

Operating systems

(MOD003218)



2023 MOD003218 TRI2 F01CAM

**SID: \_\_\_\_\_\_\_**

Table of Contents

[Part 1: Scripting Task 2](#_Toc163249994)

[Program Flow Explanation (Pseudocode) 2](#_Toc163249995)

[Code 3](#_Toc163249996)

[MS-DOS Batch 3](#_Toc163249997)

[Linux Bash Script 6](#_Toc163249998)

[Screenshots 9](#_Toc163249999)

[MS-DOS Batch Screenshots 9](#_Toc163250000)

[Linux Bash Script Screenshots 11](#_Toc163250001)

[Part 2: Research Questions 13](#_Toc163250002)

[Question 1 13](#_Toc163250003)

[Question 2 15](#_Toc163250004)

[Question 3 16](#_Toc163250005)

[Question 4 18](#_Toc163250006)

[Question 5 19](#_Toc163250007)

[Bibliography 23](#_Toc163250008)

# Part 1: Scripting Task

## Program Flow Explanation (Pseudocode)

If more than one parameter Then

Display “Too many parameters”

Exit program

If no parameter Then

While True

Display Menu

Input choice

If choice != 1 or choice != 2 Then

Display “Invalid choice”

Else if choice = 2 Then

Exit program

Else if choice = 1 Then

Input domain

Find and print IP against domain

filename = parameter

If filename not existing Then

Display “File does not exist”

Exit program

If file is not text file Then

Display “File is not text file”  
 Exit program

While line in file

Display line

Get ip against line

Store ip in file

Extract only ip form file

Display ip

## Code

### MS-DOS Batch

@ECHO OFF

::Case where there are no input parameters from command line

IF "%1" == "" GOTO :NOFILE

::Case where there are 2 or more input parameters form command line

IF NOT "%2" == "" GOTO :EXTRAPARAMTER

SET Filename=%1

::Checking if file doesnot exist display error and quit

IF NOT EXIST %Filename% ECHO File does not exist & GOTO :EOF

::Extracting the last three characters of the file to get the extension

SET FileExtension=%Filename:~-3%

::Matching the extension to txt to check that the file is a text file

IF NOT "%FileExtension%"=="txt" ECHO File is not a text file & GOTO :EOF

::outputfile cleared

TYPE NUL > outputfile.txt

::Loop to iterate over all domains in file

FOR /F "usebackq tokens=\*" %%a IN ("%Filename%") DO (

::Appending Domain to outputfile

ECHO Domain: %%a>>outputfile.txt

::Appending IP Address text to outputfile without newline

ECHO |SET /P=IP Address:>>outputfile.txt

::Loop to iterte over the output of nslookup and only extracting the ip addresses from the output

FOR /F "tokens=\*" %%i IN ('nslookup %%a ^| findstr /R "[0-9]\*\.[0-9]\*\.[0-9]\*\.[0-9]\*"') DO (

::Appending IP Addresses to the outputfile

ECHO %%i >>outputfile.txt

)

ECHO ------------------------------- >>outputfile.txt

)

::Clear screen

CLS

::Print outputfile

TYPE outputfile.txt

DEL outputfile.txt

GOTO :EOF

::TO handle no parameter case

:NOFILE

ECHO Please select the option you want to procees:

ECHO 1)input a domain name

ECHO 2)quit the program

::User input of choice

SET /P choice=""

::If user chooses to quit go to end of file

IF "%choice%" == "2" GOTO :EOF

::If user enters invalid input then print error and take input again

IF "%choice%" NEQ "1" IF "%choice%" NEQ "2" ECHO Invalid choice & GOTO :NOFILE

::Taking domain as user input if choice is 1. Displaying the ip addresses against that domain

SET /P Domain=Input the domain name:

ECHO|SET /P=IP Address:

nslookup %Domain%

ECHO -------------------------------

::Displays menu again until user quits

GOTO :NOFILE

::To handle extra parameters case

:EXTRAPARAMTER

ECHO Too many parameters entered

### Linux Bash Script

#!/bin/bash

#checking if count of parameters is greater than 1 then display error and quit

if [ "$#" -gt 1 ]; then

echo "Too many parameters entered"

exit

fi

#checking if no parameters are entered

if [ -z "$1" ]; then

#loop to keep executing code till user quits

while true; do

#displaying menu

echo Please select the option you want to process:

echo 1 Input a domain name

echo 2 Quit the program

#taking user input in choice

read choice

#case statements to check choice and perform appropriate action

case "$choice" in

"1")

#when choice in one take user input for domain

echo Enter domain name

read domain

#use host command to print IP address against that domain

echo IP address:

host $domain

echo -------------------------

;;

"2")

#when choice is 2 quit program

echo Exitting program

exit

;;

\*)

#any choice other than 1 or 2 is invalid

echo Invalid input

;;

esac

done

exit

fi

#when a filename is passed as parameter store it

filename=$1

#checking if file exists

if !(test -e "$filename"); then

echo File does not exist

exit

fi

#checking if extension is txt

if [ "${filename#\*.}" != "txt" ]; then

echo File is not text file

exit

fi

#reading file line by line

while IFS= read -r domain

do

#storing line in domain variable and displaying it

echo Domain: "$domain"

echo -n IP Address:

#writing output of host to temp.txt

host "$domain" > temp.txt

#using grep to read data from temp.txt and display the ip addresses only

grep -o "[0-9]\*\.[0-9]\*\.[0-9]\*\.[0-9]\*" "temp.txt"

echo --------------------------------------

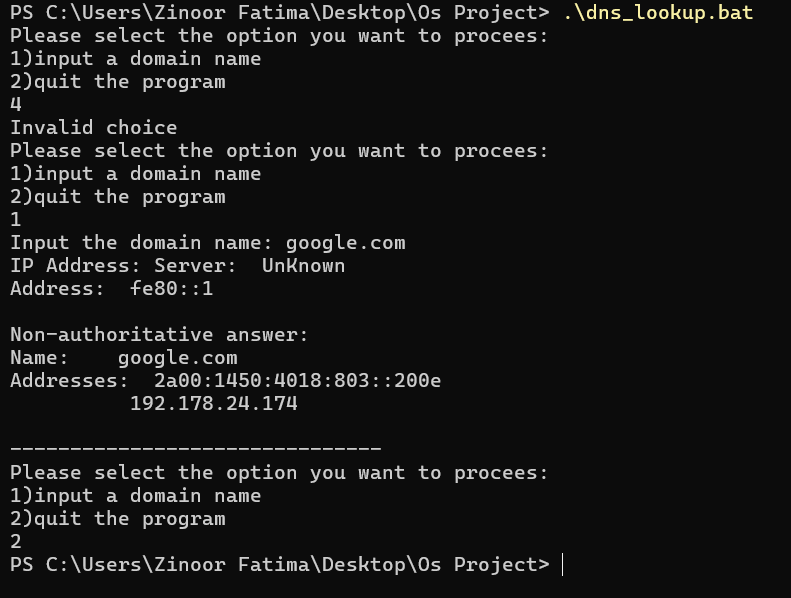
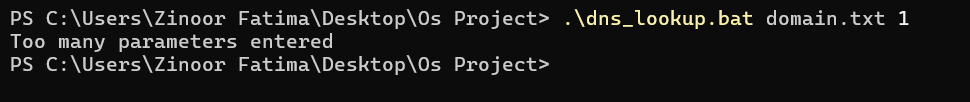
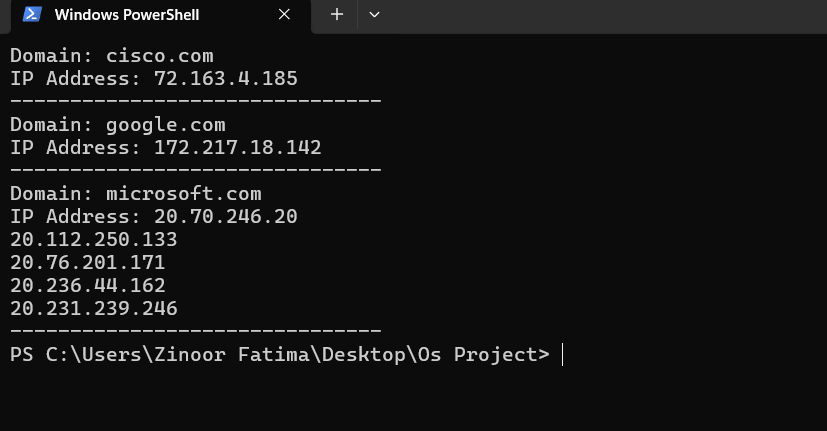
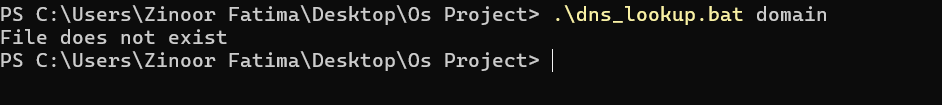
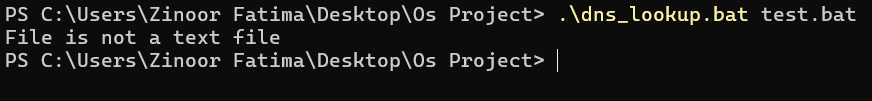
done < $filename

#deleting temp.txt

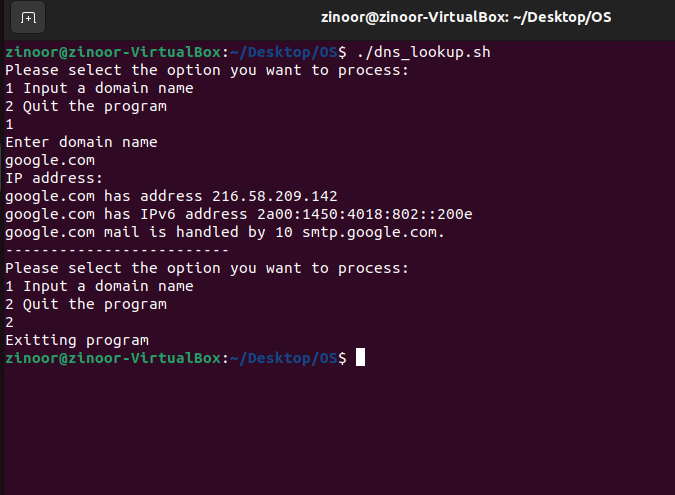
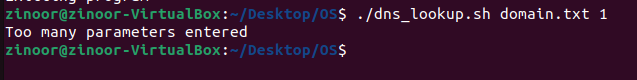
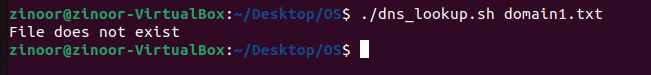
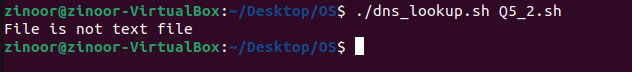
rm temp.txt

## Screenshots

### MS-DOS Batch Screenshots

* No parameter
* More than 1 parameter
* 1 parameter and correct file name
* 1 parameter and nonexistent filename
* 1 parameter and no text file

### Linux Bash Script Screenshots

* No parameter
* More than 1 parameter
* 1 parameter and correct file name
* 1 parameter and nonexistent filename
* 1 parameter and no text file

# Part 2: Research Questions

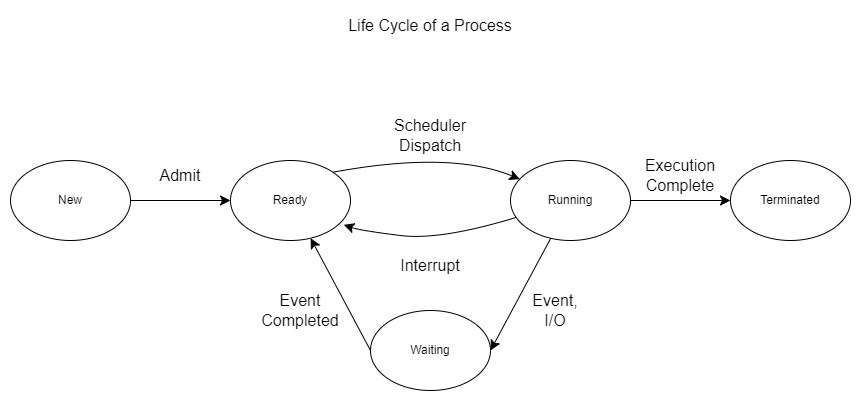
## Question 1

Describe the life cycle of a process, using diagrams, with details about each state. Write down the conditions that a process is switched:

(i) from Ready state to Running state

(ii) from Running state to Ready state

**Answer:**



Life cycle of a process is all the states that the process goes through from when it starts execution till it is completely executed by the CPU. During this life cycle, the process goes through numerous states, and I would be explaining 5 of those states.

1. Start/New state: Whenever a process is first created and enters execution it is said to be in the new state. Once the process is added to the process queue and acknowledged by the operating system it is assigned a new status.
2. Ready state: Any process that is in the process queue ready to be assigned to the processor by the operating system is in ready state. A process is changed from new state to ready state once it can start execution.
3. Running state: A process in running state is assigned a processor by the operating system and is currently in execution.
4. Waiting state: A process that is performing an event and is not using the processor but is also not in ready state as it can only start execution after the event is completed is said to be in waiting state. The event could be anything for example using I/O devices. Once the event is completed, the process is removed from the waiting state and goes back to ready state.
5. Terminated/Completed state: A process that has completed execution is terminated and said to be in terminated state. The process is removed from the process queue.

(i) Conditions to change form Ready state to Running state:

* The scheduler dispatches the process from the process queue and assigns it a processor to start execution.

(ii) Conditions to change form Running state to Ready state:

* An interrupt is called that stops execution of the process. This could be an operating system interrupted by the scheduler that preempts the running process, as it wants to halt the process execution and allocate the processor to another process.
* A process can also move from running to ready state if the process’s priority expires.

## Question 2

Shortest Job First (SJF) is a relatively simple CPU scheduling algorithm for process management. Consider that there are two busy multi-programming systems A and B. The new processes created on System A are mostly long CPU burst processes with only occasional short CPU burst processes. The new processes created on System B are mostly very short CPU burst processes.

(i) State whether process starvation is likely to occur on either or both System A and System B. (ii) If process starvation occurs on System A, state which type of processes is likely to be starved. Similarly, if process starvation occurs on System B, state which type of processes is likely to be starved. Your answer should refer to relevant characteristics of SJF algorithm.

**Answer:**

(i)Process starvation can occur on both systems, but it is highly likely to occur on System B.

(ii)There is a higher chance of process starvation to occur on System B as System B creates very short new CPU processes. If a relatively longer process is in the process queue at system B, it will not get the chance to be executed due to the SJF scheduling algorithm being used, which chooses to execute the process with less burst time first. While on System A, as there are mostly long CPU processes so if occasionally a short process enters the queue, it gets executed and then the long processes get their turns. In System B due to more short processes being created, long CPU processes are likely to be starved.

## Question 3

Consider the kitchen example and you are now the manager of the kitchen. There are 3 bowls, 2 stirrer, and 1 measuring cups in the kitchen. There are 3 chefs working in the kitchen. Chef C1 is holding 2 bowls, 1 cup, and needing 1 stirrer. Chef C2 is holding one bowl, one stirrer, and needing one cup. Chef C3 is holding one stirrer and needing one stirrer.

(i) There will be bad consequences when a deadlock occurs in the kitchen. Your boss is very concerned and ask you in theory the conditions for deadlock to occur. Answer your boss.

(ii) Draw a Resource Allocation Graph (RAG) to describe the resource allocation situation outlined above.

(iii) Based on your RAG and discuss if a deadlock has occurred or the situation concerning deadlock.

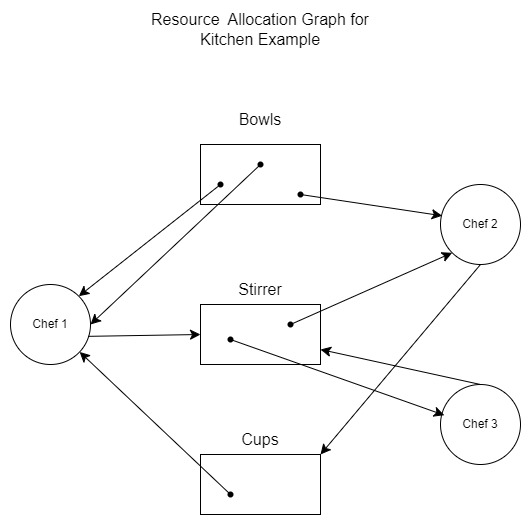
(iv) In a real computer system, what will be the chefs, bowls, cups, stirrers represented for? Justify your answers with reasons.

**Answer:**

(i) There are four condition that can surely cause a deadlock scenario:

1. Hold and Wait is a case when a process holds a resource and is in a waiting state as it does not have all the required resources to complete an event or task. Meanwhile the resources required by the process to complete the task are held by another process that is waiting for the occupied resource. Both processes hold the resources waiting for the resources to be freed. This results in all other processes to enter the waiting state, as the waiting time reaches infinity and causes a deadlock situation.
2. Mutual Exclusion is the condition when a resource is prohibited from sharing. A resource can be occupied by one process at a time. This mutual exclusion of resources can result in a hold and wait situation consequently resulting in a deadlock.
3. No Preemption is a condition when the operating system does not interrupt a process in execution. A process in execution will continue executing until it is terminated. This means if the process is in possession of a resource, the operating system cannot forcefully free the resource or halt the process to take the resource from the process. Instead, it must wait till the process has completed execution. This can again lead to a hold and wait situation and a resource stuck with a process waiting for infinite time cannot be freed resulting in deadlock.
4. Circular Wait is the condition when the processes in the process queue are dependent on each other for their execution in a circular manner. Suppose P0 is dependent on P1, which is dependent on P2, which in turn is dependent on P0. This results in a circular wait where each process waits for the other to complete execution and results in a deadlock.

(ii)



(iii) According to the Resource Allocation Graph (RAG) a deadlock occurs. Each chef is holding a resource but does not have all the resources required to complete its task. Thus, each chef is waiting for other chefs to complete their tasks and free the resources. This results in a circular Wait and a hold and wait situation; therefore, a deadlock occurs.

(iv) In a computer system the chefs will be the processes and bowls, stirrer, and cups will be the resources required by the processes. Just as in real life the chefs have to perform tasks and need the cups, bowls, and stirrer to perform their tasks, in a computer system the processes will need the resources to complete execution.

## Question 4

Consider an online computer system booking system for air tickets. It is expected that thousands of customers are using the system at the same time. A customer typically spends around 10 minutes to complete the querying, booking, and payment processes. Normally customers access the booking systems using a smartphone application that communicates with a web server for almost every operation.

(i) One design decision is about selecting the most suitable CPU scheduling algorithm for the server. In the evaluation, suggest the performance metric that is most relevant? Justify your answer.

(ii) To support more and more customers, the server needs an upgrade. Suggest whether it is more important to increase the size of the main memory or using a more powerful CPU. Justify your answer.

(iii) Study the following article. <https://www.catchpoint.com/blog/web-caching>

 Based on the information given in the article, discuss if web caching is a suitable solution to achieve the performance metric you answered in part (i).

**Answer:**

(i) In the online ticket booking system the most important and relevant performance metric would be the response time of the CPU. As there are thousands of customers accessing the system at the same time, it is essential for the system to respond to each request as fast as it can, that is minimizing the response time. The lesser the response time the better the user experience would be. Minimizing the response time would give a smoother experience and avoid issues like users booking the same seats.

(ii) In order for the system to support more customers it is important to increase the size of the main memory. More customers mean more requests to save, which requires a larger main memory. A powerful CPU can provide a better experience and faster response but only on an individual level. To cater for an increased number of customers it is firstly essential to have the memory space to store their requests.

(iii) Web caching can greatly help in achieving the performance metric, that is response time. As web caching stores the most requested pages and data on the website and does not require access to the server again and again, it reduces latency and provides the user with a faster response. Web caching greatly improves user experience by reducing the response time. Moreover, caching also reduces the number of requests going on the server, as the website will get the cached data instead of requesting the server, as a result increasing average response time at the server as well.

## Question 5

In this question, you are asked to complete the two following questions given in the practical exercises. Program code with comments and explanation should be given in the Word report.

(i) Write a MS-DOS batch program which prompts the user for the name of a batch file, checks it exists (outputting an error and stopping if it does not) then goes on to ask the user for the author of the batch file’s name. It then adds two comment (rem) lines to the start of the batch file the first says “Written by” and the author’s name and the second the date and time. When it finishes the batch file should still be runnable with no errors! <https://canvas.anglia.ac.uk/courses/33881/files/4029059?module_item_id=1877624>

(ii) Write a bash script that uses a looping menu and case statements to perform a range of simple file operations. For each option the user is prompted for the correct number of parameters for the specific command and where necessary verifies the files/directories exist or not as appropriate. Option 1 is to copy one file to another, option 2 to rename one file to another, option 3 to create a directory and option 4 quit. <https://canvas.anglia.ac.uk/courses/33881/files/4029061?module_item_id=1877627>

**Answer:**

**(i)MS-DOS script code**

@ECHO OFF

SETLOCAL DISABLEDELAYEDEXPANSION

::Taking batch filename as user input

SET /P Filename=Enter name of a batch file:

::Displaying an error message and exiting the batch file if the file entered by user does not exist

IF NOT EXIST %Filename% ECHO File does not exist & GOTO :EOF

::Extracting the last three characters of the file to get the extension

SET FileExtension=%Filename:~-3%

::Matching the extension to bat to check that the file is a batch file

IF NOT "%FileExtension%"=="bat" ECHO File is not a batch file & GOTO :EOF

::Taking file author as user input

SET /P Author=Enter name of a Author of the file:

::Storing comments to add to file in variables

SET Filedata=REM Written by %Author%

SET DateTime=REM Date: %date% %time%

::Adding comments to temp file

ECHO %Filedata%>>temp.txt

ECHO %DateTime%>>temp.txt

::Appending data from entered batch file to temp text file

CAT %Filename%>>temp.txt

::Overwriting the data in inputted batch file with the data from temp text file

TYPE temp.txt>%FILENAME%

::Deleting the temp text file

DEL temp.txt

**(ii)Linux bash script code**

#!/bin/bash

#loop to keep running the code till choice is to quit

while true; do

#displaying the menu

echo Choose an option

echo 1 Copy one file to another

echo 2 Rename one file to another

echo 3 Create a directory

echo 4 Quit

#inputting user choice

read choice

#using case statement to perform tasks according to the value in choice variable

case "$choice" in

"1")

#when choice is 1 prompting user to enter source and destination files

echo Enter source filename

read sourcefile

echo Enter destination filename

read destfile

#checking if file entered exists

if test -e "$sourcefile"; then

#copying data from sourcefile to destfile

cp "$sourcefile" "$destfile"

echo Contents of file $sourcefile are copied to $destfile

else

echo $sourcefile does not exist

fi

;;

"2")

#when choice is 2 prompting user to enter file's old name and new name

echo Enter filename

read oldname

echo Enter new name

read newname

#checking if file entered exists

if test -e "$oldname"; then

#renaming file oldname to newname

mv "$oldname" "$newname"

echo File $oldname has been renamed to $newname

else

echo $oldname does not exist

fi

;;

"3")

#when choice is 3 prompting user to enter directory name

echo Enter directory name

read directoryname

#checking if directory already exists

if test -e "$directoryname"; then

#displaying error message

echo Directory "$directoryname" already exists

else

#if directory does not exist then create directory

mkdir "$directoryname"

echo Directory "$directoryname" is created

fi

;;

"4")

#when choice is 4 quitting the code

exit

;;

\*)

#if there is any other choice than 1 2 3 4 print invalid choice

echo invalid choice

;;

esac

done

# Bibliography

1. Extract filename and extension in Bash. (n.d.). Retrieved April 5, 2024, from Stack Overflow website: https://stackoverflow.com/questions/965053/extract-filename-and-extension-in-bash
2. Friends Batch File Question- Is there a way to append to top of file in batch. (n.d.). Retrieved April 5, 2024, from www.computerhope.com website: https://www.computerhope.com/forum/index.php?topic=148705.0
3. How to Filter Strings in Bash Scripts. (2023, October 18). Retrieved April 5, 2024, from www.squash.io website: https://www.squash.io/how-to-filter-strings-in-bash-scripts/
4. Need a little help filtering lines in batch. (n.d.). Retrieved April 5, 2024, from Stack Overflow website: https://stackoverflow.com/questions/14045275/need-a-little-help-filtering-lines-in-batch
5. NSLOOKUP in a batch file. (n.d.). Retrieved April 5, 2024, from groups.google.com website: https://groups.google.com/g/alt.msdos.batch.nt/c/WjM48879WvM
6. Rahman, N. (2023, December 16). 6 Ways to Check If a File Exists or Not in Bash. Retrieved April 5, 2024, from LinuxSimply website: https://linuxsimply.com/bash-scripting-tutorial/conditional-statements/if/check-if-file-exists/
7. shell script to remove a file if it already exist. (n.d.). Retrieved April 5, 2024, from Stack Overflow website: https://stackoverflow.com/questions/31318068/shell-script-to-remove-a-file-if-it-already-exist

(n.d.-a).

(n.d.-b).