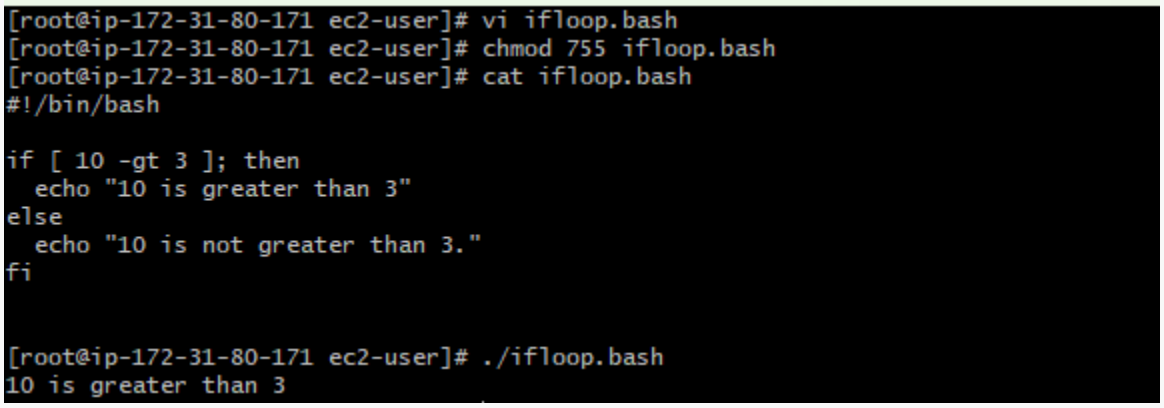
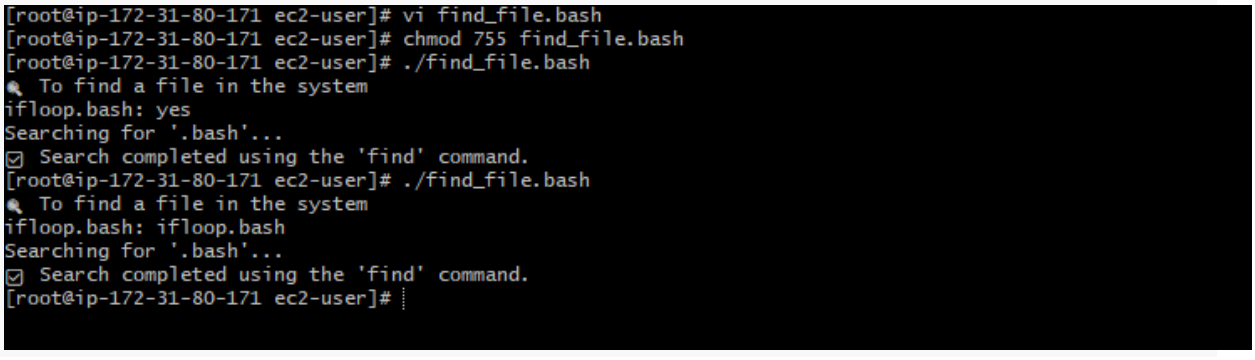
1. Execute all scripts from the devops-cloud-documents-13 channel.





```
[root@ip-172-31-80-171 ec2-user]# vi case_script.bash
[root@ip-172-31-80-171 ec2-user]# cat case_script.bash
#!/bin/bash
echo
echo Please chose one of the options below
echo
echo 'a = Display Date and Time'
echo 'b = List file and directories'
echo 'c = List users logged in'
echo 'd = Check System uptime'
echo
        read choices
        case $choices in
a) date;;
b) ls;;
c) who;;
d) uptime;;
*) echo Invalid choice - Bye.
                esac


[root@ip-172-31-80-171 ec2-user]# chmod 755 case_script.bash
[root@ip-172-31-80-171 ec2-user]# ./case_script.bash

Please chose one of the options below

a = Display Date and Time
b = List file and directories
c = List users logged in
d = Check System uptime

a
Fri Jun  6 07:23:32 UTC 2025
```

```
[root@ip-172-31-80-171 ec2-user]# vi case_script.bash
[root@ip-172-31-80-171 ec2-user]# cat case_script.bash
#!/bin/bash

NOW=$(date +"%a")  # Get current day abbreviation (Mon, Tue, etc.)

case $NOW in
  Mon)
    echo "Full backup"
    ;;
  Tue|Wed|Thu|Fri)
    echo "Partial backup"
    ;;
  Sat|Sun)
    echo "No backup"
    ;;
  *)
    echo "Invalid day"
    ;;
esac


[root@ip-172-31-80-171 ec2-user]# chmod 755 case_script.bash
[root@ip-172-31-80-171 ec2-user]# ./case_script.bash
Partial backup
```

```
[root@ip-172-31-80-171 ec2-user]# vi do_while.bash
[root@ip-172-31-80-171 ec2-user]# cat do_while.bash
#!/bin/bash
c=1
while [ $c -le 5 ]
do
        echo "Welcone $c times"
        (( c++ ))
done



[root@ip-172-31-80-171 ec2-user]# chmod 755 do_while.bash
[root@ip-172-31-80-171 ec2-user]# ./do_while.bash
Welcone 1 times
Welcone 2 times
Welcone 3 times
Welcone 4 times
Welcone 5 times
```

```
[root@ip-172-31-80-171 ec2-user]# vi do_while.bash
[root@ip-172-31-80-171 ec2-user]# chmod 755 do_while.bash
[root@ip-172-31-80-171 ec2-user]# cat do_while.bash
#!/bin/bash
count=0
num=10
while [ $count -lt 10 ]
do
        echo
        echo $num seconds left to stop this process $1
        echo
        sleep 1
num=`expr $num - 1`
count=`expr $count + 1`
done
echo
echo $1 process is stopped!!!
echo



[root@ip-172-31-80-171 ec2-user]# ./do_while.bash

10 seconds left to stop this process


9 seconds left to stop this process


8 seconds left to stop this process


7 seconds left to stop this process
```

```
[root@ip-172-31-80-171 ec2-user]# vi for_loop.bash
[root@ip-172-31-80-171 ec2-user]# chmod 755 for_loop.bash
[root@ip-172-31-80-171 ec2-user]# cat for_loop.bash
#!/bin/bash
for i in 1 2 3 4 5
do
echo "Welcome $i times"
done


[root@ip-172-31-80-171 ec2-user]# ./for_loop.bash
Welcome 1 times
Welcome 2 times
Welcome 3 times
Welcome 4 times
Welcome 5 times
```

```
[root@ip-172-31-80-171 ec2-user]# vi for_loop.bash
[root@ip-172-31-80-171 ec2-user]# chmod 755 for_loop.bash
[root@ip-172-31-80-171 ec2-user]# cat for_loop.bash
#!/bin/bash
for i in eat run jump play
do
echo See Imran $i
done


[root@ip-172-31-80-171 ec2-user]# ./for_loop.bash
See Imran eat
See Imran run
See Imran jump
See Imran play
```

```
[root@ip-172-31-80-171 ec2-user]# vi for_loop.bash
[root@ip-172-31-80-171 ec2-user]# chmod 755 for_loop.bash
[root@ip-172-31-80-171 ec2-user]# cat for_loop.bash
#!/bin/bash
for i in {1..5}
do
   touch $i
done


[root@ip-172-31-80-171 ec2-user]# ./for_loop.bash
[root@ip-172-31-80-171 ec2-user]# ll
total 72
-rw-r--r-- 1 root root    0 Jun  6 08:06 1
-rw-r--r-- 1 root root    0 Jun  6 08:06 2
-rw-r--r-- 1 root root    0 Jun  6 08:06 3
-rw-r--r-- 1 root root    0 Jun  6 08:06 4
-rw-r--r-- 1 root root    0 Jun  6 08:06 5
```

```
[root@ip-172-31-80-171 ec2-user]# vi for_loop.bash
[root@ip-172-31-80-171 ec2-user]# chmod 755 for_loop.bash
[root@ip-172-31-80-171 ec2-user]# cat for_loop.bash
#!/bin/bash
for i in {1..5}
do
   rm $i
done



[root@ip-172-31-80-171 ec2-user]# ./for_loop.bash
[root@ip-172-31-80-171 ec2-user]# ll
total 72
-rwxr-xr-x 1 root root  246 Jun  5 13:26 arrayex.sh
-rwxr-xr-x 1 root root  438 Jun  5 14:23 bacup_directory.sh
-rwxr-xr-x 1 root root 1056 Jun  6 05:08 calculator.sh
```

```
[root@ip-172-31-80-171 ec2-user]# vi for_loop.bash
[root@ip-172-31-80-171 ec2-user]# chmod 755 for_loop.bash
[root@ip-172-31-80-171 ec2-user]# cat for_loop.bash
#!/bin/bash
i=1
for username in `awk -F: '{print $1}' /etc/passwd`
do
 echo "Username $((i++)) : $username"
done

[root@ip-172-31-80-171 ec2-user]# ./for_loop.bash
Username 1 : root
Username 2 : bin
Username 3 : daemon
Username 4 : adm
Username 5 : lp
Username 6 : sync
Username 7 : shutdown
Username 8 : halt
Username 9 : mail
Username 10 : operator
Username 11 : games
```

2. Create a bash script to check if a directory is available or not

```
[root@ip-172-31-80-171 ec2-user]# vi check_dir.sh
[root@ip-172-31-80-171 ec2-user]# cat check_dir.sh
#!/bin/bash

# Check if the user passed a directory path as an argument
if [ -z "$1" ]; then
  echo "Usage: $0 /path/to/directory"
  exit 1
fi

# Check if the directory exists
if [ -d "$1" ]; then
  echo "Directory '$1' exists."
else
  echo "Directory '$1' does NOT exist."
fi

[root@ip-172-31-80-171 ec2-user]# chmod 755 check_dir.sh
[root@ip-172-31-80-171 ec2-user]# ./check_dir.sh
Usage: ./check_dir.sh /path/to/directory
[root@ip-172-31-80-171 ec2-user]# ^C
[root@ip-172-31-80-171 ec2-user]# ./check_dir.sh /path/to/directory
Directory '/path/to/directory' does NOT exist.
```

3. Create a bash script to create multiple files.

```
[root@ip-172-31-80-171 ec2-user]# vi create_files.sh
[root@ip-172-31-80-171 ec2-user]# cat create_files.sh
#!/bin/bash

# Check if at least one filename is provided
if [ "$#" -lt 1 ]; then
  echo "Usage: $0 filename1 [filename2 ... filenameN]"
  exit 1
fi

# Loop through all the provided filenames
for filename in "$@"; do
  # Create the file if it doesn't exist
  if [ -e "$filename" ]; then
    echo "File '$filename' already exists. Skipping."
  else
    touch "$filename"
    echo "File '$filename' created."
  fi
done

[root@ip-172-31-80-171 ec2-user]# chmod 755 create_files.sh
[root@ip-172-31-80-171 ec2-user]# ./create_files.sh
Usage: ./create_files.sh filename1 [filename2 ... filenameN]
```

4. Create a bash script to take a backup of a directory.

```
[root@ip-172-31-80-171 ec2-user]# vi bacup_directory.sh
[root@ip-172-31-80-171 ec2-user]# cat bacup_directory.sh
#!/bin/bash

# Check if directory is provided
if [ -z "$1" ]; then
  echo "Usage: $0 /path/to/directory"
  exit 1
fi

# Get directory name and current date-time
SOURCE_DIR="$1"
DIR_NAME=$(basename "$SOURCE_DIR")
TIMESTAMP=$(date +"%Y%m%d_%H%M%S")
BACKUP_NAME="${DIR_NAME}_backup_${TIMESTAMP}.tar.gz"

# Create backup
tar -czf "$BACKUP_NAME" -C "$(dirname "$SOURCE_DIR")" "$DIR_NAME"

# Confirm backup
echo "Backup created: $BACKUP_NAME"

[root@ip-172-31-80-171 ec2-user]# chmod 755 bacup_directory.sh
[root@ip-172-31-80-171 ec2-user]# ./bacup_directory.sh
Usage: ./bacup_directory.sh /path/to/directory
```

5. Create a bash script to install Nginx on an EC2 server.

```
[root@ip-172-31-80-171 ec2-user]# vi install_nginx.sh
[root@ip-172-31-80-171 ec2-user]# chmod 755 install_nginx.sh
[root@ip-172-31-80-171 ec2-user]# cat install_nginx.sh
#!/bin/bash

# Exit immediately if a command exits with a non-zero status
set -e

# Detect OS type
if [ -f /etc/os-release ]; then
  . /etc/os-release
else
  echo "Cannot detect OS type. Exiting."
  exit 1
fi

echo "Detected OS: $ID"

# Install Nginx based on OS
if [[ "$ID" == "amzn" ]]; then
  echo "Installing Nginx on Amazon Linux..."
  sudo yum update -y
  sudo amazon-linux-extras enable nginx1
  sudo yum install -y nginx
elif [[ "$ID" == "ubuntu" ]]; then
```

```
Now you can install:
 # yum clean metadata
 # yum install nginx
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Existing lock /var/run/yum.pid: another copy is running as pid 28106.
Another app is currently holding the yum lock; waiting for it to exit...
  The other application is: yum
    Memory : 192 M RSS (425 MB VSZ)
    Started: Fri Jun  6 02:54:49 2025 - 00:06 ago
    State  : Uninterruptible, pid: 28106
Another app is currently holding the yum lock; waiting for it to exit...
  The other application is: yum
    Memory : 208 M RSS (443 MB VSZ)
    Started: Fri Jun  6 02:54:49 2025 - 00:08 ago
    State  : Running, pid: 28106
amzn2extra-docker
amzn2extra-kernel-5.10
amzn2extra-nginx1
Package 1:nginx-1.26.3-1.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
Created symlink from /etc/systemd/system/multi-user.target.wants/nginx.service to /usr/lib/systemd/system/nginx.service.
   Active: active (running) since Fri 2025-05-30 07:49:50 UTC; 6 days ago
☑ Nginx installation and service start complete!
```

6. Create a bash script to install Apache Tomcat on an EC2 server.

```
[root@ip-172-31-80-171 ec2-user]# vi install_tomcat.sh
[root@ip-172-31-80-171 ec2-user]# chmod 755 install_tomcat.sh
[root@ip-172-31-80-171 ec2-user]# cat install_tomcat.sh
#!/bin/bash

# Exit on any error
set -e

# Variables
TOMCAT_VERSION=9.0.85
TOMCAT_USER=tomcat
INSTALL_DIR=/opt/tomcat
DOWNLOAD_URL="https://dlcdn.apache.org/tomcat/tomcat-9/v$TOMCAT_VERSION/bin/apac
he-tomcat-$TOMCAT_VERSION.tar.gz"

# Detect OS
if [ -f /etc/os-release ]; then
  . /etc/os-release
else
  echo "Cannot detect OS type. Exiting."
  exit 1
fi

echo "Detected OS: $ID"

# Install Java
if [[ "$ID" == "amzn" ]]; then
  sudo yum update -y
  sudo yum install -y java-1.8.0-openjdk
elif [[ "$ID" == "ubuntu" ]]; then
  sudo apt update -y
```

```
Installed:
  java-1.8.0-openjdk.x86_64 1:1.8.0.452.b09-1.amzn2.0.1

Dependency Installed:
  atk.x86_64 0:2.22.0-3.amzn2.0.2
  avahi-libs.x86_64 0:0.6.31-20.amzn2.0.7
  cairo.x86_64 0:1.15.12-4.amzn2
  copy-jdk-configs.noarch 0:3.3-10.amzn2
  cups-libs.x86_64 1:1.6.3-51.amzn2.0.5
  fribidi.x86_64 0:1.0.2-1.amzn2.1.2
  gdk-pixbuf2.x86_64 0:2.36.12-3.amzn2
  graphite2.x86_64 0:1.3.10-1.amzn2.0.2
  gtk-update-icon-cache.x86_64 0:3.22.30-3.amzn2.0.1
  gtk2.x86_64 0:2.24.31-1.amzn2.0.3
  harfbuzz.x86_64 0:1.7.5-2.amzn2.0.2
  hicolor-icon-theme.noarch 0:0.12-7.amzn2
  jasper-libs.x86_64 0:1.900.1-33.amzn2.0.1
```

7. Create a bash script to check if the Nginx service is running , and start it if not.

```
[root@ip-172-31-80-171 ec2-user]# vi check_nginx.sh
[root@ip-172-31-80-171 ec2-user]# cat check_nginx.sh
#!/bin/bash

# Function to check Nginx status
check_nginx() {
  systemctl is-active --quiet nginx
}

# Check if Nginx is running
if check_nginx; then
  echo "☑ Nginx is already running."
else
  echo "⚠ Nginx is not running. Attempting to start it..."
  sudo systemctl start nginx

  # Re-check after attempting to start
  if check_nginx; then
    echo "☑ Nginx started successfully."
  else
    echo "✗ Failed to start Nginx. Check logs with: sudo journalctl -u nginx"
  fi
fi

[root@ip-172-31-80-171 ec2-user]# chmod 755 check_nginx.sh
[root@ip-172-31-80-171 ec2-user]# ./check_nginx.sh
☑ Nginx is already running.
```

8. Create a bash script for a calculator.

```
[root@ip-172-31-80-171 ec2-user]# vi calculator.sh
[root@ip-172-31-80-171 ec2-user]# cat calculator.sh
#!/bin/bash

# Function to show menu
show_menu() {
  echo "Simple Calculator"
  echo "-----------------"
  echo "1. Addition"
  echo "2. Subtraction"
  echo "3. Multiplication"
  echo "4. Division"
  echo "5. Exit"
  echo -n "Choose an option [1-5]: "
}

# Loop until user chooses to exit
while true; do
  show_menu
  read choice

  if [ "$choice" -eq 5 ]; then
    echo "Exiting calculator. Bye!"
    break
  fi

  echo -n "Enter first number: "
```

```
[root@ip-172-31-80-171 ec2-user]# chmod 755 calculator.sh
[root@ip-172-31-80-171 ec2-user]# ./calculator.sh
Simple Calculator
------------------
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
Choose an option [1-5]: 1
Enter first number: 23
Enter second number: 21
Result: 23 + 21 = 44

Simple Calculator
------------------
1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Exit
```

9. Create a bash script to check if a directory exists , and create it if not.

```
[root@ip-172-31-80-171 ec2-user]# vi check_create_dir.sh
[root@ip-172-31-80-171 ec2-user]# chmod 755 check_create_dir.sh
[root@ip-172-31-80-171 ec2-user]# cat check_create_dir.sh
#!/bin/bash

# Check if directory path is passed
if [ -z "$1" ]; then
  echo "Usage: $0 /path/to/directory"
  exit 1
fi

DIR="$1"

# Check if the directory exists
if [ -d "$DIR" ]; then
  echo "☑ Directory already exists: $DIR"
else
  echo "🗁 Directory does not exist. Creating: $DIR"
  mkdir -p "$DIR"
  echo "☑ Directory created: $DIR"
fi


[root@ip-172-31-80-171 ec2-user]# ./check_create_dir.sh
Usage: ./check_create_dir.sh /path/to/directory
```

10. Create a bash script to delete the last 3 lines of a file.

```
[root@ip-172-31-80-171 ec2-user]# chmod 755 check_create_dir.sh
[root@ip-172-31-80-171 ec2-user]# ./check_create_dir.sh
Usage: ./check_create_dir.sh /path/to/directory
[root@ip-172-31-80-171 ec2-user]# cat delete_last_3_lines.sh
#!/bin/bash

# Check if file name is provided
if [ -z "$1" ]; then
  echo "Usage: $0 filename"
  exit 1
fi

FILE="$1"

# Check if file exists
if [ ! -f "$FILE" ]; then
  echo "✗ File not found: $FILE"
  exit 1
fi

# Create a backup first (optional but recommended)
cp "$FILE" "$FILE.bak"

# Delete last 3 lines and overwrite the original file
head -n -3 "$FILE" > temp_file && mv temp_file "$FILE"

echo "☑ Last 3 lines deleted from: $FILE"
```

1. April 5th Midnight
2. 5 th of Every November ,Jan ,June if it is a Thursday.
3. At 05 and 27th minutes of 9,10,11 hours everyday.
4. 34 min. of 9th hour on 15th Aug.
5. Every midnight

6. Every Weekend (Saturday night 11.59)
7. After every reboot

```
[root@ip-172-31-80-171 ec2-user]# sudo systemctl status crond
● crond.service - Command Scheduler
   Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled; vendor preset: enabled)
   Active: active (running) since Fri 2025-05-30 06:43:14 UTC; 1 weeks 0 days ago
 Main PID: 3163 (crond)
    Tasks: 1
   Memory: 800.0K
   CGroup: /system.slice/crond.service
           └─3163 /usr/sbin/crond -n

May 30 06:43:14 ip-172-31-80-171.ec2.internal systemd[1]: Started Command Scheduler.
May 30 06:43:14 ip-172-31-80-171.ec2.internal crond[3163]: (CRON) INFO (RANDOM_DELAY will be scaled with factor 67% if used.)
May 30 06:43:14 ip-172-31-80-171.ec2.internal crond[3163]: (CRON) INFO (running with inotify support)
Jun 06 09:45:01 ip-172-31-80-171.ec2.internal crond[3163]: (root) RELOAD (/var/spool/cron/root)
Jun 06 09:48:01 ip-172-31-80-171.ec2.internal crond[3163]: (root) RELOAD (/var/spool/cron/root)
[root@ip-172-31-80-171 ec2-user]#
```

```
#april 5th midnight
0 0 5 4 6

#5th of every november,jan,june if it is a thursday
0 0 5 11,1,6 4

#at 05 and 27th minutes of 9,10,11 hours everyday
27 9,10,11 5 0 *

#34min. of 9th hour on 15th aug.
34 9 15 8 5

#every midnight
0 0 * * *

#every weekend (saturday night 11.59)
59 23 * * 6

#after every reboot
@reboot
```