Shell script part 3

Loops

1- for loop:

* Syntax:

 A- Iterating Over a List

 

 Example:

  

  

 B- Iterating Over a Range of Numbers (with bash shell)

 

 Example :

  

  

 C- The for Without the List

* اذا ما اعطيتها list رح تعمل loop على arguments :

 



* في شكل ثاني ل for loop الي هو :

 

* لكن هذا الشكل لا يعمل الا مع bash shell ولا يعمل مع sh shell
* عشان احدد نوع shell بستخدم شي اسمه shebang (#!) وهاي بتكون اول سطر في shell script :

 

* اذا غيرتها ل sh مش رح تشتغل لانه هاذ الشكل (syntax) مش معروف ب sh :

 

* طبعا ال default shell هي bash

2- while

  

3- until:

* هي عكس while يعني ما دام الشرط خطأ (برجع exit status > 0 ) بضل تشتغل:

 

 

4- break and continue:

* break:

 

* break n :

 معناها اطلع من n loop يعني اذا في 3 loops في جوا بعض nested وحطيت break 2 رح يطلع من 2 inner loop

 

* continue:

 

* continue n : نفس مبدأ break n
* Summary:
* **break n**: Exits from the n-th enclosing loop.
* **continue n**: Skips to the next iteration of the n-th enclosing loop.

5- getopts :

* هو shell command بستخدمه عشان اعمل option ل shell script
* طبعا بعطي الاوبشن من خلال arguments
* لازم استخدمه مع loop
* syntax :

getopts optstring variable

* optstring:
* بستخدمها عشان اعرف ال option ل shell script
* يكون عبارة عن string ، كل char في string يعبر عن option ، واذا بدي اخلي option يوخذ argument بحط وراه( : )
* مثلا "a:b" معناها انه في 2 options الي هم a و b ، و a بتوخذ argument لانه وراها (:)
* variable:

 رح يكون فيه قيم option الحالي الذي يتم معالجته

* في كمان variables 2 الي هم :
* OPTIND: بتكون فيه قيمة الاندكس ل الاوبشن التالي (with bash shell)
* OPTARG: بتكون فيه قيمة arg ل الاوبشن الحالي

Example :

 

* في هاد المثال بحكيله في 3 options الي هم (a,b,c ) وكمان c بتوخذ argument
* في كل لفة في اللوب بمسكهم اوبشن اوبشن ف ببلش ب a ، اذا انا معطيه اوبشن a رح ينفذ السطر التالي :

echo “$flag” $OPTIND $OPTARG

* طبعا flag الي هو بكون في الاوبشن a
* و OPTIND هو الاندكس ل الاوبشن الي بعد a ،
* و OPTARG هو قيم argument ل الاوبشن a لكن اوبشن a ما بوخذ argument ف مش رح يطبع شي
* خلينا نشغل script ونشوف الاوتبوت :

 

* لما شغلت script مررت الاوبشن هيك -abc وممكن تحطو كل وحده لحال
* We can use the options -a, -b, and -c with an argument like this: -a -b -c "soso". However, since options -a and -b do not require arguments, we can also combine them and write it as -abc "soso"
* كيف ينفذ الكود؟ اول شي بفحص بالكود اوبشن a وبشوف هل انا مررت اوبشن a
* اه انا مررت اوبشن a لما شغلت script ف بروح بدخل بجملة الطباعة echo ….
* ف بطبع flag الي بكون فيه قيمة الاوبشن الحالي وهو a
* وبعدين بطبع قيمة OPTIND الي هو الاندكس للاوبشن الي بعد a وهو اوبشن b ، وطبعا اوبشن b موجود ب اندكس 1، ليش؟
* 
* لانه b موجود في argument الاولى
* لما يوصل عند c ، ايش الاندكس للاوبشن الي بعدها ، اكيد مش 2 لانه الاندكس 2 هو عبارة عن argument لاوبشن c ف عشان هيك هو بتوقع يجي كمان اوبشن في اندكس 3 ف عشان هيك عند c بطبع 3

6- read : بستخدمها عشان اقرأ قيمة من اليوزر او من ملف

 

* في هاد المثال قرأنا من اليوزر الاسم والعمر وخزناهم في name, age وبعدين طبعناهم
* و ممكن في نفس السطر اقرأ اكثر من قيمة واحطهم في variables :

 

* وممكن تخفي الانبوت الي بكتبه اليوزر في حال مثلا بده يكتب باسورد (read -s) (with bash shell):

 

* منقدر نستخدمها عشان نقرأ ملف كمان :

 

* -r : يعني اذا لقى \ مش رح يعتبرها escape رح يشوفها ك حرف عادي

 

7- sleep :

* من خلالها بقدر اعمل delay او يعني رح يوقف تنفيذ code لفترة زمنية معينة
* Examples :

 

 

 

 

8- printf :

* بتستخدمها عشان اطبع اي شي نفس echo
* لكن الي بميزها عن echo انه بقدر اتحكم في شكل output

 

* Examples :

 

 



Tasks)

1. Write a shell script that accepts an unknown number of integer arguments and calculates their average.
2. Write a shell script that checks the current minutes every 3 seconds. If the minutes exceed 30, it will print the current time in the format (hour:min:sec).
3. Write a shell script that accepts an unknown number of inputs from the user and calculates their average. The script will print the average when the user enters 0.
4. Write a shell script that accepts four options: `-a`, `-s`, `-n1`, and `-n2`. If the `-a` option is specified, the script will print the sum of the values for `n1` and `n2`. If the `-s` option is specified, the script will print the subtraction of `n1` from `n2`.
5. Write a shell script that reads a list of products along with their prices and quantities from the user, then prints a formatted table displaying this information along with the total price for each product. The script should also print the grand total at the end.
6. Write a shell script that takes a string input from the user and splits it into individual characters.
7. Develop a shell script to output the odd lines from a particular file (without using the read command).
8. Write a shell script that creates a backup of a specified directory. The backup should be stored in a time stamped directory, and it should include all subdirectories and files.