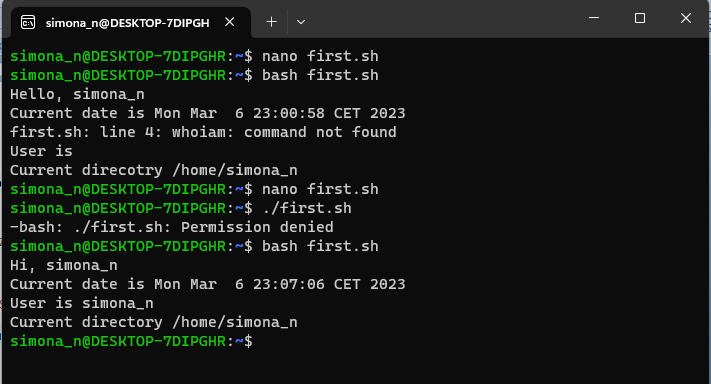
Exercise: Shell scripts

1. Write a shell script to get the current date, time, username and current working directory.



*The picture is showing the result after running the script. The commands below are written in the script*

#!/bin/bash

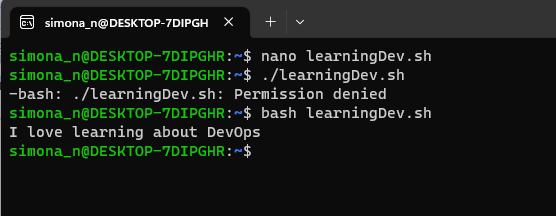
echo "Hi, $LOGNAME"

echo "Current date is `date`"

echo "User is `whoami`"

echo "Current directory `pwd`"

2. Write a shell script that prints “I love learning about DevOps” on the screen. Message should be a variable.

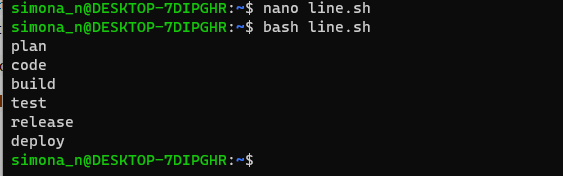


#!/bin/bash

MSG='I love learning about DevOps'

echo $MSG

3. Write a shell script that displays “plan code build test release deploy” on the screen with each appearing on a separate line.



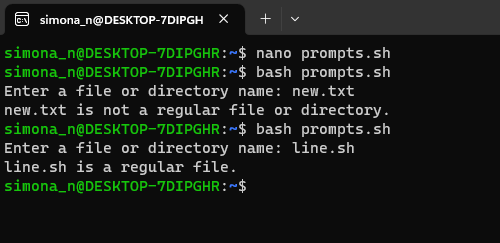
#!/bin/bash

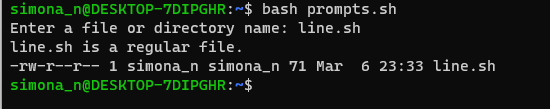
tr ' ' '\n' <<< "plan code build test release deploy"

*another command that gives the same result is*:

sed "s/ /\n/g" <<< "plan code build test release deploy"

4. Write a shell script that prompts the user for a name of a file or directory and reports if it is a regular file, a directory, or another type of file. Also perform a ls command against the file or directory with the long listing option.





#!/bin/bash

# Prompt the user for a file or directory name

read -p "Enter a file or directory name: " name

# Check if the name is a regular file

if [ -f "$name" ]; then

 echo "$name is a regular file."

# Check if the name is a directory

elif [ -d "$name" ]; then

 echo "$name is a directory."

# Otherwise, the name is of another type

else

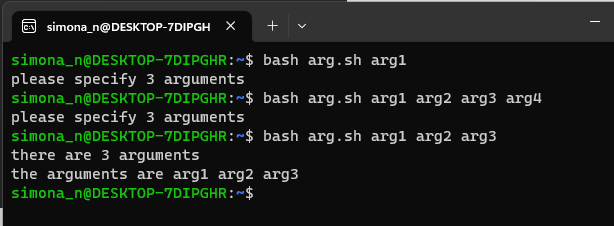
 echo "$name is not a regular file or directory."

fi

ls -l $name

*This script uses the -f and -d options of the [ (test) built-in command to check if the name is a regular file or a directory*

5. Use arguments in a script. Total number of arguments should be three.



#!/bin/bash

if [ $# -ne 3 ]; then

echo "please specify 3 arguments"

else

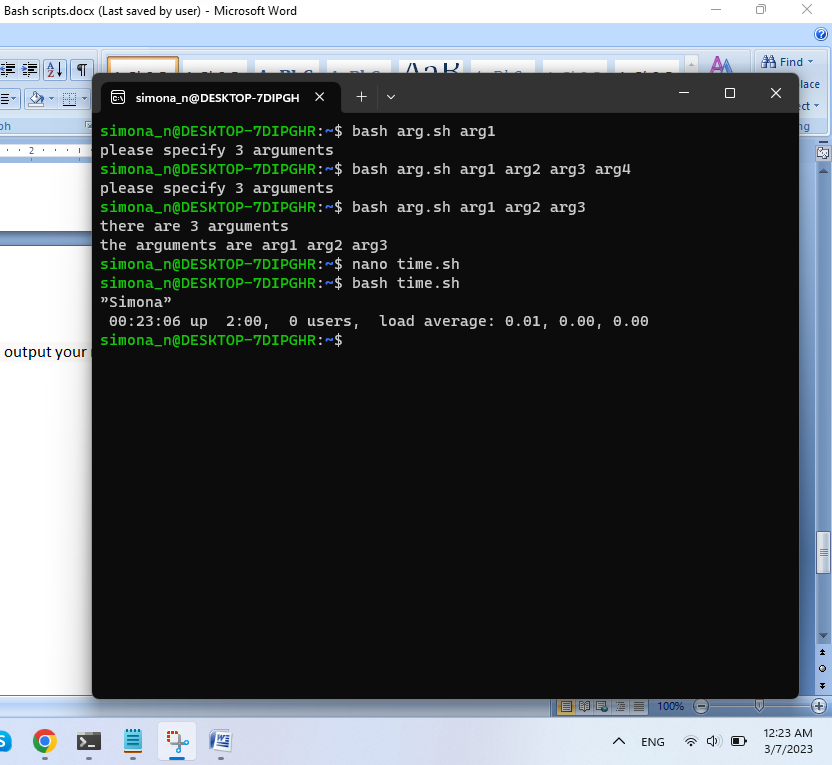
echo "there are 3 arguments"

echo "the arguments are $@"

fi

The script will only proceed if it detects there are a total of three arguments. –ne- . checks that the thing on the left ($?) is "not equal" to "zero", $# stores the total number of arguments and $@" Stores all the arguments that were entered on the command line, individually quoted

6. Write a script that till output your name out of a variable and will display the server uptime



#!/bin/bash

name=”Simona”

echo $name

uptime