Exercices shell scripts

Entraînez-vous avec Bash (ou autres langages shell compatibles)

Par Idriss NEUMANN

Date de publication : 13 décembre 2009

Dernière mise à jour : 20 juillet 2014

DÉBUTANT

Cet article regroupe des exercices corrigés en shell scripts Bash et autres langages shell compatibles. Les exercices sont classés par niveaux et peuvent posséder plusieurs solutions.

N'hésitez pas à proposer vos solutions où les énoncés via ce lien : Commentez ♪

I - Niveau débutant

I-A - Exercice 1 - Appréciation de note

I-A-1 - Énoncé

Créer un script qui demande à l'utilisateur de saisir une note et qui affiche un message en fonction de cette note :

- « très bien » si la note est entre 16 et 20 ;
- « bien » lorsqu'elle est entre 14 et 16 ;
- « assez bien » si la note est entre 12 et 14 ;
- « moyen » si la note est entre 10 et 12 ;
- « insuffisant » si la note est inférieur à 10.

I-A-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 1
```
#!/bin/bash

echo "Entrez votre note :"
read -r note

if [ "$note" -ge 16 ]; then
    echo "très bien"
elif [ "$note" -ge 14 ]; then
    echo "bien"
elif [ "$note" -ge 12 ]; then
    echo "assez bien"
elif [ "$note" -ge 10 ]; then
    echo "moyen"
else
    echo "insuffisant"
fi
```

I-B - Exercice 2 - Appréciation de note (v2)

I-B-1 - Énoncé

Reprenez l'exercice 1 et faites en sorte que le programme se répète tant que l'utilisateur n'a pas saisi une note négative ou 'q' (pour quitter).
Le script doit calculer le nombre de notes de saisies et en faire la moyenne tout à la fin.

I-B-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 2
```
#!/bin/bash

note=0
moyenne=0
i=0

until [ "$note" -lt 0 ]; do
```

solution exercice 2

```
    echo "Entrez votre note (q pour quitter) :"
    read -r note
    if [ "$note" = "q" ]; then
        note=-1
        echo "au revoir !"
    elif [ "$note" -ge 16 ]; then
        echo "très bien"
    elif [ "$note" -ge 14 ]; then
        echo "bien"
    elif [ "$note" -ge 12 ]; then
        echo "assez bien"
    elif [ "$note" -ge 10 ]; then
        echo "moyen"
    elif [ "$note" -ge 0 ]; then
        echo "insuffisant"
    else
        echo "au revoir !"
    fi

    if [ "$note" -ge 0 ]; then
        let moyenne=$moyenne+$note
        let i=$i+1
    fi
done

if [ "$i" -le 0 ]; then
    let i=1
fi

let moyenne=$moyenne/$i
echo "La moyenne est de $moyenne ($i notes)"
```

I-C - Exercice 3 - Nombre élevé à sa propre puissance

I-C-1 - Énoncé

Créer un script qui prend un nombre en saisie et l'élève à sa propre puissance. C'est un peu le même principe que la factorielle mais cette fois, l'usage de la boucle for est imposé.

Exemple d'exécution :

```
[ ~] ./NomDuScript.sh
Saisir une valeur :
2
2^2 = 4
```

I-C-2 - Solution 1

Solution proposée par ok.Idriss :

Solution 1 exercice 3

```
#!/bin/bash
echo "Saisir une valeur"
read -r value
result=1
for (( i=0 ; i<$value ; i++ )); do
    let result=$result*$value
done
echo "$value^$value = $result"
```

I-C-3 - Solution 2

Solution proposée par ok.Idriss :

solution 2 exercice 3
```
#!/bin/bash

operation () {
    result=1
    for (( i=0 ; i<$value ; i++ ))
    do
        let result=$result*$value
    done
    echo "$value^$value = $result"
}

if [ "$#" -eq 0 ]; then
    echo "Saisir une valeur"
    read -r value
else
    value=$1
fi
operation
```

II - Niveau intermédiaire

II-A - Exercice 1 - Appréciation de note (v3)

II-A-1 - Énoncé

Reprenez uniquement la version 1 de l'exercice. La note devra être donnée en paramètre ou bien saisie en cas d'absences d'arguments. La comparaison de la note devra être faite dans une fonction appreciation().

II-A-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 1
```
#!/bin/bash

appreciation () {
    if [ "$note" -ge 16 ]; then
        echo "très bien"
    elif [ "$note" -ge 14 ]; then
        echo "bien"
    elif [ "$note" -ge 12 ]; then
        echo "assez bien"
    elif [ "$note" -ge 10 ]; then
        echo "moyen"
    else
        echo "insuffisant"
    fi
}

# programme principal
clear
if [ "$#" -ne 0 ]; then
    note=$1
else
    echo "Saisir une note"
    read -r note
```

solution exercice 1
```
fi
appreciation
```

II-B - Exercice 2 - TestUser

II-B-1 - Énoncé

Créer un script qui vous propose le menu suivant :

```
1 - Vérifier l'existence d'un utilisateur
2 - Connaître l'UID d'un utilisateur
q - Quitter
```

L'utilisateur devra être saisi, à l'aide d'une fonction. Son existence devra être vérifiée à l'aide d'une autre fonction.

II-B-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 2
```
#!/bin/bash

function pause {
    echo "Appuyez sur ENTER pour continuer"
    read
}

function saisirUser {
    echo "Saisir l'utilisateur"
    read -r util
}

function verifyUser {
    if grep "^$util:" /etc/passwd > /dev/null; then
        echo "L'utilisateur existe"
    else
        echo "L'utilisateur n'existe pas"
    fi
    pause
}

rep=1
while [ "$rep" -eq 1 ]; do
    clear
    printf "menu :\n\n"
    echo "1. Vérifier l'existence d'un utilisateur"
    echo "2. Connaître l'UID d'un utilisateur"
    echo -e "3. Quitter\n"
    read -r choix
    case "$choix" in
        1)
            saisirUser
            verifyUser ;;

        2)
            saisirUser
            id $util
            pause ;;

        q)
            echo "Au revoir"
            pause
            rep=0 ;;
```

solution exercice 2
```
        *)
            echo "Erreur de saisie"
            pause ;;
    esac
done
```

II-C - Exercice 3 - Calculatrice

II-C-1 - Énoncé

Créer un script dans lequel deux nombres opérandes et un signe opérateur (+-*/) devront être donnés en paramètres, ou saisis. Le script doit réaliser l'opération souhaitée.

Exemple :

```
[ ~] ./calculette.sh 7 + 4
Le résultat est : 11
```

Le calcul devra être fait à l'aide d'une fonction calcul ().

II-C-2 - Solution 1

Solution proposée par ok.Idriss :

solution 1 exercice 3
```
#!/bin/bash

saisir () {
    printf "Saisir le premier nombre, puis le signe de l'opération puis le deuxième nombre :\n
\n"
    read -r nb1
    read -r s
    read -r nb2
}

calcul () {
    case "$s" in
        "+") let result=$nb1+$nb2 ;;
        "-") let result=$nb1-$nb2 ;;
        "*") let result=$nb1*$nb2 ;;
        "/") let result=$nb1/$nb2 ;;
        *)
            let result=0
            echo -e "Erreur de saisie !\nLe résultat est faux.";;
    esac
}

calcul2 () {
    let result=$nb1$s$nb2
}

if [ "$#" -eq 3 ]; then
    nb1=$1 ; s=$2 ; nb2=$3
else
    saisir
fi
calcul
echo "Le résultat est $result"
calcul2
echo "Calculé d'une autre façon : $result"
```

II-C-3 - Solution 2

Solution proposée par ok.Idriss :

solution 2 exercice 3
```
#!/bin/bash

if [ "$#" -lt 3 ]; then
    echo "Erreur : Il manque des paramètres !"
elif  [[ "$1" =~ ^[0-9]+$ ]] && [[ "$3" =~ ^[0-9]+$ ]]; then
    if [[ "$2" =~ ^(\+|\-|\/|\*){1}$ ]]; then
        if [ $3 -ne 0 ] || [ "$2" != "/" ]; then
            echo "Le résultat est : "$(( $1 $2 $3 ))
        else
            echo "Erreur : division par 0 !"
        fi
    else
        echo "Erreur : opérateur invalide !"
    fi
else
    echo "Erreur : opérandes invalides !"
fi
```

II-C-4 - Solution 3

Solution proposée par ok.Idriss :

solution 3 exercice 3
```
#!/bin/sh

if [ "$#" -lt 3 ]; then
    echo "Erreur : Il manque des paramètres !"
elif echo "$1$3" | grep -E "^[0-9]{2,}$" > /dev/null; then
    if echo "$2" | grep -E "^(\+|\-|\/|\*){1}$" > /dev/null; then
        if [ $3 -ne 0 ] || [ "$2" != "/" ]; then
            echo "Le résultat est : "$(( $1 $2 $3 ))
        else
            echo "Erreur : division par 0 !"
        fi
    else
        echo "Erreur : opérateur invalide !"
    fi
else
    echo "Erreur : opérandes invalides !"
fi
```

II-D - Exercice 4 - La factorielle

II-D-1 - Énoncé

Créer un script qui permet de calculer et d'afficher la factorielle d'un nombre donné en paramètre (ou saisi en cas d'absence de paramètres).

II-D-2 - Solution 1

Solution proposée par ok.Idriss :

solution 1 exercice 4
```
#!/bin/bash
```

solution 1 exercice 4

```
if [ "$#" -eq 0 ]; then
    echo "Saisir une valeur : "
    read -r val
else
    val=$1
fi

# Dans le cas où c'est négatif, on rend la valeur positive
if [ "$val" -lt 0 ]; then
    let val=-1*$val
fi

result=1
val2="$val"

while [ "$val" -ne 0 ]; do
    printf "$val "
    let result=$result*$val
    let val=$val-1
    if [ "$val" -ne 0 ]; then
        printf "* "
    fi
done

echo "= $result"
```

II-D-3 - Solution 2

Solution proposée par Sve@r :

solution 2 exercice 2

```
#!/bin/sh

if test "$#" -eq 0; then
    echo "Saisissez une valeur correcte"
    read -r val
    set -- $val
fi

nb=${nb:-$1}
res=${res:-1}
if test "$nb" -eq 0; then
    echo $res
    exit
fi

res=`expr $res \* $nb`
nb=`expr $nb - 1`
. $0
```

II-E - Exercice 5 - Les fichiers

II-E-1 - Énoncé

Créer un script qui doit calculer le nombre de fichiers standard, de sous-répertoires, et d'exécutables d'un répertoire quelconque qui sera donné en paramètre (ou saisis en cas d'absence du paramètre).

II-E-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 5

```
#!/bin/bash

j=0
k=0
l=0

if [ "$#" -eq 0 ]; then
    echo "Saisir le répertoire"
    read -r rep
else
    rep=$1
fi

cd $rep

for i in *; do
    if [ -d "$i" ]; then
        echo "$i"
        let j=$j+1
    fi
    if [ -f "$i" ]; then
        echo $i
        let k=$k+1
    fi
    if [ -x "$i" ]; then
        echo $i
        let l=$l+1
    fi
done
echo "Il y a $j répertoires, $k fichiers et $l exécutables dans $rep"
```

II-F - Exercice 6 - Tri à bulle

II-F-1 - Énoncé

Créer un script qui devra enregistrer à l'aide d'un tableau, un nombre d'entiers donné en paramètre (ou en saisie) puis trier ceux-ci dans l'ordre croissant dans ce même tableau (sans passer par un autre) et enfin afficher le contenu du tableau (ordonné) sur la sortie standard.

II-F-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 6

```
#!/bin/bash

if [ "$#" -lt 1 ]; then
    echo "Saisir le nombre d'éléments à ordonner"
    read -r SIZE
else
    SIZE=$1
fi

echo "Saisir les éléments :"

for (( i=0 ; i<SIZE ; i++ )); do
    read -r tab[i]
done

for (( i=0 ; i<SIZE ; i++ )); do
    for (( j=$i+1 ; j<SIZE ; j++ ))
    do
        if [ "${tab[$j]}" -le "${tab[$i]}" ]; then
```

solution exercice 6

```
                tampon="${tab[$i]}"
                tab[$i]="${tab[$j]}"
                tab[$j]="$tampon"
            fi
        done
done

echo "Valeurs ordonnées :"

for (( i=0 ; i<SIZE ; i++ )); do
    echo "${tab[$i]}"
done
```

II-G - Exercice 7 - ls avec ordre inversé

II-G-1 - Énoncé

Créer un script qui renvoie la même sortie que la commande ls, mais dans l'ordre décroissant (autrement dit : le script devra lister le contenu d'un répertoire dans l'ordre décroissant). Vous ne devez ni vous servir de la commande ls, ni de la commande sort.

II-G-2 - Solution 1

Solution proposée par ok.Idriss :

solution 1 exercice 7

```
#!/bin/bash

# Si on a au moins un paramètre et que le premier paramètre est un répertoire
if [ "$#" -ge 1 ] && [ -d "$1v" ]; then
    cd $1
fi

nb=0

for fichier in *; do
    tab[$nb]="$fichier"
    let nb=$nb+1
done

# affichage inversé
for (( i=$nb ; i>=0 ; i=$i-1 )); do
    echo ${tab[$i]}
done
```

II-G-3 - Solution 2

Solution proposée par becket :

solution 2 exercice 7

```
#!/bin/bash

listing=( * )

for (( i=${#listing[@]}-1 ; i >= 0 ; i=i-1 )); do
    echo ${listing[$i]}
done
```

II-G-4 - Solution 3

Solution proposée par N_Bah :

solution 3 exercice 7
```
#!/bin/bash

read -rp 'Entrez le nom du répertoire : ' repertoire

if [ -d "$repertoire" ]; then
    [[ $repertoire != */  ]] && repertoire="${repertoire:+$repertoire/}"
    shopt -s nullglob #sinon repertoireVide/*, retournera repertoireVide/*
    fichiers=( ${repertoire}* )
    ((${#fichiers[@]})) && {
        for (( x=${#fichiers[@]}-1; x>=0; x-- )); do
            echo "${fichiers[x]}"
        done
    } || echo "$repertoire est vide"
else
    echo "$repertoire n'est pas un répertoire"
fi
```

II-G-5 - Solution 4

Solution proposée par chardclo :

solution 4 exercice 7
```
#!/bin/bash
read -rp 'Entrez le nom du répertoire : ' repertoire
[ -d "$repertoire" ] || {
    printf "%s n'est pas un nom de dossier valide." "$repertoire"
    exit 1
}
compgen -o default "${repertoire}/" | tac
```

II-G-6 - Solution 5

Solution proposée par chardclo :

solution 5 exercice 7
```
#!/bin/bash
read -rp 'Entrez le nom du répertoire : ' repertoire
[ -d "$repertoire" ] || {
    printf "%s n'est pas un nom de dossier valide.\n" "$repertoire"
    exit 1
}

text=""
while read -r; do
    text="${REPLY}\n${text}"
done < <(compgen -o default "${repertoire}/")

[[ ${text} ]] && printf "$text" || printf "le dossier %s est vide.\n" "$repertoire"
```

III - Niveau confirmé

III-A - Exercice 1 - Fichier de notes

III-A-1 - Énoncé

Créer un script qui va devra exploiter les données d'un fichier de notes que vous allez créer au préalable dans le même répertoire que le script.
Ce fichier sera appelé FichierNote.txt et devra se présenter comme ceci :

```
Dupont François 12
Durand Françoise 8
Dujardin Nicole 14
```

Le script devra afficher les lignes dans lesquelles la note est supérieure ou égale à 10.

Exemple :

solution exercice 1
```
[ ~] ./NomDuScript
Dupont        François        12
Dujardin Nicole        14
```

III-A-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 1
```
#!/bin/bash

fichier="FichierNote.txt"

while read -r ligne; do
    set -- "$ligne"
    if [ "$3" -ge 10 ]; then
        echo "$ligne"
    fi
done < $fichier
```

III-B - Exercice 2 - Liste d'utilisateurs

III-B-1 - Énoncé

Créer un script qui prend en paramètre (ou en saisie en cas d'absence du paramètre) un fichier qui contient des lignes comme ceci : Login + Tabulation + Nom + Tabulation + Prénom.

Exemple :

```
dupontf Dupont François
fdurand Durand Françoise
nicoled Dujardin Nicole
```

Le script devra vérifier, à l'aide d'une fonction, l'existence des utilisateurs enregistrés dans le fichier. Admettons, par exemple, que seul Dupont François soit un utilisateur et que le fichier se nomme ~/Documents/FichierUser, le script devra s'exécuter comme ceci :

```
[ ~] ./NomDuScript ~/Doucuments/FichierUser
dupontf        Dupont        François
[ ~] ./NomDuScript
Saisissez le fichier à traiter :
~/Documents/UserFichier
Le fichier n'existe pas.
[ ~]
```

Le script devra donc, au préalable, vérifier l'existence du fichier avant de comparer son contenu au fichier /etc/passwd. Le script devra également quitter la boucle si le fichier est vide.

III-B-2 - Solution

Solution proposée par ok.Idriss :

solution exercice 2

```
#!/bin/bash

TestUser () {
    if grep "^$util:" /etc/passwd > /dev/null
    then
        echo $ligne
    fi
}

if [ "$#" -eq 0 ]; then
    echo "Chemin et nom du fichier :"
    read -r fichier
else
    fichier="$1"
fi

if [ -e "$fichier" ]; then
    while read -r ligne; do
        set -- "$ligne"
        util="$1"
        TestUser
    done < $fichier
else
    echo "Le fichier $fichier n'existe pas."
fi
```

III-C - Exercice 3 - Convertisseur décimal/binaire

III-C-1 - Énoncé

Créer un script qui prend en paramètre (ou en saisie en cas d'absence de paramètres) une valeur décimale et qui doit la convertir en binaire.
Vous devez travailler sur 8 bits et chaque bit devra être contenu dans une case d'un tableau monodimensionnel et à la fin on affiche toutes les cases de ce tableau pour avoir la valeur en binaire lisible de droite à gauche, à partir de la valeur décimale saisie au départ.

III-C-2 - Solution 1

Solution proposée par ok.Idriss :

solution 1 exercice 3

```
#!/bin/bash

MaxBits=8
```

solution 1 exercice 3

```
pow () {
    value2=1
    for (( k=1 ; k<$i ; k ++ )); do
        let value2=$value2*2
    done
}

if [ "$#" -eq 0 ]; then
    echo "Saisir une valeur décimale"
    read -r value
else
    value="$1"
fi

declare -a bin

j=0
for (( i="$MaxBits" ; i>=0 ; i-- )); do
    pow
    if [ "$value" -ge "$value2" ]; then
        bin[$j]="1"
        let value=$value-$value2
    else
        bin[$j]="0"
    fi
    let j=$j+1
done

printf "La valeur binaire est de : "
for (( i=0 ; i<$MaxBits ; i++ )); do
    printf "${bin[$i]}"
done
echo ""
```

III-C-3 - Solution 2

Solution proposée par becket :

solution 2 exercie 3

```
#!/bin/bash

unset resultat
echo  "Entrez une valeur a convertir : "
read -r i
for (( cpt=8 ; $cpt > 1  ; cpt-- )); do
        let resultat[$cpt]="$i & 1"
        let i="i >>= 1"
done
echo "${resultat[@]}"
```

III-C-4 - Solution 3

Solution proposée par chardclo :

solution 3 exercice 3

```
#!/bin/bash

unset res
dec=$1
[[ $dec ]] || read -rp  "Entrez une valeur a convertir : " dec
while true; do
    res[$dec]=$((dec & 1))
    ((dec>>=1 )) || break
```

solution 3 exercice 3
```
done
echo "${res[@]}"
```

III-D - Exercice 4 - Moyenne de notes sur un fichier

III-D-1 - Énoncé

Créer un script qui prend en paramètre ou en saisie le nom d'un fichier contenant le nom des élèves et leurs trois notes. Le script devra : afficher les noms des élèves, puis calculer et afficher la moyenne de chaque élève.
Voici comment se présente le fichier :

```
Durand 12 9 14
Lucas 8 11 4
Martin 9 12 1
```

III-D-2 - Solution 1

Solution proposée par ok.Idriss :

solution 1 exercice 4
```
#!/bin/bash

if [ "$#" -lt 1 ]; then
    echo "Saisir le nom du fichier"
    read -r fichier
else
    fichier="$1"
fi

while read -r ligne; do
    set -- "$ligne"
    let moyenne=($2+$3+$4)/3
    echo "L'élève $1 a pour moyenne $moyenne"
done < "$fichier"
```

III-D-3 - Solution 2

Solution proposée par N_Bah :

solution 2 exercice 4
```
#!/bin/bash

moyenne() {
    declare -i somme
    for i in "$@"; do
        somme+="$i"
    done
    echo "$((somme / ${#@}))"
}

[ -f "$1" ] && fichier="$1" || read -rep 'Entrez le nom du fichier qui contient les données : '
 fichier

while read -r nom notes; do
    echo -n "$nom : "
    moyenne $notes
done < "$fichier"
```

IV - Liens utiles

Quelques liens utiles permettant d'acquérir de bonnes bases :

Advanced Bash-Scripting Guide (traduction)

Un cours complet sur la programmation Shell

Quelques bonnes pratiques dans l'écriture de scripts en Bash

Présentation et cours Korn Shell (compatible avec le Bash)

La section « Le Shell » de la FAQ Linux

V - Remerciements

Je souhaite remercier tous les contributeurs qui m'ont aidé à enrichir cet article par des exercices et/ou des solutions. Il s'agit en l'occurrence de becket, chardclo, N_Bah et Sve@r merci à eux.

Je tiens également à remercier LittleWhite, Louson et paissad pour leur relecture technique et leurs conseils.

Je tiens enfin à remercier jacques_jean pour son effort de relecture orthographique.