Operating System (CS301)

Assignment - 4

**U19CS012**

1) Write a shell script which takes **filename** as argument and checks whether file is *regular file, directory, block special file, character special file, named pipe, symbolic link, socket, ~~device file~~* *[Block & Character are Device Files]* etc.

**Script:**

*# If the Number of Arguments are Not Equal to 1, [Invalid Input]*

*if* [ $# -ne 1 ]; *then*

    echo "Right usage: ./q1.sh <filename>"

    exit 2

*fi*

*# Take File Name from Argument 1*

filename=$1

*# Display Directory Information Using "ls -ld" and redirect it to temp.txt*

ls -ld $filename > temp.txt

*# Let "file" hold that temp.txt content*

file=temp.txt

*# All Details about that File [Modes Author Date Create Name Size]*

res=$(cat "$file")

*# cut command prints first character of each line from the file.*

*# -rwxrwxrwx 1 bhagya bhagya 187 Aug 31 15:52 regular\_file.txt*

*# - <- DASH*

type=$(echo $res | cut -c1-1)

*case* $type *in*

    "-") echo "It is a Regular File." ;;

    "d") echo "It is a Directory." ;;

    "c") echo "It is a Character Device file" ;;

    "l") echo "It is a Symbolic Link." ;;

    "s") echo "It is a Local Socket file." ;;

    "b") echo "It is a Block Device file." ;;

    "p") echo "It is a Named Pipe." ;;

    \*) echo "It is not a Valid Type." ;;

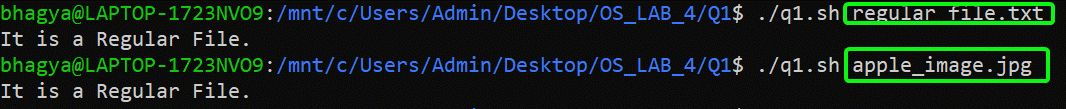
*esac*

*# Delete the temp File*

rm temp.txt

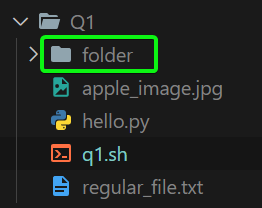
**Output:**

Regular File



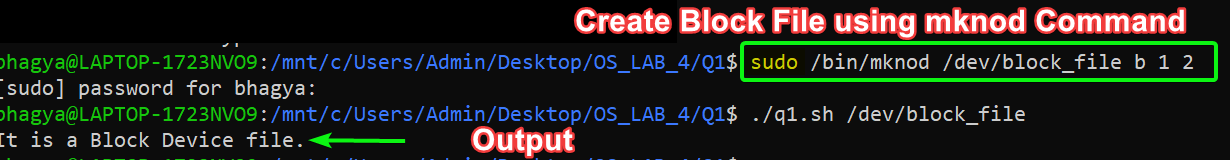
Directory





Block Special File

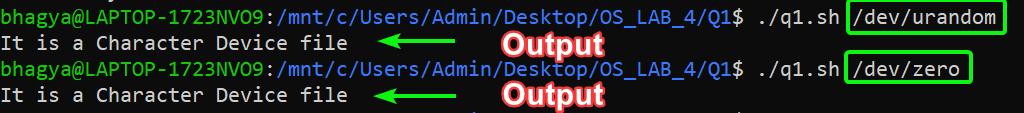
Create a Block File “**block\_file**” using **mknod** Command [Sudo Mode – Since Operation Not Permitted in Normal User Mode]



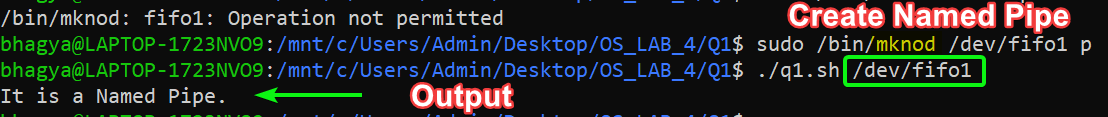
[IBM Documentation - <https://www.ibm.com/docs/en/aix/7.2?topic=m-mknod-command>]

Character Special File



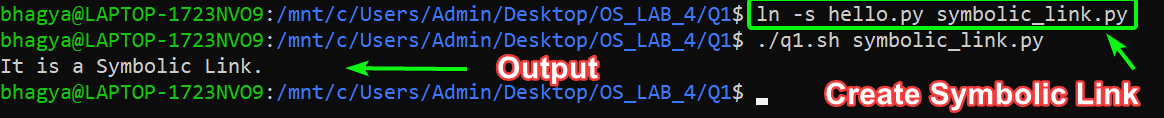


Named Pipe



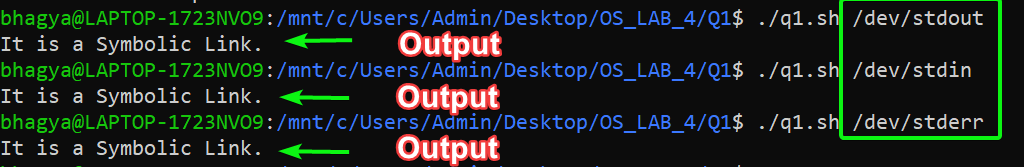
Symbolic Link

Created Own Symbolic File “**symbolic\_link.py**” using “ln – s” Command



[Wikipedia - <https://en.wikipedia.org/wiki/Ln_(Unix)>]

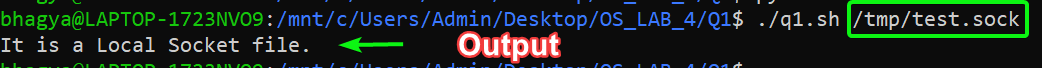
Default Symbolic Files

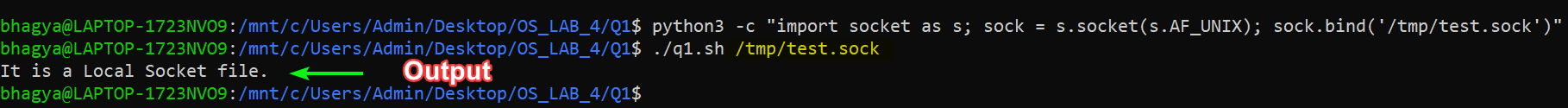


Socket

Created Own Socket File “**test.sock**”







[StackOverflow - <https://serverfault.com/questions/358866/create-unix-named-socket-from-the-command-line/358979>]

Device File

[Character and Block are Two Types of Device Files Discussed Above]

2) Write a shell script which will take file name as argument and check whether the file name is a directory or not and then proceed further only if it is a directory, else give usage message.

The script should then print in the tabular format, name of each sub-directory (within the argument directory) and a count of the number of top level files in that sub-directory.

[Modify the program to work with multiple numbers of arguments, too.]

**Script:**

*if* [ $# -eq 0 ]; *then*

    echo "Right usage: ./q2.sh [<dirs>,..]"

    exit 2

*fi*

*for* filename *in* "$@";

*do*

*# If the Given FileName is Directory*

*if* [ -d "$filename" ];

*then*

*# Copy the Contents of File in q2.txt [STEP 1]*

        find $filename -depth -maxdepth 1 > q2.txt

*# command1 && command2 -> command2 will execute if command1 has executed successfully.*

*# If "count.txt" Exist, then Remove it!*

        [ -f "count.txt" ] && rm count.txt

*# If "final.txt" Exist, then Remove it!*

        [ -f "final.txt" ] && rm final.txt

        echo "List of Sub-directories & Count of top level files in $filename"

        echo "Directory,Count" >>count.txt

*# Read all the Files in Given Folder  {Stored in q2.txt From Step 1}*

*while* read line;

*do*

*if* [[ $filename != $line ]]; *then*

*# It will Go inside $line Folder and Count all the Files in it*

                count=0

*for* entry *in* "$line"/\*;

*do*

*# Increment the Count, If it is File*

                    [ -f "$entry" ] && count=$(($count + 1))

*done*

*# Relative Path of File*

                file=$(echo basename $line)

                new="$file,$count"

*# Redirect it to "count.txt" File*

                $new >> count.txt

*fi*

*done* < q2.txt

*# Method to Read File {count.txt} Line by Line*

*while* IFS=, read -r a b;

*do*

*# File Name & Count redirected to "final.txt"*

            echo "$a $b" >>final.txt;

*done* < count.txt

*# Print in Tabular Form*

        awk '{printf "|%-30s|%20s|\n",$1,$2}' final.txt

        echo ""

*else*

        echo "$fileName is Not A Directory!"

*fi*

*done*

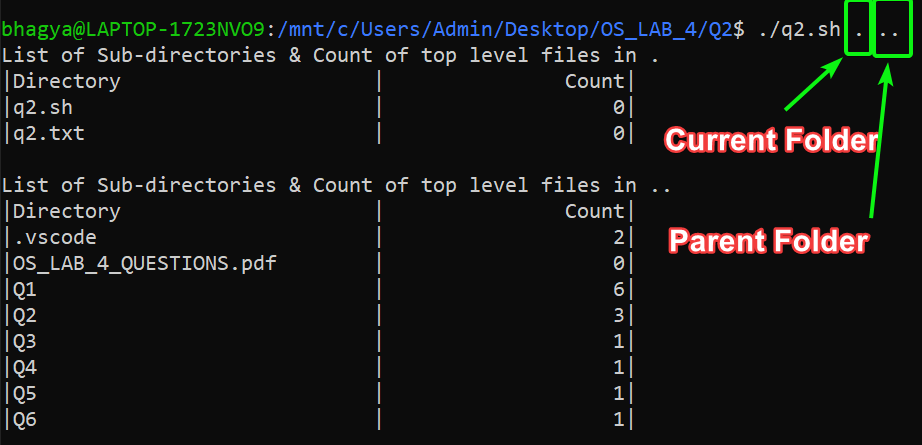
*# Remove all the Text Files Created*

rm \*.txt

*# >,>>, < (Redirection) -> Redirects the output of a command or a group of commands to a file or stream.*

*# 1.) https://www.cyberciti.biz/faq/unix-howto-read-line-by-line-from-file/*

**Output:**



3) Write a script that will search for a specific word in all the files in the current dictionary and then prompt with the file name in which word is found.

**Script:**

*# Check for Invalid Input*

*if* [ $# -ne 1 ]; *then*

    echo "Right Usage: ./q3.sh <word>"

    exit 2

*fi*

*# Get all Files in files*

files=$(find . -type f)

*# Get the Word as Argument*

word=$1

*for* file *in* $files;

*do*

    res=$(grep -w $word $file)

*# -n -> string {res} is not null*

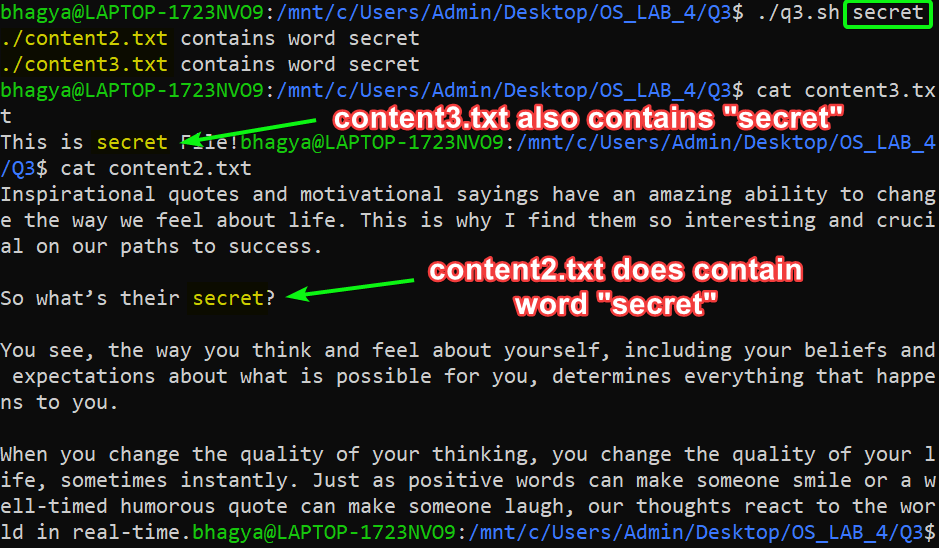
*if* [ -n "$res" ]; *then*

        echo "$file contains word $word"

*fi*

*done*

**Output:**



4) Write a script to print only the number of executable files in each sub-dir of the argument directory specified.

**Script:**

*if* [ $# -ne 1 ]; *then*

    echo "Right usage: ./q4.sh <dir>"

    exit 2

*fi*

*# Take Directory Name in Variable "name" from Command Line Arguments*

name=$1

echo "The Executable Files are : "

find $name -executable -type f

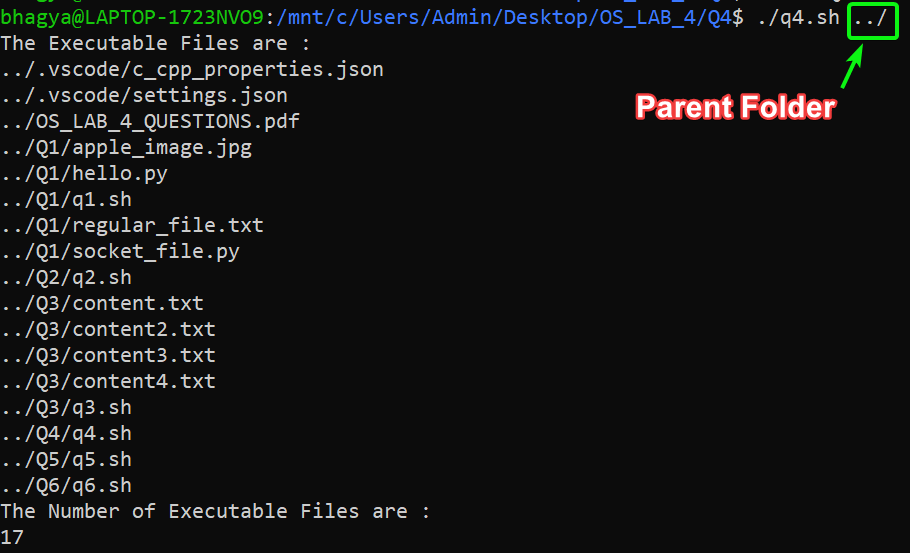
echo "The Number of Executable Files are : "

find $name -executable -type f | wc -l

*# References*

*# 1.) https://superuser.com/questions/38981/how-can-i-find-only-the-executable-files-under-a-certain-directory-in-linux*

**Output:**



Yep - because the NTFS file system doesn't support the Linux permission system, Linux assumes all text files on NTFS files systems are executable, just in case they are supposed to be.

5) Write a non-interactive script that takes in any no. of directory name as argument and calculates total no. of blocks of disk space occupied by the ordinary files in all the directories.

**Script:**

space=0

*for* arg *in* "$@";

*do*

*# If Argument is Directory*

*if* [ -d $arg ]; *then*

*# Get all the Files in res*

        res=$(find $arg -type f)

*# For Each File in Files Available*

*for* file *in* $res; *do*

*# Use Long Listing Format and List Only Directories*

            ls -ld $file >tmp.txt

            fileNew=tmp.txt

*# Copy All the Contents in res*

            res=$(cat "$fileNew")

*# Get the First Char in Each Line*

            type=$(echo $res | cut -c1-1)

*# If it is Regular File*

*if* [ $type == "-" ]; *then*

*# du Command -> Disk Utilization {Summarize and Produce a Grand Total}*

*# size=$(du -sc $file | tail -n 1 | cut -c1-1) [WRONG]*

*# cut -c1-1 -> Will NOT WORK, Since Number Can be Double Digit or More*

                size=$(du -sc $file | tail -n 1 | grep -o -E '[0-9]+')

*# echo $size*

                space=$(($space + $size))

*fi*

*done*

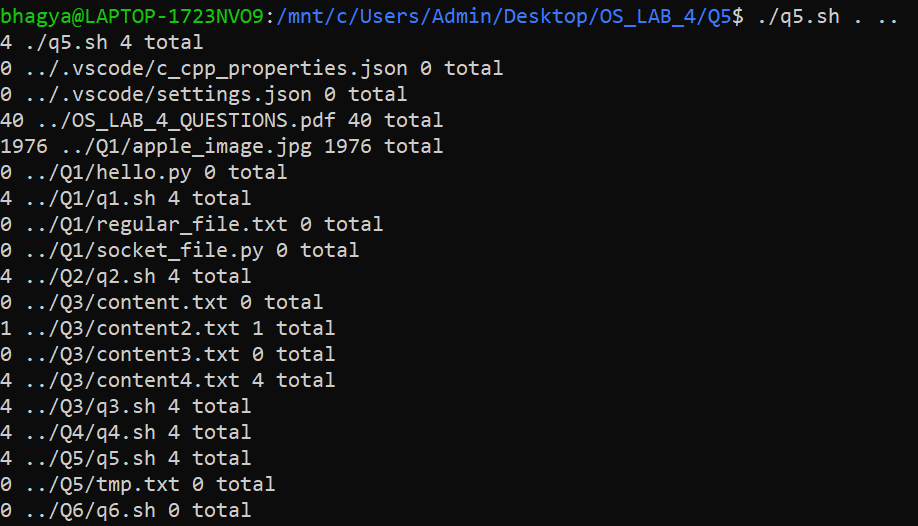
*fi*

*done*

rm tmp.txt

echo "Disk Space Used : $space"

**Output:**





6) Write a shell script file ~~named exercise2.sh~~ that makes a list of files in your home directory that were changed less than 24 hours ago, but leave out directories.

**Script:**

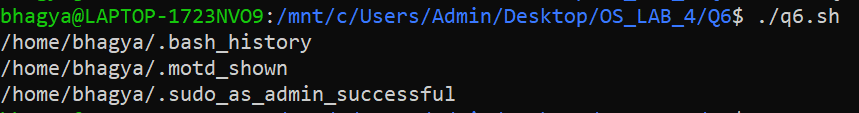
find ~ -mtime -1 -type f -maxdepth 1 2> /dev/null

*# References:*

*# 1.) https://www.thegeekstuff.com/2009/06/15-practical-unix-linux-find-command-examples-part-2/*

*# 2.) https://askubuntu.com/questions/350208/what-does-2-dev-null-mean*

**Output:**



SUBMITTED BY:

**U19CS012**

BHAGYA VINOD RANA