**TASK 1**

Create a script for the following task

Make 100 directories name test1 to 100

Each directory should have a file in the following manner:

•test1 should have file1,test2 should have file2………test100 should have file100

Each file should have the content in the following manner:

•file1 should have content 'This is file1',file2 should have content 'This is file2'……….file100 should have content 'This is file100'

**#!/bin/bash**

**rm -r test\***

**for (( i=1;i<=100;i++ ));**

**#for i in {1..10}**

**do**

**# echo "$i"**

**mkdir -p /home/ditiss-02/Desktop/test$i**

**touch /home/ditiss-02/Desktop/test$i/file$i**

**echo "This is file$i" > /home/ditiss-02/Desktop/test$i/file$i**

**done**

rm -r test\*

**TASK 2**

Write a script which greet Good Morning to user.

**#!/bin/bash**

**echo "Good Morning $(whoami)"**

**TASK 3**

\*Write a script which checks entered no is positive or not. If it is positive, it

checks that number is in range from 30 to 50 or not.

**#!/bin/bash**

**read -p "Enter a number:" num**

**if (( num>0 ));then**

**if (( num >= 30 && num <= 50 ));then**

**echo "Provided Number is positive and in between 30-50"**

**else**

**echo "Provided number is positive but not in between 30-50"**

**fi**

**else**

**echo "Provided Number is negative"**

**fi**

**TASK 4**

Write a calculator script.

**#!/bin/bash**

**add(){**

**result=$(( $1 + $2 ))**

**echo "${result}"**

**}**

**subtraction(){**

**result=$(( $1 - $2 ))**

**echo "${result}"**

**}**

**multiplication(){**

**result=$(( $1 \* $2 ))**

**echo "${result}"**

**}**

**division(){**

**result=$(( $1 / $2 ))**

**echo "${result}"**

**}**

**power(){**

**result=$(( $1 \*\* $2 ))**

**echo "${result}"**

**}**

**module(){**

**result=$(( $1 % $2 ))**

**echo "${result}"**

**}**

**echo "1. Addition"**

**echo "2. Subtraction"**

**echo "3. Multiplication"**

**echo "4. Division"**

**echo "5. Power"**

**echo "6. Module"**

**read -p "Operation(1|2|3|4|5|6): " operation**

**read -p "First Number: " n1**

**read -p "Second Number: " n2**

**case $operation in**

**1) add $n1 $n2;;**

**2) subtraction $n1 $n2;;**

**3) multiplication $n1 $n2;;**

**4)**

**if [ $n2 -eq 0 ]; then**

**echo "Error: Division by zero!"**

**else**

**division $n1 $n2**

**fi;;**

**5) power $n1 $n2;;**

**6)**

**if [ $n2 -eq 0 ]; then**

**echo "Error: Division by zero!"**

**else**

**module $n1 $n2**

**fi;;**

**\*) echo "Invalid operation";;**

**esac**

**TASK 5**

Create a script that asks for a path and tell if the entered path is a file or directory or

doesn’t exist. If File or directory exists print numeric file permission.

**#!/bin/bash**

**#read -p "Enter the path: " p**

**if [ -e "$p" ]; then**

**if [ -f "$p" ] || [ -L "$p" ] || [ -b "$p" ] || [ -c "$p" ] || [ -p "$p" ] || [ -S "$p" ]; then**

**echo "$1 is a file."**

**echo "Numeric permissions: $(stat -c '%a' "$p")"**

**elif [ -d "$p" ]; then**

**echo "$p is a directory."**

**echo "Numeric permissions: $(stat -c '%a' "$p")"**

**else**

**echo "$p exists but is neither a file nor a directory."**

**fi**

**else**

**echo "$p does not exist."**

**fi**

**TASK 6**

Write a script which check entered user exists or not. If exists, display uid, gid and default

Shell.

**#!/bin/bash**

**read -p "Enter username: " username**

**if getent passwd | grep -q "^$username:"; then**

**uid=$(getent passwd "$username" | cut -d ":" -f 3)**

**gid=$(getent passwd "$username" | cut -d ":" -f 4)**

**shell=$(getent passwd "$username" | cut -d ":" -f 7)**

**echo "User $username exists:"**

**echo "UID: $uid"**

**echo "GID: $gid"**

**echo "Default shell: $shell"**

**else**

**echo "User $username does not exist."**

**fi**

**TASK 7**

Create script which create 50 users named as 'cdac\_test\_1' to 'cdac\_test\_50' with password as username. And create a file for alternate users starting from 1 in /tmp from root user name as 'username\_flag.txt' and the data should be "username:CDAC'

**#!/bin/bash**

**for (( i=1;i<=50;i++ )); do**

**username="cdac\_test\_$i"**

**useradd -s /bin/bash ${username}**

**echo -e "$username\n$username" | passwd $username**

**if [[ $((i% 2)) -eq 0 ]]; then**

**echo "$username:CDAC" > "/tmp/${username}\_flag.txt"**

**fi**

done

**TASK 8**

Display following menu to user:-

1. Line count --- accept a file name from user, and display line count, if file exists and file has read permission. Otherwise display appropriate messages.

2. Write data to file----- accept file name from user, accept a string from user, check whether file exists and has write permissions then append the data, otherwise create new file and add data in the file.

3. Search pattern ---- accept file name from user, accept a search-string from user, check whether file exists and has read permissions then display all lines which has given search-string , if search-string not found then display appropriate message.

4. Create directory. Accept a directory name from user, if the directory does not exists then create new directory otherwise display message directory exists

5. exit

**#!/bin/bash**

**while true; do**

**echo "Menu:"**

**echo "1. Line count"**

**echo "2. Write data to file"**

**echo "3. Search pattern"**

**echo "4. Create directory"**

**echo "5. Exit"**

**read -p "Enter your choice: " choice**

**case $choice in**

**1)**

**read -p "Enter the file name: " file**

**if [[ -f $file && -r $file ]]; then**

**count=$(wc -l < "$file")**

**echo "Total number of lines in $file: $count"**

**else**

**echo "File not found or permission denied."**

**fi**

**;;**

**2)**

**read -p "Enter the file name: " fname**

**read -p "Enter the content to be added: " string**

**if [[ -w $fname ]]; then**

**echo "${string}" >> "${fname}"**

**echo "Content appended in the file ${fname}"**

**else**

**echo "${string}" > "${fname}"**

**echo "Content added after creating a new file ${fname}"**

**fi**

**;;**

**3)**

**read -p "Enter the file name: " fname**

**read -p "Enter the string to search: " string**

**if [[ -f ${fname} && -r ${fname} ]]; then**

**if grep -q "${string}" "${fname}"; then**

**grep -n "${string}" "${fname}"**

**else**

**echo "No line found containing the string: ${string}"**

**fi**

**else**

**echo "File not found or permission denied."**

**fi**

**;;**

**4)**

**read -p "Enter the directory name: " dir**

**if [ ! -d "$dir" ]; then**

**echo "Directory '$dir' does not exist"**

**echo "Creating ${dir} directory"**

**mkdir -p "${dir}"**

**else**

**echo "Directory ${dir} already exists"**

**fi**

**;;**

**5)**

**echo "Exiting... "**

**exit 0**

**;;**

**\*)**

**echo "Invalid choice."**

**;;**

**esac**

**done**

**TASK 10**

Create a script /root/myscript.sh that should print "redhat" when user input "fedora" as argument, & vice-versa, it should print "redhat|fedora" without argument.

**#!/bin/bash**

**if [ -z "$1" ]; then**

**echo "redhat|fedora"**

**else**

**option=$(echo "$1" | tr '[:upper:]' '[:lower:]')**

**case "$option" in**

**"fedora")**

**echo "redhat"**

**;;**

**"redhat")**

**echo "fedora"**

**;;**

**\*)**

**echo "Invalid option. Please provide 'fedora' or 'redhat'."**

**;;**

**esac**

**fi**

**TASK 11**

Write a menu based script which displays the following options :

1. Make a file.

2. Display contents

3. Copy the file

4. Rename the file

5. Delete the file

6. Exit

Enter your option:

If the user selects option 1, accept a file name from the user. If the file exists,

then display an error message pass the control to the menu. If the file does not exist,

then allow the user to enter some data. Pressing would save the contents and display the menu.

If the user selects option 2, then accept a file name from the user. If the file exists,

then display the contents of the file. If the file does not exist, then display suitable error message.

After this process, display the menu to accept another option.

Selecting Option 3 allows the user to accept

the source file and target file. If the source file exists and is readable,

then accept the target file name.

If the sourcefile does not exist, then display suitable error message.

If the target file does not exist,

then copy the contents of the source file to the target file. If the target file exists,

then display suitable

message and go back to the menu.

Option 4 is similar to option 3 but rename the file instead of copying.

Selecting option 5 allows the user to enter a file name. If the file exists, then check to see if it is writable.

If so, then delete the file with confirmation from the user.

If the file does not exist, then display suitable error message.

**#!/bin/bash**

**while true; do**

**read -p "1. Make a file.**

**2. Display contents**

**3. Copy the file**

**4. Rename the file**

**5. Delete the file**

**6. Exit**

**Enter your option:" option**

**case "$option" in**

**1)**

**read -p "Enter the file name to be created:" fname**

**if [[ -e ${fname} ]];then**

**echo "$fname file already exists"**

**else**

**echo "Enter data to add to the file"**

**cat > $fname**

**echo "File ${fname} created."**

**fi**

**ls**

**;;**

**2)**

**read -p "Enter the file name:" fname**

**if [[ -e ${fname} ]];then**

**cat ${fname}**

**else**

**echo "File ${ifname} does not exists."**

**fi**

**ls**

**;;**

**3) read -p "Enter the file name:" fname**

**if [ -f "$fname" ]; then**

**read -p "Enter target file name:" target**

**if [ -e "$target" ]; then**

**echo "Error: Target file '$target' already exists."**

**else**

**cp "$fname" "$target"**

**echo "File copied from '$fname' to '$target'."**

**fi**

**else**

**echo "Error: Source file '$fname' not found."**

**fi**

**ls**

**;;**

**4) read -p "Enter the file name:" fname**

**if [ -f "$fname" ]; then**

**read -p "Enter new name:" newName**

**if [ -e "$newName" ]; then**

**echo "Error'$newName' already exists."**

**else**

**mv "$fname" "$newName"**

**echo "File renamed from '$fname' to '$newName'."**

**fi**

**else**

**echo "Error: File '$fname' not found."**

**fi**

**ls**

**;;**

**5) read -p "Enter the file name:" fname**

**if [ -f "$fname" ]; then**

**read -p "Do you want to remove this file(y|n):" confirm**

**if [ "$confirm" = "y" ]; then**

**rm "$fname"**

**echo "File '$fname' deleted."**

**else**

**echo "File deletion cancelled."**

**fi**

**else**

**echo "Error: File '$fname' not found."**

**fi**

**ls**

**;;**

**6) echo "Exiting..."**

**exit 0**

**;;**

**\*)**

**echo "Invalid option."**

**;;**

**esac**

**done**

**TASK 12**

Create a script which take backup of /home/user/data directory and

save backup to /backup directory.

**#!/bin/bash**

**# Source directory to backup**

**source\_dir="/home/aryan/data"**

**# Destination directory for backup**

**backup\_dir="/backup"**

**# Create backup filename with timestamp**

**backup\_file="${backup\_dir}/data\_backup\_$(date +'%Y%m%d\_%H%M%S').tar.gz"**

**# Check if source directory exists**

**if [ ! -d "$source\_dir" ]; then**

**echo "Source directory '$source\_dir' does not exist."**

**exit 1**

**fi**

**# Check if destination directory exists, else create it**

**if [ ! -d "$backup\_dir" ]; then**

**echo "Creating backup directory..."**

**sudo mkdir -p "$backup\_dir"**

**sudo chown "$(whoami)" "$backup\_dir"**

**fi**

**# Check if destination directory is writable**

**if [ ! -w "$backup\_dir" ]; then**

**echo "Destination directory '$backup\_dir' is not writable."**

**exit 1**

**fi**

**# Create tar.gz archive of source directory**

**tar czf "$backup\_file" -C "$(dirname "$source\_dir")" "$(basename "$source\_dir")"**

**# Check if backup was successful**

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

**echo "Backup created successfully: $backup\_file"**

**else**

**echo "Backup failed."**

**fi**

**TASK 13**

Write a shell script which display system information in following format.

$ bash sys\_info

Hostname: localhost.localdomain

OS: CentOS Linux

Kernal: 3.10.0-1160.76.1.el7.x86\_64

Processor Name: AMD Ryzen 7 5700U with Radeon Graphics

Architecture: x86\_64

Memory: 1.8G

Swap Memory: 2.0G

Uptime: 22:16:13

**echo "Hostname: $(hostname)"**

**echo "OS: $(cat /etc/\*-release | grep "^PRETTY\_NAME=" | cut -d= -f2 | tr -d '"' | cut -d' ' -f1-2)"**

**echo "Kernel: $(uname -r)"**

**echo "Processor Name: $(cat /proc/cpuinfo | grep "model name" | cut -d: -f 2 | head -1)"**

**echo "Architecture: $(uname -m)"**

**echo "Memory: $(free -h | grep "Mem:" | tr -s " " | cut -d " " -f 2)"**

**echo "Swap Memory: $(free -h | grep "Swap" | tr -s " " | cut -d " " -f 2)"**

**echo "Uptime: $(uptime | cut -d " " -f4- | cut -d "," -f1)"**

**TASK 14**

Write a shell script that accepts a file name, starting and ending line numbers as arguments

and displays all the lines between the given line numbers. If file does not exist, display proper

message.

Create a user menu driven program:

1: user addition

2: user deletion

3: user info

4: check numeric permission of a file or directory

**#!/bin/bash**

**addUser(){**

**read -p "Enter user:" username**

**echo "ashu" | sudo useradd ${username}**

**}**

**delUser(){**

**read -p "Enter user:" username**

**echo "ashu" | sudo userdel -r ${username}**

**}**

**infoUser(){**

**read -p "Enter user:" username**

**id ${username}**

**}**

**checkPermission() {**

**read -p "Enter file or directory path: " filepath**

**stat -c "%a" $filepath**

**}**

**while true;do**

**read -p "1: user addition**

**2: user deletion**

**3: user info**

**4: check numeric permission of a file or directory**

**5. exit**

**Enter your choice:" option**

**case $option in**

**1)**

**addUser**

**;;**

**2)**

**delUser**

**;;**

**3)**

**infoUser**

**;;**

**4)**

**checkPermission**

**;;**

**5)**

**exit 0**

**;;**

**\*)**

**echo "Invalid option"**

**;;**

**esac**

**done**

**TASK 15**

Create a script that shows the status of Selinux.If selinux is enforced,

then prompt the user to choose whether to make selinux permissive or not.

If so, selinux status is changed to permissive; else, leave.

**#!/bin/bash**

**checkStatus() {**

**seStatus=$(sestatus | grep "SELinux status" | tr -d " " | cut -d ":" -f 2)**

**echo "SELinux status: $seStatus"**

**}**

**changeStatus() {**

**echo "Changing SELinux to permissive..."**

**sudo setenforce 0**

**echo "SELinux status changed to permissive."**

**checkStatus**

**}**

**checkStatus**

**if [ "$seStatus" == "enabled" ] && [ "$(getenforce)" == "Enforcing" ]; then**

**read -p "Do you want to change it to permissive? (y|n)): " choice**

**case $choice in**

**y|Y)**

**changeStatus**

**;;**

**n|N)**

**echo "SELinux status unchanged."**

**;;**

**\*)**

**echo "Invalid choice."**

**;;**

**esac**

**else**

**echo "SELinux is already permissive."**

**fi**