**Guida Bash Alternativa**

La programmazione Bash (Bourne Again Shell) è uno strumento potente per automatizzare operazioni in ambienti Unix-like come Linux e macOS. La Bash offre la possibilità di eseguire comandi direttamente nel terminale e di creare script per automatizzare sequenze di operazioni.

---

**1. Comandi Essenziali**

Alcuni comandi di base indispensabili in Bash sono:

ls: Elenca i file e le cartelle nella directory corrente.

cd: Cambia directory.

mkdir: Crea una nuova directory.

rm: Elimina file o cartelle.

cp: Copia file o cartelle.

mv: Sposta o rinomina file o cartelle.

echo: Stampa a schermo il testo passato.

Esempio:

echo "Benvenuto nella guida Bash!"

ls

mkdir nuova\_cartella

---

**2. Creazione di uno Script Bash**

Uno script Bash è un file di testo contenente comandi Bash eseguiti sequenzialmente.

Passaggi:

1. Crea un file con estensione .sh.

2. Aggiungi la direttiva shebang (#!/bin/bash) all'inizio del file.

3. Rendi il file eseguibile con chmod +x script.sh.

4. Esegui lo script con ./script.sh.

Esempio:

#!/bin/bash

echo "Ciao, questo è uno script Bash!"

---

**3. Uso delle Variabili**

Le variabili permettono di memorizzare e richiamare valori.

Sintassi:

nome="Anna"

echo "Ciao, $nome!"

Variabili speciali:

$0 → Nome dello script.

$? → Codice di uscita dell'ultimo comando.

$1, $2... → Argomenti passati allo script.

$# → Numero totale di argomenti.

Esempio:

#!/bin/bash

echo "Lo script si chiama: $0"

echo "Hai passato $# argomenti"

---

**4. Condizionali (if-else)**

I condizionali controllano il flusso del programma.

Sintassi:

#!/bin/bash

numero1=”8”

numero2=”8”

if [[ ${numero1} -eq ${f} ]]

then

echo"uguali"

else

echo "diversi"

fi

Operatori comuni:

-eq → uguale

-ne → diverso

-lt → minore

-gt → maggiore

---

**5. Ciclo For**

Il ciclo for permette di eseguire un'operazione più volte.

Esempio:

#!/bin/bash

for (( i=0; i<3; i++))

do

echo "marameo"

done

---

**6. Ciclo While**

Il ciclo while esegue un'operazione finché una condizione è vera.

Esempio:

#!/bin/bash

cont=”1”

while [[ $cont -le 3 ]]; do

echo "Iterazione: $cont"

((cont++))

done

---

**7. Switch Case**

case è utile per gestire più condizioni.

Sintassi:

#!/bin/bash

V=”3”

case ${V} in

1 | 2)

echo "aa"

;;

3)

echo "bb"

;;

\*)

echo"cc"

;;

esac

---

**8. Esempio Pratico: Backup di una Directory**

Un semplice script per creare un backup.

#!/bin/bash

origine="/home/utente/documenti"

destinazione="/home/utente/backup"

# Crea la cartella di destinazione se non esiste

if [[ ! -d "$destinazione" ]]; then

mkdir -p "$destinazione"

fi

# Copia dei file

cp -r "$origine"/\* "$destinazione"

echo "Backup completato!"

---

**9. Generazione di una Password Casuale**

Genera una password usando caratteri casuali.

#!/bin/bash

PASSWORD=$(date +%s%N | sha256sum | head -c12)

CARATTERI="!@#$%^&\*"

SPECIALE=$(echo "$CARATTERI" | fold -w1 | shuf | head -c1)

echo "${SPECIALE}${PASSWORD}${SPECIALE}"

---

**10. Creazione di un Utente (Login)**

Uno script con controllo dei privilegi per creare un nuovo utente.

#!/bin/bash

# Verifica se lo script è eseguito come root

if [[ "$UID" -ne 0 ]]; then

echo "Esegui lo script con privilegi di root"

exit 1

fi

# Richiedi dati utente

read -p "Inserisci il nome utente: " USER

read -p "Inserisci il commento: " COMMENT

read -sp "Inserisci la password: " PASSWORD

# Crea l'utente

useradd -c "$COMMENT" -m $USER

echo $PASSWORD | passwd --stdin $USER

# Verifica se la creazione ha avuto successo

if [[ "$?" -eq 0 ]]; then

echo "Utente $USER creato con successo!"

else

echo "Errore nella creazione dell'utente."

exit 1

Fi

---

**Esercizio 1: Creazione di uno script per il backup**

-Si utilizza un editor (come nano) per creare uno script di shell.

Script di shell (userbackup.sh):

#!/bin/bash  
  
# Ottieni il nome utente attualmente loggato  
user=$(whoami)  
  
# Definisci il nome del file di backup  
file=/tmp/${user}\_home\_$(date +%Y-%m-%d\_%H%M%S).tar.gz  
  
# Esegui il backup e crea il file compresso  
tar -czf $file /home/$user

# Messaggio di completamento  
echo "Backup di /home/$user completato. Dettagli sul file di backup creato:"  
ls -lh $file

Comandi per eseguire lo script:

Apri l'editor di testo: nano userbackup.sh

Inserisci lo script sopra e salva.

Rendi lo script eseguibile: chmod +x userbackup.sh

Esegui lo script: ./userbackup.sh

---

**Esercizio 2: Pianificare l’esecuzione automatica dello script con crontab**

Comandi per configurare crontab:

Apri l'editor di crontab:

crontab -e

Aggiungi la seguente riga alla fine del file:

0 4 \* \* \* /home/tuo\_utente/scripts/userbackup.sh

0 4: Alle 4:00.

/home/tuo\_utente/scripts/userbackup.sh: Percorso dello script.

Salva e chiudi. Il sistema eseguirà automaticamente lo script ogni giorno alle 4:00.

---

**Esercizio 3: Configurare uno script che esegua backup in base a condizioni diverse**

Eseguire lo script ogni giorno alle 2:30:

30 2 \* \* \* /path/myscript.sh

Eseguire lo script tutte le domeniche a mezzogiorno:

0 12 \* \* 0 /path/myscript.sh

Eseguire lo script ogni giorno alle 9:01 e alle 18:01:

1 9,18 \* \* \* /path/myscript.sh

Eseguire lo script il giorno 15 di ogni mese alle ore 20:00:

0 20 15 \* \* /path/myscript.sh

Verifica del funzionamento

Testa manualmente lo script per assicurarti che funzioni correttamente.

Controlla che crontab sia attivo e correttamente configurato:

crontab -l

Questo mostra l'elenco dei task pianificati.

Verifica l'esecuzione automatica:

Controlla i log di sistema per confermare che lo script sia stato eseguito.

grep CRON /var/log/syslog

Zanin Davide 4°A