Read Chapter 11 & 12. Run all examples in chapter 11 and 12. Read Chapter 13 because we will cover loops next week to prepare for the holiday weekend that is to follow.

1. Finish the lab script.

Add another elif statement to check for symbolic link. If file is a symbolic link, then show me the original file

that it points to.

Example output example:

```
calfile-link is a symbolic link.
The original file for calfile-link is : calfile
The size of calfile is : 453 bytes.
The calfile is owned by: <owner name>
```

Make sure it runs without errors.

```
#!/bin/bash
#......
# This script will check the permissions of any directory
# We would want to check to see that the directory exists first
#
echo -n “Please select file type you want to check: ”
read filetype

if [ -d $filetype]
```

```
then echo “$filetype is a directory”
    echo -n “Do you want to see the permissions of $filetype? ”
    read answer
        if [ $answer = “yes” ]
            then
                echo “Doing listing of : $filetype”
                ls -ld $filetype
            fi
        elif [ -f $filetype ] && [ ! -L $filetype ]
        then
            echo “$filetype is a file”
            echo -n “Do you want to see the permissions of: $filetype? ”
            read answer
                if [ $answer = “yes” ]
                    then
                        echo “Doing listing of: “$filetype”
                        ls -lh $filetype
                fi
        elif [ -L $filetype ]
        then
            echo “$filetype is a Symbolic link”
            echo -n “Getting the original file for $filetype! ”
            original_file=$(readlink -f $filetype)
            echo “Original file for :$filetype is : $original_file”
        else

            echo “$filetype is not a file or directory!”

    fi
```

2. Comment every line in the scrip bellow and tell what each line is doing. Look up command in the man pages if you don't understand it. The script name is 'myargs'. Copy the script and run it on your system and give a detail comment of what each line is doing.

```
--------------------------------------------------

#!/bin/bash

# Script name: myargs

#

# Date:

# Name:


# Script to test command line arguments

if [ $# == 0 ] ; then                  # execute next three lines if no command line parameters are input

  echo "Usage:  $(basename $0)  arg1 arg2 ... argn" \         # displays function usage directions

  1>&2            # redirects standard output to standard error

  exit 1            # returns exit status 1 (general errors)

fi          # ends if statement

echo "The name of this script is $0 ."            # displays the script’s name (myargs)

echo "The arguments are $* "            # displays all the entered arguments

echo "The first argument is $1"         #displays the first argument

echo "The second argument is $2"    #displays the second argument
echo "The number of arguments is $#"         # displays the number of arguments

previous_args=$*        # assigns all entered arguments to the variable “previous_args”

set niel khail nobo       # sets the positional parameters to “niel, khalil, nobo”

echo "All the positional parameters are $*"    # displays the new positional parameters (niel, khalil, nobo)

echo "The number of positional parameters is $#"    # displays the number of positional parameters (3)

echo $previous_args  # displays the value of previous arguments

set --    $          # doesn’t set anything to positional parameters

echo "Current args after -- are :  $*"  # displays the current arguments (nothing)

set $previous_args     #  sets the positional parameters to the value of $previous_args

echo $*            # displays all entered arguments (previous_args)
```

----

3. Copy the script and run it on your system and give a detail comment of what each line is doing.

```
#!/bin/bash

# Scriptname: tellage

#

read -p"How old are you? "     # displays the prompt (“How old are you?”) and accept one input line

age=$REPLY             # takes the input value and assign it to the “age” variable
```

```
if (( age < 0 || age > 120 ))      # executes the below lines until the fi if the age value is less than 0 or greater than 120

then

    echo "You are not a real person! "          # displays "You are not a real person! "

    exit 1         # returns exit status 1 (general errors)

fi



if (( age >= 0 && age < 13 ))  # executes the line after “then” if the age is greater than or equal to 0 and less than 13

then

    echo "You still have some of the best years of your life ahead."# displays "You still have some of the best years of your life ahead."

elif (( age > 12 && age < 20 ))           # executes the next line after the “then” if not caught by the previous if-statement, and if the age is greater than 12 and less than 20

then

    echo "Important years to learn"    # displays "Important years to learn"

elif (( age >= 20 && age < 30 ))         # executes the next line after the “then” if not caught by the previous if-statement, and if the age is greater than or equal to 20 and less than 30

then

    echo "Time to find a potential mate!!"      # displays "Time to find a potential mate!!"

elif (( age >= 30 && age < 40 ))         # executes the next line after the “then” if not caught by the previous if-statement, and if the age is greater than or equal to 30 and less than 40

then

    echo "You are probably changing diapers"         # displays "You are probably changing diapers"

else     # executes the next line if not caught by any of the previous if-statements

    echo "Sorry I asked"        # displays "Sorry I asked"

fi         # ends if statement
```
