعملي في كل مرة بدك تعمل نفس الكوماندز وتكررهم؟ اكيد لأ

  طيب لو كتبنا هاي الكوماندز على فايل معين وصرنا بس ننفذ هذا الفايل او الملف، برأيك بتكون الامور اسهل و وفرت وقت وجهد ؟ اكيد اه

  طب هل هذا الاشي موجود؟ اكيد وهو هذا الي منحكيله shell script

- طبعا هاد الملف لازم يكون shell file يعني الامتداد تاعه يكون **sh.**
- في انواع من shell اشهرها Bourne shell والي منحددها ب (sh) وفي كمان bash والي منحددها ب (bash) وفي كمان csh ، طبعا الفرق بينهم في syntax
- طبعا bash هي زي اضافة على sh عشان هيك فيها syntax بكون اسهل من sh ، يعني بشبهو بعض كثير
- طبعا shell script بقدر اكتب فيها كوماندز وجمل شرطية (if-else) وحلقات تكرار (loops) ف هي ك لغات البرمجة الاخرى التي تعمل ب interpreter
- Interpreter: انه بقرأ الكود وبنفذه سطر سطر

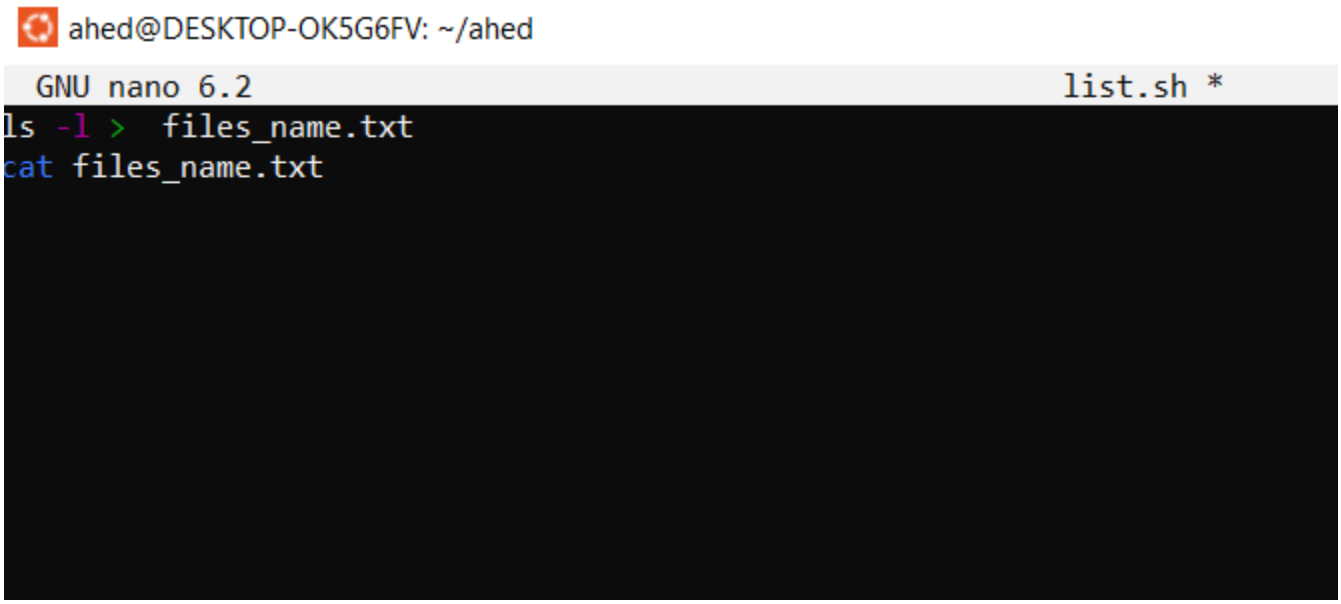
- خلينا نبلش ونتعرف على syntax تاعها .

1- introduction

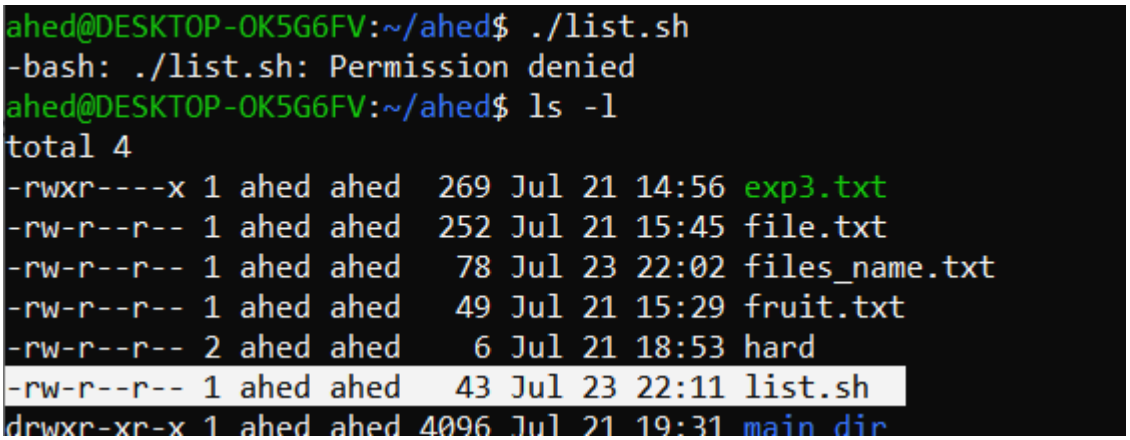
- طيب خلينا نكتب كوماند بكتب اسماء الملفات والمجلدات في المسار الحالي على فايل وبعدين نطبع محتوى الفايل:

```
hed@DESKTOP-OK5G6FV:~/ahed$ ls
xp3.txt  file.txt  fruit.txt  hard  main_dir  msg.txt  new.txt  soft
hed@DESKTOP-OK5G6FV:~/ahed$ ls > files_name.txt
hed@DESKTOP-OK5G6FV:~/ahed$ cat files_name.txt
xp3.txt
ile.txt
iles_name.txt
ruit.txt
ard
ain_dir
sg.txt
ew.txt
oft
hed@DESKTOP-OK5G6FV:~/ahed$
```

- نفرض انه زاد عدد الفايلات، ف منحتاج نرجع نعمل الكوماندز كمان مرة وطبعا هيك بصير الموضوع ممل ومرهق ومش عملي
- ف الحل زي محكينا قبل انه نكتب هاي الكوماندز على فايل من نوع sh ونصير نشغل الفايل بس (shell script)
- خلينا ننشأ فايل نسميه ، **list.sh** طبعا منشأ الفايل عادي باستخدام touch list.sh
- بعدين مندخل على الفايل من خلال ال nano او اي text editor
- ومنكتب الكوماندز الي بدنا اياهم



- بعدين منحفظ الفايل ومنطلع منه عشان ننفذه
- انتبهو انه الفايل الي فيه الكوماندز (list.sh) بختلف عن الفايل الي مكتوب فيه اسماء الملفات (files_name.txt)
- عشان ننفذ الفايل نكتب هيك ./file_name.sh ف في مثالنا رح نكتب ./list.sh
- شو معنى /. ؟؟
- وفي طريقة ثانية عشان نشغل الملف انه نكتب sh file_name.sh



- اذا لاحظتو ما نفذ الفايل لانه ما في صلاحية excute الي بكون رمزها (x)
- فعشان نعطي هاي الصلاحية اما نستخدم طريقة التجربة الاولى يعني منعبر عن الصلاحية ب الارقام او منحط +x وهاي معناها اعطي صلاحية التنفيذ (x)

```
ahed@DESKTOP-OK5G6FV:~/ahed$ chmod +x list.sh
ahed@DESKTOP-OK5G6FV:~/ahed$ ./list.sh
total 4
-rwxr----x 1 ahed ahed  269 Jul 21 14:56 exp3.txt
-rw-r--r-- 1 ahed ahed  252 Jul 21 15:45 file.txt
-rw-r--r-- 1 ahed ahed    0 Jul 23 22:15 files_name.txt
-rw-r--r-- 1 ahed ahed   49 Jul 21 15:29 fruit.txt
-rw-r--r-- 2 ahed ahed    6 Jul 21 18:53 hard
-rwxr-xr-x 1 ahed ahed   43 Jul 23 22:11 list.sh
drwxr-xr-x 1 ahed ahed 4096 Jul 21 19:31 main_dir
-rw-r--r-- 2 ahed ahed    6 Jul 21 18:53 msg.txt
-rw-r--r-- 1 ahed ahed 1629 Jul 23 13:38 new.txt
lrwxrwxrwx 1 ahed ahed    7 Jul 21 18:53 soft -> msg.txt
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- وهيك بكون نفذ الفايل وطبع نتيجة الكوماندز، ف هاي الطريقة اسرع واسهل
- وهيك مكون كتبنا shell script بتطبع اسماء الفايلات على فايل وبطتبع محتوى الفايل.
- طبعا ممكن نستخدم echo عشان نعمل اشي تجميلي:

Select ahed@DESKTOP-OK5G6FV: ~/ahed

```
 GNU nano 6.2                                                list.sh
ls -l >  files_name.txt
echo "the name of files and directories:"
echo "---------------------------------"
cat files_name.txt
```

- وهيك بتكون النتيجة :

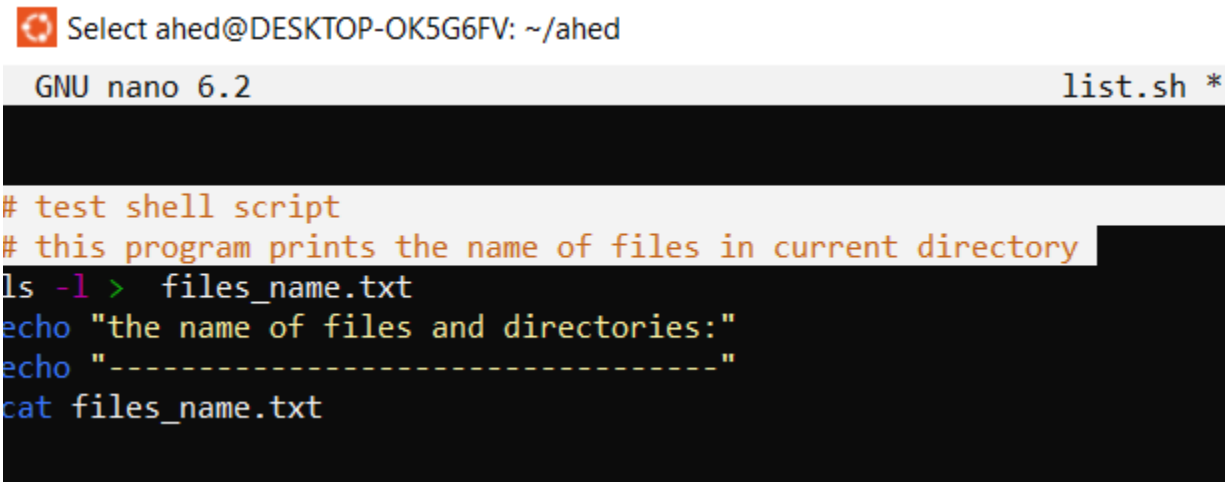
```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./list.sh
the name of files and directories:
---------------------------------
total 8
-rwxr----x 1 ahed ahed  269 Jul 21 14:56 exp3.txt
-rw-r--r-- 1 ahed ahed  252 Jul 21 15:45 file.txt
-rw-r--r-- 1 ahed ahed    0 Jul 23 22:26 files_name.txt
-rw-r--r-- 1 ahed ahed   49 Jul 21 15:29 fruit.txt
-rw-r--r-- 2 ahed ahed    6 Jul 21 18:53 hard
-rwxr-xr-x 1 ahed ahed  127 Jul 23 22:26 list.sh
drwxr-xr-x 1 ahed ahed 4096 Jul 21 19:31 main_dir
-rw-r--r-- 2 ahed ahed    6 Jul 21 18:53 msg.txt
-rw-r--r-- 1 ahed ahed 1629 Jul 23 13:38 new.txt
lrwxrwxrwx 1 ahed ahed    7 Jul 21 18:53 soft -> msg.txt
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- مثل ما انتو ملاحظين منقدر نستخدم اي كوماند تعلمناه قبل

Q1) write a shell script that prints the current date and time.
Q2) write a shell script that counts the number of characters in the “hello world” string.

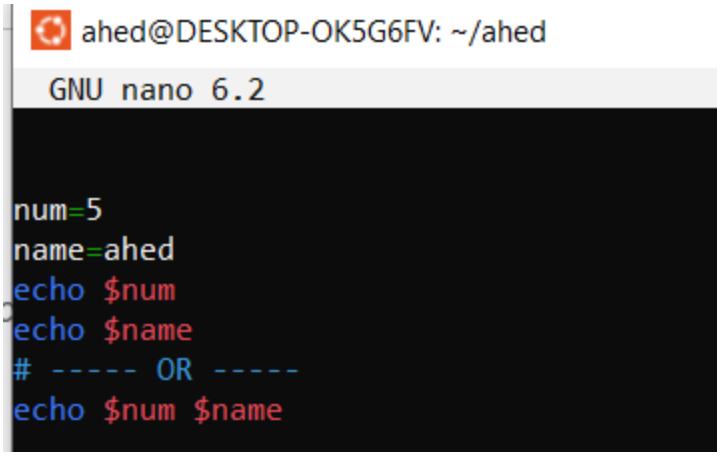
2- Comments:

- ما هو comments وما اهميته بالكود؟
- عشان اقدر اعمل comments في shell script بستخدم (#):



3- Variables المتغيرات

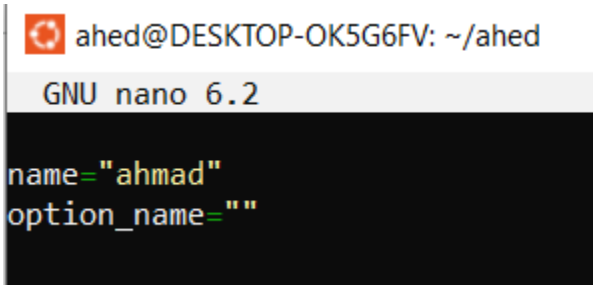
- ما اهمية variables ؟
- لتعريف variable نستخدم syntax التالي : variable_name=value
- انتبهو انه ما في space قبل وبعد =
- لا يوجد data type مثلا (...,int,string,char)
- في shell قيمة variable بتكون دائما عبارة عن مجموعة chat او string حتى لو دخلتو ارقام، طبعا يوجد طريقة معينه للتعامل مع الارقام.
- لنطبع قيمة variable اكيد منستخدم echo ومنحط مع اسم المتغير اشارة $:



- والنتيجة كالتالي :

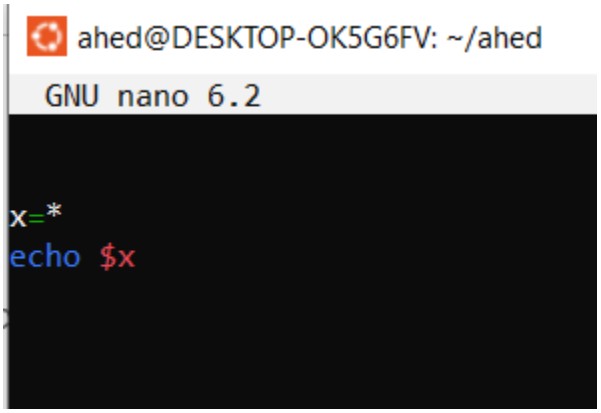
```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./list.sh
5
ahed
5 ahed
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- الافضل انه نحط قيمة variable في " "" single or double quotation
  → name=”ahed”
- طبعا ممكن اعرف variable ويكون ما في له قيمة :



4- Filename Substitution and Variables (*):

- (*) معناها كل الملفات والمجلدات :



- والنتيجة كالتالي:

```
K5G6FV:~/ahed$ ./list.sh
 txt files_name.txt fruit.txt hard list.sh main_dir msg.txt new.txt soft
K5G6FV:~/ahed$
```

5- The ${variable} Construct:

- منستخدمها اذا بدي اعدل على اسم الفايل مثلا واضيف حرف معين على اخر اسمه:

```
OK5G6FV:~/ahed$ ls
e.txt  files_name.txt  hard  list.sh  main_dir  msg.txt  new.txt  soft
OK5G6FV:~/ahed$ f="file.txt"
OK5G6FV:~/ahed$ mv $f ${f}x
OK5G6FV:~/ahed$ ls
e.txtx  files_name.txt  hard  list.sh  main_dir  msg.txt  new.txt  soft
OK5G6FV:~/ahed$
```

- ملاحظة : طبقت المفهوم على التيرمنال لتسريع عملية الشرح فقط، الافضل ان تكتبها داخل shell script
- في الامثلة الجاية رح تشوفو اني بكتب الكوماندز على التيرمنال مش ع فايل بس لتسريع عملية الشرح.

6- Built-in Integer Arithmetic:

- في هذا الجزء رح نتلعم كيف نتعامل مع الارقام ونعمل العمليات الحسابية والمطنقية
- في syntax 2 منقدر نتعامل فيهم مع الارقام والعلميات على الارقام:

((expression))

```
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $(( 5+5   ))
10
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $(( 5 + 5   ))
10
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $((5 + 5))
10
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $((5+5))
10
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- طبعا ممكن استخدم في exprission متغيرات و كمان ممكن اخزن النتيجة في متغير:

```
ahed@DESKTOP-OK5G6FV:~/ahed$ num1=5
ahed@DESKTOP-OK5G6FV:~/ahed$ res=$(( $num1 + 2  ))
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $res
7
ahed@DESKTOP-OK5G6FV:~/ahed$ res=$(( num1 + 2  ))
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $res
7
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- عشان اعمل access على اي variable بستخدم $، بزبط اذا ما استخدمناها لكن الافضل والشكل العملي اكثر انه نستخدم $
- في عملية الطباعة لل var يجب استخدام **$**
- ويمكن استخدام نفس الطريقة للعمليات المنطقية:

```
ahed@DESKTOP-OK5G6FV:~/ahed$ num=6
ahed@DESKTOP-OK5G6FV:~/ahed$ res=$(( $num>=2 && $num<=100    ))
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $res
1
ahed@DESKTOP-OK5G6FV:~/ahed$ res=$(( $num>=8 && $num<=100    ))
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $res
0
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- 1 (True), 0 (False)

expr expression

```
ahed@DESKTOP-OK5G6FV:~/ahed$ expr 1+2+3
1+2+3
ahed@DESKTOP-OK5G6FV:~/ahed$ expr 1 + 2 + 3
6
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- مهمممم جداا يكون في فراغ بين الارقام والعمليات
- expr بتطبع الناتج على التيرمنال بدون استخدام echo
- طبعا مسنتخدم نفس الطريقة مع باقي العمليات ك الطرح (-) والقسمة (/) لكن في الضرب نستخدم هذا (*\) لانه * لوحدها معناها كل الفايلات ف عشان نفقدها معناها منحط \ (escape)

- طبعا exp لا تقبل () لعمل اولوية لعملية حسابية الا اذا وضعتهم في '(' or ')' :

```
ahed@DESKTOP-OK5G6FV:~/ahed$ expr ( 6 + 6 ) / 2
-bash: syntax error near unexpected token `6'
ahed@DESKTOP-OK5G6FV:~/ahed$ expr '(' 6 + 6 ')' / 2
6
```

- عشان هيك الافضل انه نستخد syntax الاول الي هو$((expression))

```
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $((    ( 6+6 ) / 2     ))
6
ahed@DESKTOP-OK5G6FV:~/ahed$
```

Q) calculat the value of 5*10 / 6+2

7- single quotes:

- بتعمل على تجاهل كل chars واعتبارهم ك string فمثلا اذا عملنا * داخل '*' بتكون معناها char * وليس كل الملفات كما في var ايضا:

```
ahed@DESKTOP-OK5G6FV:~/ahed$ x=*
ahed@DESKTOP-OK5G6FV:~/ahed$ echo '$x'
$x
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- كما واضح في المثال " تجاهلت معنى * و $
- احد استخدماتها عشان اربط الاسماء الي فيها space، مثلا بدي ابحث عن ahed mafarjeh ف لازم احط الاسم في 'ahed mafajreh' عشان يعتبره ك اسم واحد مش اسمين وهاي منستخدمها لما بدنا نبحث عن اسماء:

```
ahed@DESKTOP-OK5G6FV:~/ahed$ cat names.txt
ahmad
ahed mafarjeh
amal jamal
mohammad
ahed@DESKTOP-OK5G6FV:~/ahed$ grep ahed mafarjeh names.txt
grep: mafarjeh: No such file or directory
names.txt:ahed mafarjeh
ahed@DESKTOP-OK5G6FV:~/ahed$ grep 'ahed mafarjeh' names.txt
ahed mafarjeh
ahed@DESKTOP-OK5G6FV:~/ahed$
```

8- Double quotes “”:

- بتعمل على تجاهل كل chars واعتبارهم ك string ما عدا التالية :

($) dollar sign

```
ahed@DESKTOP-OK5G6FV:~/ahed$ x=*
ahed@DESKTOP-OK5G6FV:~/ahed$ echo "$x"
*
ahed@DESKTOP-OK5G6FV:~/ahed$
```

(`) back quotes → حرف ذ

Backslash (\)

9- Backslash (\) : \ = ‘’

- لما تستخدمها بتصير كأنك حاطت " single quotation

```
ahed@DESKTOP-OK5G6FV:~/ahed$ echo \$x
$x
ahed@DESKTOP-OK5G6FV:~/ahed$
```

Q) what is the output of the following:

```
x=*
echo \$x
echo '\$x'
echo "\$x"
echo \$x
echo \\
```

10- Command Substitution:

- بستخدمها عشان انفذ داخل string في قيم المتغيرات و داخل double quotation
  - Why not in single quotation?
- في لها 2 syntax : يا اما $(command) او `commands` الي هي back quote

```
ahed@DESKTOP-OK5G6FV:~/ahed$ echo "the cuttent time and date is $(date)"
the cuttent time and date is Wed Jul 24 00:00:54 IDT 2024
ahed@DESKTOP-OK5G6FV:~/ahed$ echo "the current time and date is `date`"
the current time and date is Wed Jul 24 00:04:43 IDT 2024
ahed@DESKTOP-OK5G6FV:~/ahed$
```

```
ahed@DESKTOP-OK5G6FV:~/ahed$ t=date
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $t
date
ahed@DESKTOP-OK5G6FV:~/ahed$ t=$(date)
ahed@DESKTOP-OK5G6FV:~/ahed$ echo "the current date and time is   $t"
the current date and time is   Wed Jul 24 00:06:19 IDT 2024
ahed@DESKTOP-OK5G6FV:~/ahed$
```

Q) get the value of current time only and save it in variables, then print the value of the variable.

11- Passing Arguments to Shell Scripts:

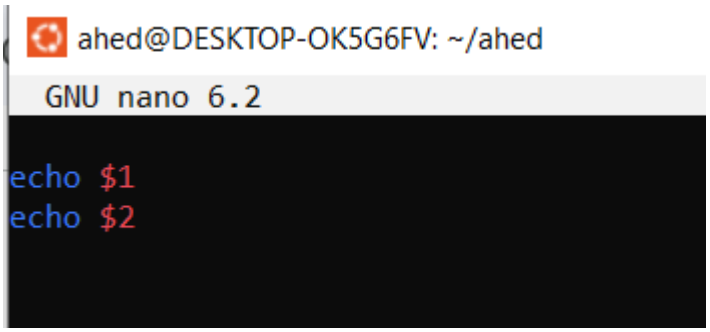
- بتقدر تبعت او تمرر arguments ل shell script من خلال التيرمنال
- يمكنك اضافة arguments بعد اسم الفايل هكذا :

./file_name.sh arg1 arg2 arg3 arg4 …… argn

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./list.sh ahed 5
```

- يمكنك ان تستقبل قيم هذه arguments في متغيرات خاصة يكون اسمها كالتالي:

  $1 (arg1), $2(arg2),..... $n(argn)



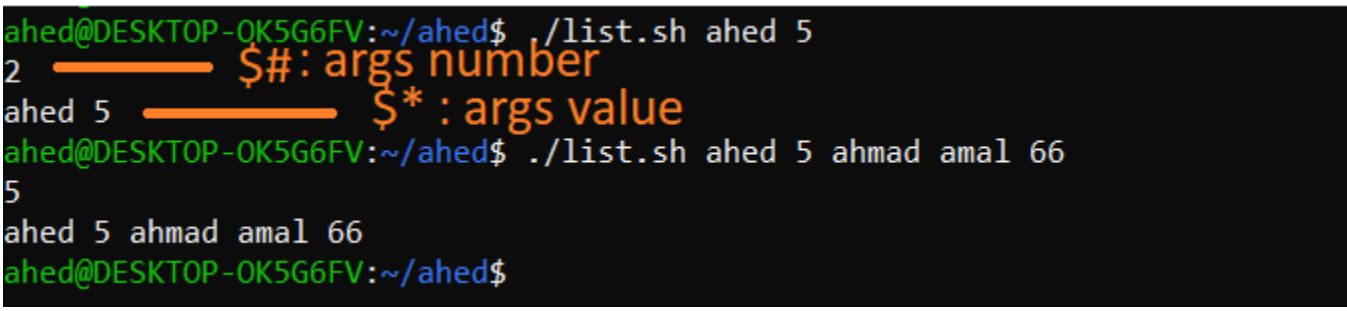
Output:

```
ahed@DESKTOP-OK5G6FV:~/ahed$ ./list.sh 'ahed mafarjeh' 5
ahed mafarjeh
5
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- اي argument فوق 10 لايمكن استقبالها هيك $10، لازم تعمل هيك ${10}$
- ويوجد متغيريين اخرين مهمين وهم :

  $# (the number of args), $* (the value of args)

Output:

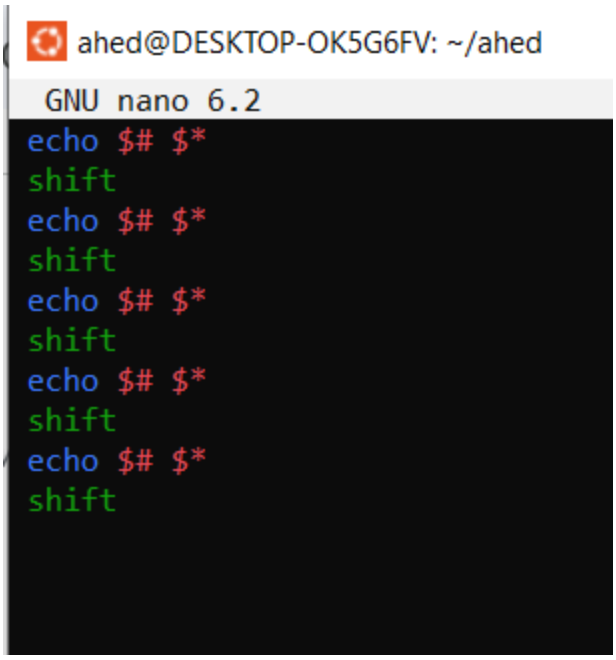


Q)

- Create text file and write the names inside it
- Write a shell script that adds any name to the text file, name must be passed as argument.
- Write a shell script that removes and given name in the file, also the name must be passed as argument.

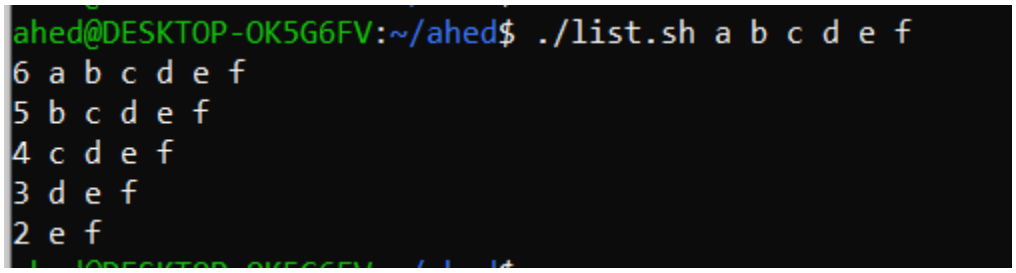
12- shift command:

- يستخدم لعمل ازاحة shift بين arg variable مثلا اذا بعمل shift بمقدار 1 بتنتقل قيمة $2 الى $1 ، وقيمة $1 خلص بتروح
- واذا عملت shift بمقدار 2 بصير قيمة $3 تنتقل الى $1 وقيمة $2 خلص بتروح لانه انعملها shift بمقدار 2



Shift by 1

Output :



- If you want to shift be 3 just change shift to shift 3

13- exit status:

- من خلالها بقدر اعرف حالة الكوماند يعني هل تنفذ صح او لأ
- اذا اعطاني 0 معناها تنفذ صح ، اذا اعطتني اي رقم اكبرمن 0 معناها الكوماند تنفذ غلط .
- لمعرف exit status نستخدم المتغير الخاص $?

```
txt  hard  list.sh  main_dir  msg.txt  names.txt  new.txt  soft
o $?
```

- مثلا في الصورة ، استخدمت كوماند ls وتم تنفيذه بشكل صحيح ف لما طبعت exit status اعطاني 0 يعني تم تنفيذ الكوماند بنجاح

```
ahed@DESKTOP-OK5G6FV:~/ahed$ rm ahed.txt
rm: cannot remove 'ahed.txt': No such file or directory
ahed@DESKTOP-OK5G6FV:~/ahed$ echo $?
1
ahed@DESKTOP-OK5G6FV:~/ahed$
```

- في المثال هاد عملت حذف لملف مش موجود اصلا ف ما تم تنفيذ الكوماند بنجاح ولما طبعت exit status اعطاني 1 يعني لم يتم تنفيذ الكوماند بنجاح.

Task: write a shell script that takes 7 numbers as arguments, then your script must print the average value for the 7 numbers.