

Introduction au Scripting Bash

COURS & EXERCICES

MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Section 1 : Introduction au Scripting Bash

Objectifs de la Section

les stagiaires seront capables de :

- Comprendre l'importance et les usages du scripting Bash.
- Créer, exécuter et documenter un script Bash de base.
- Utiliser les commentaires pour améliorer la lisibilité des scripts.

Introduction et Concepts de Base

Section 1 : Introduction au Scripting Bash

Objectifs de la Section

les stagiaires seront capables de :

- Comprendre l'importance et les usages du scripting Bash.
- Créer, exécuter et documenter un script Bash de base.
- Utiliser les commentaires pour améliorer la lisibilité des scripts.

Introduction et Concepts de Base

Section 1 : Introduction au Scripting Bash

1.1. Présentation du Shell Bash

Qu'est-ce que Bash ?

- Bash (Bourne Again Shell) est une interface en ligne de commande et un langage de script.
- Utilisé pour interagir avec le système d'exploitation, exécuter des commandes et automatiser des tâches.

Introduction et Concepts de Base

Section 1 : Introduction au Scripting Bash

1.1. Présentation du Shell Bash

Pourquoi utiliser Bash ?

- Efficacité : Automatisation des tâches répétitives.
- Puissance : Accès direct aux fonctionnalités du système.
- Portabilité : Les scripts Bash peuvent être utilisés sur toutes les distributions Linux et sur macOS.

Introduction et Concepts de Base

Section 1 : Introduction au Scripting Bash

1.1. Présentation du Shell Bash

Exemples concrets :

1. Automatisation des backups quotidiens d'un répertoire important.
2. Surveillance de l'utilisation du disque et envoi d'alertes par email.
3. Déploiement de mises à jour logicielles sur plusieurs serveurs.

Introduction et Concepts de Base

Section 1 : Introduction au Scripting Bash

1.2. Importance du Scripting Bash dans l'Administration Système

Utilisations courantes :

- Automatisation des tâches : Backup, mise à jour des systèmes, surveillance.
- Gestion des utilisateurs et des permissions.
- Maintenance du système : Nettoyage, monitoring.
- Déploiement d'applications.

Introduction et Concepts de Base

Section 1 : Introduction au Scripting Bash

1.2. Importance du Scripting Bash dans l'Administration Système

Exemples concrets :

1. Script pour créer des comptes utilisateurs en masse.
2. Script pour nettoyer les fichiers temporaires et les journaux système.
3. Script pour surveiller les services système et les redémarrer si nécessaire.

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.1. Création d'un Script Bash

Exemples pour créer un script :

- 1. Créer un fichier script :**
 - Utiliser un éditeur de texte (nano, vim, etc.)
 - Exemple : nano hello_world.sh
- 2. Ajouter des commandes au script :**
 - Commencer par la ligne de shebang (#!/bin/bash) pour indiquer l'interpréteur.
 - Ajouter des commandes Bash sous cette ligne.

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.1. Création d'un Script Bash

Exemple de script :

```
#!/bin/bash  
  
# Ceci est un script Bash simple  
  
echo "Hello, World!" # Ceci est commentaire
```

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.1. Création d'un Script Bash

Démonstration :

- Créez un script en direct et expliquer chaque étape.
- Exemple 1 : Script pour afficher la date et l'heure actuelles.

```
#!/bin/bash
```

```
echo "Current date and time: $(date)"
```



Introduction et Concepts de Base



Section 2 : Introduction au Scripting Bash

2.1. Création d'un Script Bash

Démonstration :

- Créez un script.
- Exemple 2 : Script pour afficher le répertoire courant.

```
#!/bin/bash
```

```
echo "Current directory: $(pwd)"
```

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.1. Création d'un Script Bash

Exercice Pratique 1:

Enoncé :

- Créez un script appelé **greet_user.sh** qui demande à l'utilisateur son nom et lui souhaite la bienvenue.
- Utilisez la commande **read** pour obtenir le nom de l'utilisateur.
- Affichez un message de bienvenue personnalisé en utilisant la variable contenant le nom de l'utilisateur.



Qualiopi
processus certifié

■ REPUBLIQUE FRANCAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.1. Crédit d'un Script Bash

Exercice Pratique 1 :

Correction :

```
#!/bin/bash
```

Script pour souhaiter la bienvenue à l'utilisateur

```
echo "Quel est votre nom ?"
```

```
read name
```

```
echo "Bienvenue, $name!"
```

MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.2. Rendre le Script Exécutable

Commande :

```
chmod +x hello_world.sh
```

Explication :

- **chmod** est utilisé pour changer les permissions des fichiers.
- **+x** rend le fichier exécutable.

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.2. Rendre le Script Exécutable

Démonstration :

- Rendre le script greet_user.sh exécutable

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.2. Rendre le Script Exécutable

Exemples supplémentaires :

1. Script pour afficher l'espace disque disponible.

```
#!/bin/bash
```

```
df -h
```

1. Script pour lister les fichiers d'un répertoire ou le répertoire courant.

```
#!/bin/bash
```

```
ls -l
```

Introduction et Concepts de Base

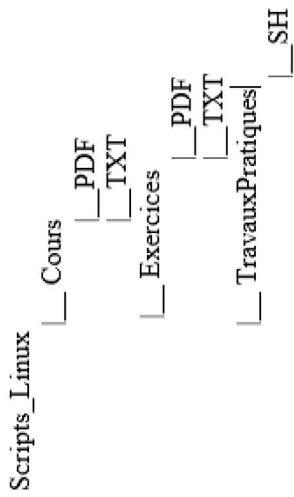
Section 2 : Introduction au Scripting Bash

2.2. Rendre le Script Exécutable

Exercice Pratique 2 :

Enoncé :

- Créez un script appelé disk_usage.sh qui affiche l'espace disque disponible sur le système.
- Utilisez la commande df -h pour obtenir les informations sur l'espace disque.
- Créer l'arborescence suivante dans le répertoire courant :



MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.2. Rendre le Script Exécutable

Correction :

```
#!/bin/bash
```

```
# Script pour afficher l'espace disque disponible
```

```
# Et création d'une arborescence
```

```
df -h
```

```
mkdir -p Scripts_Linux/Cours/{PDF, TXT}
```

```
mkdir -p Scripts_Linux/Exercices/{PDF, TXT}
```

```
mkdir -p Scripts_Linux/TravauxPratiques/SH
```

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.3. Exécuter le Script

Commande :

```
./hello_world.sh
```

Explication :

- ./ indique que le script se trouve dans le répertoire courant.
- Comme on peut utiliser aussi le chemin relatif



Qualiopi
processus certifié

■ REPUBLIQUE FRANCAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.3. Exécuter le Script

Exemples supplémentaires :

1. Script pour afficher les 5 derniers journaux système.

```
#!/bin/bash
```

```
tail -n 5 /var/log/syslog
```

MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.3. Exécuter le Script

Exemples supplémentaires :

2. Script pour afficher les informations sur l'uptime du système.

```
#!/bin/bash
```

```
uptime
```

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.3. Exécuter le Script

Exemples supplémentaires :

2. Script pour afficher les informations sur l'uptime du système.

```
#!/bin/bash
```

```
uptime
```

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.3. Exécuter le Script

Exemples supplémentaires :

2. Script pour afficher les informations sur l'uptime du système.

```
#!/bin/bash  
  
uptime
```

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.3. Exécuter le Script

Exercice Pratique 3 :

Enoncé :

- Créez un script appelé `system_info.sh` qui affiche les informations système telles que la charge du processeur, la mémoire disponible, etc.



Qualiopi
processus certifié

■ REPUBLIQUE FRANCAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Introduction et Concepts de Base

Section 2 : Introduction au Scripting Bash

2.3. Exécuter le Script

Exercice Pratique 3 :

Correction :

```
#!/bin/bash
```

```
# Script pour afficher les informations système
```

```
uptime
```

```
free -m
```

```
df -h
```

MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Section 3 : Commentaires dans les Scripts Bash

3.1. Utilisation des Commentaires

Pourquoi utiliser des commentaires ?

Améliorer la lisibilité et la maintenabilité du code.

Documenter l'objectif et le fonctionnement des parties du script.

Syntaxe :

Un commentaire commence par un # et tout ce qui suit sur la ligne est ignoré par l'interpréteur.

Introduction et Concepts de Base

Section 3 : Commentaires dans les Scripts Bash

3.1. Utilisation des Commentaires

Exemple :

```
# ! /bin/bash
```

```
# Ce script affiche "Hello, World!" à l'écran
```

```
echo "Hello, World!" # Affiche un message
```

Introduction et Concepts de Base

Section 3 : Commentaires dans les Scripts Bash

3.1. Utilisation des Commentaires

Exemples supplémentaires :

1. Script commenté pour afficher les processus en cours :

```
#!/bin/bash  
  
# Ce script affiche les processus en cours  
ps aux # Affiche la liste des processus
```

Introduction et Concepts de Base

Section 3 : Commentaires dans les Scripts Bash

3.1. Utilisation des Commentaires

Exemples supplémentaires :

- 2.** Script commenté pour vérifier la connectivité réseau :

```
#!/bin/bash
```

```
# Ce script ping une adresse IP pour vérifier la connectivité réseau  
ping -c 4 8.8.8.8 # Ping Google DNS avec 4 requêtes
```

Introduction et Concepts de Base

Section 3 : Commentaires dans les Scripts Bash

3.1. Utilisation des Commentaires

Exercice Pratique 4:

Enoncé :

- Ajoutez des commentaires à votre script `greet_user.sh` pour expliquer chaque étape.



La certification Qualité a été délivrée au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Introduction et Concepts de Base

Section 3 : Commentaires dans les Scripts Bash

3.1. Utilisation des Commentaires

Exercice Pratique 4:

Correction :

```
#!/bin/bash
```

```
# Script pour souhaiter la bienvenue à l'utilisateur
echo "Quel est votre nom ?" # Demande à l'utilisateur son nom
read name # Lit le nom de l'utilisateur depuis l'entrée standard
echo "Bienvenue, $name !" # Affiche un message de bienvenue personnalisé
```

MEL : 27/05/2024 au 29/05/2024



Qualiopi
processus certifié

RÉPUBLIQUE FRANÇAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations



IT GLOBAL INSTITUTE

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.1. Définition et Utilisation des Variables

Variables en Bash :

- Définition : variable _ name=value
- Utilisation : \$variable _ name

MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.1. Définition et Utilisation des Variables

Exemple :

```
# ! /bin/bash
```

```
name="Monir EL MOUNAOUI"
```

```
echo "Hello, $name"
```

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.1. Définition et Utilisation des Variables

Démonstration :

- Créer un script qui utilise des variables pour afficher un message personnalisé.
- Exemple 1 : Script pour afficher le nom de l'utilisateur courant :

```
#!/bin/bash
```

```
user=$(whoami)
```

```
echo "Current user: $user"
```

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.1. Définition et Utilisation des Variables

- Exemple 2 : Script pour définir et utiliser plusieurs variables :

```
#!/bin/bash
```

```
first_name="Monir"
```

```
last_name="EL MOUNAOUI"
```

```
echo "Full name: $first_name $last_name"
```



Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.1. Définition et Utilisation des Variables

Exercice Pratique 5 :

Enoncé :

Créez un script appelé `personal_info.sh` qui stocke votre nom, votre âge et votre ville dans des variables, puis les affiche à l'écran



Qualiopi
processus certifié
■ REPUBLIQUE FRANÇAISE
La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :
Action de formations

GTS Global Technology Solutions à Services

IT GLOBAL INSTITUTE

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.1. Définition et Utilisation des Variables

Exercice Pratique 5 :

Correction :

```
#!/bin/bash
```

```
# Script pour stocker et afficher des informations personnelles
```

```
name= "Samy"
```

```
age=25
```

```
city="Paris"
```

```
echo "Nom: $name, Âge: $age, Ville: $city"
```

MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.2. Opérations sur les Variables

Opérations arithmétiques :

- Addition : `sum=$((a + b))`
- Soustraction : `diff=$((a - b))`
- Multiplication : `prod=$((a * b))`
- Division : `quot=$((a / b))`



Qualiopi
processus certifié

■ REPUBLIQUE FRANCAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations



IT GLOBAL INSTITUTE

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

4.2. Opérations sur les Variables

Exemple :

```
# ! /bin/bash
```

```
a=5
```

```
b=3
```

```
sum=$((a + b))
```

```
echo "Sum: $sum"
```

MEL : 27/05/2024 au 29/05/2024

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

Démonstration :

- Exemple 1 : Script pour calculer et afficher la moyenne de trois nombres :

```
#!/bin/bash
```

```
num1=10
```

```
num2=20
```

```
num3=30
```

```
avg=$(( ( num1 + num2 + num3 ) / 3 ))
```

```
echo "Average: $avg"
```

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

- Exemple 2 : Script pour convertir une température de Celsius en Fahrenheit :

```
#!/bin/bash
```

```
celsius=25
```

```
fahrenheit=$( ( (celsius * 9/5) + 32 ) )
```

```
echo "$celsius°C = $fahrenheit°F"
```



Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

Exercice Pratique 5:

Enoncé :

- Créez un script appelé `température_converter.sh` qui convertit une température donnée en Celsius en Fahrenheit.
- Demandez à l'utilisateur de saisir une température en Celsius et affichez sa conversion correspondante en Fahrenheit.

Introduction et Concepts de Base

Partie 4 : Variables et Types de Données

Exercice Pratique 5:

```
#!/bin/bash
```

```
# Script pour convertir une température de Celsius en Fahrenheit  
  
echo "Entrez la température en Celsius : "  
  
read celsius  
  
fahrenheit=$( ( $(celsius * 9/5) + 32 ) )  
  
echo "Scelsius°C = $fahrenheit°F"
```

MEL : 27/05/2024 au 29/05/2024



Qualiopi
processus certifié

RÉPUBLIQUE FRANÇAISE
La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

Scripting Bash Avancé



IT GLOBAL INSTITUTE

Objectifs de la Section

À la fin de cette session, les stagiaires seront capables de :

- Utiliser des structures de contrôle telles que les boucles et les conditions dans les scripts Bash.
- Manipuler des chaînes de caractères et des tableaux.
- Comprendre les fonctions et les appels de fonctions dans Bash.

MEL : 27/05/2024 au 29/05/2024



Qualiopi
processus certifié

RÉPUBLIQUE FRANÇAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

Partie 1 : Structures de Contrôle

1.1. Conditions

Les structures de contrôle **`if-else`** :

- Permettent de prendre des décisions basées sur des conditions.

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

Partie 1 : Structures de Contrôle

1.1. Conditions

Syntaxe :

```
if condition  
then
```

```
# code à exécuter si la condition est vraie
```

```
else
```

```
# code à exécuter si la condition est fausse
```

```
fi
```



Qualiopi
processus certifié

■ REPUBLIQUE FRANCAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

Partie 1 : Structures de Contrôle

1.1. Conditions

Exemple :

```
#!/bin/bash

age=20

if [ $age -ge 18 ]
then
    echo "Vous êtes majeur."
else
    echo "Vous êtes mineur."
fi
```

MEL : 27/05/2024 au 29/05/2024



Qualiopi
processus certifié

RÉPUBLIQUE FRANÇAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

Partie 1 : Structures de Contrôle

1.1. Conditions

Démonstration :

- Crée un script qui utilise des instructions if-else pour déterminer si un nombre est pair ou impair.
- Exemple : Script pour vérifier si un nombre est pair ou impair :

MEL : 27/05/2024 au 29/05/2024



Scripting Bash Avancé

Partie 1 : Structures de Contrôle

1.1. Conditions

```
#!/bin/bash
```

```
echo "Entrez un nombre :"  
  
read num  
  
if [ $(($num % 2)) -eq 0 ]  
then  
    echo "Le nombre est pair."  
else  
    echo "Le nombre est impair."  
fi
```



Qualiopi
processus certifié

■ REPUBLIQUE FRANCAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

Partie 1 : Structures de Contrôle

1.1. Conditions

Exercice Pratique :

1. Exercice :

- o Créez un script appelé `check_grade.sh` qui demande à l'utilisateur sa note sur 100 et affiche "Réussi" si la note est supérieure ou égale à 60, sinon affiche "Échec".

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

Partie 1 : Structures de Contrôle

1.1. Conditions

Exercice Pratique :

Correction :

```
#!/bin/bash

# Script pour vérifier la réussite d'un examen

echo "Entrez votre note sur 100 : "
read note

if [ $note -ge 60 ]
then
    echo "Réussi"
else
    echo "Échec"
fi
```



Scripting Bash Avancé

1.2. Boucles

Les boucles `for` et `while`:

- Permettent de répéter des instructions plusieurs fois.

Syntaxe `for`:

```
for variable in liste
```

```
do
```

```
# code à exécuter
```

```
done
```

Scripting Bash Avancé

1.2. Boucles

Exemple :

```
# ! /bin/bash
```

```
for i in {1..5}
```

```
do
```

```
echo "Valeur de i : $i"
```

```
done
```



Qualiopi
processus certifié

■ REPUBLIQUE FRANÇAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

1.2. Boucles

Syntaxe while :

while condition

do

code à exécuter

done

MEL : 27/05/2024 au 29/05/2024



Qualiopi
processus certifié

■ REPUBLIQUE FRANCAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

1.2. Boucles

Exemple :

```
#!/bin/bash
```

```
count=1
```

```
while [ $count -le 5 ]
```

```
do
```

```
echo "Compteur : $count"
```

```
( (count++) )
```

```
done
```

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

1.2. Boucles

Démonstration :

- Créez un script qui utilise une boucle `for` pour afficher les noms des fichiers dans un répertoire spécifique.
- Exemple : Script pour afficher les noms des fichiers dans le répertoire courant :

```
#!/bin/bash
for file in *
do
    echo "Nom du fichier : $file"
done
```



processus certifié
■ REPUBLIQUE FRANÇAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

Scripting Bash Avancé



IT GLOBAL INSTITUTE

1.2. Boucles

Exercice Pratique 6 :

Énoncé :

- Créez un script appelé `countdown.sh` qui affiche un compte à rebours de 10 à 1, puis "Décollage !".

MEL : 27/05/2024 au 29/05/2024



Qualiopi
processus certifié

■ REPUBLIQUE FRANÇAISE

La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

1.2. Boucles

Exercice Pratique 6 :

Correction :

```
#!/bin/bash
```

```
# Script pour compte à rebours
```

```
for ((i=10; i>=1; i--))
```

```
do
```

```
echo "$i"
```

```
sleep 1
```

```
done
```

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

1.2. Boucles

Utilisation des arguments à l'exécution du script : \$0 , \$1 ,.....,\$9 :

Exemple 1:

Pour exécuter se script, il faut d'abord lui attribué le droit nécessaire
Puis appeler le script en lui passant les arguments comme suivant:
./script-argument.sh Bonjour Hello

```
# Afficher le nom du script ($0)
echo "Nom du script: $0"
# Afficher l'argument $1
echo " Valeur de l'argument 1 : $1"
# Afficher l'argument $2
echo " Valeur de l'argument 1 : $2"
```

1.2. Boucles

Utilisation des arguments à l'exécution du script : \$0 , \$1 ,....,\$9 :

Exemple 2:

Pour exécuter ce script, il faut d'abord lui attribué le droit nécessaire
Puis appeler le script en lui passant les arguments comme suivant:
`./script-argument.sh Jérôme 21`

```
# Afficher le nom du script ($0)
echo "Nom du script: $0"
# Utilisation des arguments $1 et $2 dans des variables
nom=$1
age=$2
# Afficher un message de bienvenue
echo "Bonjour $nom! Vous avez $age ans."
```

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 1 : Menu Simple pour l'Utilisateur

Etape 1 :

```
#!/bin/bash
```

```
# Afficher le menu
echo "Choisissez une option:"
echo "1) Afficher la date et l'heure"
echo "2) Afficher l'espace disque"
echo "3) Afficher les utilisateurs connectés"
echo "4) Quitter"
# Lire le choix de l'utilisateur
read -p "Entrez votre choix [1-4]: " choice
```

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 1 : Menu Simple pour l'Utilisateur

Etape 2 :

Utiliser la structure case pour exécuter une action en fonction du choix

```
case $choice in
```

```
    1)
```

```
        echo "La date et l'heure actuelles sont :"
```

```
        date
```

```
        ;;
```

```
    2)
```

```
        echo "L'espace disque utilisé est :"
```

```
        df -h
```

```
        ;;
```

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 1 : Menu Simple pour l'Utilisateur

Etape 3 :

- 3) echo "Les utilisateurs actuellement connectés sont :"
who
;;
 - 4) echo "Quitter le programme. Au revoir!"
exit 0
;;
- *) echo "Choix invalide! Veuillez entrer un nombre entre 1 et 4."
;;
esac

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action

:

```
#!/bin/bash
service=$1
action=$2
```

Exemple 2 : Gestionnaire de Services

Ce script permet de démarrer, arrêter ou redémarrer un service spécifié par l'utilisateur (il faut passer le nom du service et l'action en tant que arguments).

```
# Utiliser la structure case pour gérer le service
case $action in
    start)
        echo "Démarrage du service $service..."
        sudo systemctl start $service
        ;;
    stop)
        echo "Arrêt du service $service..."
        sudo systemctl stop $service
        ;;
    *) echo "Action non reconnue"
        exit 1
    esac
```

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 2 : Gestionnaire de Services

```
restart)
    echo "Redémarrage du service $service..."
    sudo systemctl restart $service
;;
status)
    echo "Statut du service $service :"
    sudo systemctl status $service
;;
*)
    echo "Action inconnue! Utilisez l'une des actions suivantes : start, stop, restart, status."
;;
esac
```

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 2 : Gestionnaire de Services

Exercice pratique 7 :

Enoncé :

1- Lister les Services :
systemctl list-unit-files --type=service --no-pager --no-legend | awk '{print \$1}'

2- Stocker les Services dans une Variable :
services=\$(...)

- 3- Interaction Utilisateur :
- Demande à l'utilisateur de choisir un service parmi ceux listés.
 - Vérifie si le service choisi existe.
 - Demande à l'utilisateur de choisir une action (status, start, stop, restart).

4- Exécuter l'Action :

Utilise une structure case pour exécuter l'action choisie par l'utilisateur sur le service spécifié.

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 2 : Gestionnaire de Services

Exercice pratique 7 :

Corrigé :

```
#!/bin/bash
# Lister tous les services installés et stocker dans une variable
services=$(systemctl list-unit-files --type=service --no-pager --no-legend | awk '{print $1}')

# Afficher la liste des services
echo "Services installés :"
echo "$services"

# Demander à l'utilisateur de choisir un service
read -p "Entrez le nom du service à gérer : " service

# Vérifier si le service existe
if echo "$service" | grep -q "^$service\$"; then
    # Demander à l'utilisateur de choisir une action
    echo "Choisissez une action pour le service $service :"
    echo "1) status"
    echo "2) start"
    echo "3) stop"
    echo "4) restart"
```

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 2 : Gestionnaire de Services

read -p "Entrez votre choix [1-4] : " action

Exercice pratique 7 :

```
# Utiliser la structure case pour exécuter l'action
case $action in
  1)
    echo "Vérification du statut du service $service..."
    sudo systemctl status $service
    ;;
  2)
    echo "Démarrage du service $service..."
    sudo systemctl start $service
    ;;
  3)
    echo "Arrêt du service $service..."
    sudo systemctl stop $service
    ;;
  *)
    echo "Action non reconnue"
  esac
```

Scripting Bash Avancé

1.3. Menu de Choix entre ensemble d'action :

Exemple 2 : Gestionnaire de Services

```
4)      echo "Redémarrage du service $service..."  
       sudo systemctl restart $service  
       ;;  
*)      echo "Choix invalide! Utilisez 1, 2, 3 ou 4."  
       ;;  
esac  
else  
echo "Service non trouvé: $service"  
fi
```

Exercice pratique 7 :

Corrigé :

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.1. Manipulation des Chaînes de Caractères

Opérations sur les chaînes de caractères :

- Concaténation : `concat="\$str1 \$str2"`
- Longueur : `length=${#str}`
- Extraction de sous-chaînes : `substring=${str:position:length}`

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.1. Manipulation des Chaînes de Caractères

Exemple :

```
# ! /bin/bash

str1="Hello"
str2="World"
concat="$str1 $str2"
echo "Concaténation : $concat"
```

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.1. Manipulation des Chaînes de Caractères

Démonstration :

- Créer un script qui manipule des chaînes de caractères en les concaténant et en extrayant des sous-chaînes.
- Exemple : Script pour extraire une sous-chaîne d'une chaîne donnée :

```
#!/bin/bash  
  
str="Bonjour tout le monde"  
  
substring=${str:8:4}  
  
echo "Sous-chaîne : $substring"
```



La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.1. Manipulation des Chaînes de Caractères

Exercice Pratique 7:

Énoncé :

- Créez un script appelé `reverse_string.sh` qui prend une chaîne de caractères en entrée et affiche sa version inversée.

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.1. Manipulation des Chaînes de Caractères

Correction :

```
#!/bin/bash
```

```
# Script pour inverser une chaîne de caractères
```

```
echo "Entrez une chaîne de caractères : "
```

```
read input
```

```
reversed=$ (echo $input | rev)
```

```
echo "Chaîne inversée : $reversed"
```

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.2. Utilisation des Tableaux

Définition d'un tableau :

- `array=(val1 val2 val3)`

Accès aux éléments du tableau :

- `${array[index]} }`

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.2. Utilisation des Tableaux

Exemple :

```
#!/bin/bash  
  
colors=("Rouge" "Vert" "Bleu")  
  
echo "Première couleur : ${colors[0]}"
```

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.2. Utilisation des Tableaux

Exemple :

```
#!/bin/bash  
  
colors= ("Rouge" "Vert" "Bleu")  
  
echo "Première couleur : ${colors[0]}"
```



La certification Qualité a été délivrée
au titre des catégories d'actions suivantes :

Action de formations

IT GLOBAL INSTITUTE



Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.2. Utilisation des Tableaux

Démonstration :

- Créer un script qui utilise un tableau pour stocker une liste de noms et les affiche.
- Exemple : Script pour afficher une liste de noms stockée dans un tableau : Philippe, Hassani, Mohamed, Nans

MEL : 27/05/2024 au 29/05/2024

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.2. Utilisation des Tableaux

Démonstration :

```
#!/bin/bash

names=("Philippe" "Hassani" "Mohamed" "Nans")

for name in "${names[@]}"
do
    echo "Nom : $name"
done
```

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.2. Utilisation des Tableaux

Exercice Pratique 8 :

Enoncé :

- Créez un script appelé `sum_array.sh` qui prend un tableau de nombres en entrée et calcule leur somme.

Scripting Bash Avancé

Partie 2 : Manipulation des Chaînes de Caractères et des Tableaux

2.2. Utilisation des Tableaux #!/bin/bash

Exercice Pratique 8 :

- Correction :

```
numbers=(10 20 30 40 50)
sum=0

for num in ${numbers[@]}; do
    sum=$((sum + num))
done

echo "Somme : $sum"
```