Logo

Description automatically generated

Faculty of Science and Engineering

School of Computing and Information Science

**Module MOD003218:**

**Operating Systems**

**Element 010**

**Operating Systems Assignment**

**SID NUMBER: 2179610**

Trimester 2 – 2022

Submission deadline: 24/04/2023

Table of Contents

## Part 1:Scripting Task……………………………………………………...2

## 1.Code for Linux………………………………………………...2

## 2.Program Flow for linux………………………………………4

## 3.Code for MS-DOS..…………………………………………...5

## 4.Test cases for MS-DOS…...…………………………………6

## 5.Program Flow for MS-DOS.………………………………....9

## Part 2:Research Questions……………………………………………..10

## 1.Question 1..…………………………………….…………….10

## 2.Question 2…...………………………...………………….....10

## 3.Question 3.…………………………………………….…......11

## 4.Question 4..…………………………….…………………….11

## 5.Question 5..………………………………………….……….12

## 5.Reference List…...………………………………….……….12

**CODE FOR LINUX**

#!/bin/bash

# Check if the user enters no parameter

if [[ $# -eq 0 ]]

then

echo "Enter source folder: "

read sourceFolder

if [[ ! -d $sourceFolder ]];

then

echo "The source folder does not exist."

exit

else

echo "Enter the destination folder: "

read destinationFolder

if [[ ! -d $destinationFolder ]]

then

mkdir $destinationFolder

# Do copy command

cp $sourceFolder $destinationFolder

exit

else

# prompt the user to overwrite or not

echo "Overwrite? "

exit

fi

fi

fi

# Check if the user enters more than one parameter

if [[ $# -gt 1 ]]

then

echo "Too many parameters "

exit

fi

# Check the command line parameter provided by the user

if [[ $# - eq 1 ]];

then

filename=$1

fi

#Check if the file exists

if [[ ! -f $filename ]];

then

echo "The file $filename does not exist."

exit

fi

# Read each line of the text file and perform a copy task

while read -r line; do

the\_source\_folder=$(echo "$line" | awk '{print $1}')

the\_destination\_folder=$(echo "$line" | awk '{print $2}')

# Check if the source folder exists

if ! check\_folder\_existence "$the\_source\_folder"; then

continue

fi

# Check if the destination folder exists

if ! check\_folder\_existence "$the\_destination\_folder"; then

read -p "The destination folder $the\_destination\_folder already exists. Do you want to remove it? (y/n)" the\_choice

if [[ "$the\_choice" != "y" ]]; then

continue

fi

# Remove the destination folder

sudo rm -rf "$the\_destination\_folder"

fi

#Copy from source folder to destination folder

cp -r $the\_source\_folder $the\_destination\_folder

#Check if the copy task was successful

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

echo "The contents of the folder $the\_source\_folder has been successfully copied

to the $the\_destination\_folder

else

echo "Error occurred when copying content from the $the\_source\_folder to the $the\_destination\_folder

fi

#Check if file exits

function check\_file\_existence {

if [[ -f "$1" ]]; then

echo "File $1 exists."

return 0

else

echo "File $1 does not exist."

return 1

fi

}

# Check if folder exists

function check\_folder\_existence {

folder\_path=$(dirname "$1")

if [[ -d "$folder\_path" ]]; then

echo "Folder $folder\_path exists."

return 0

else

echo "Folder $folder\_path does not exist."

return 1

fi

}

done < $filename

**PROGRAM FLOW FOR LINUX**

1. Start the program
2. Check if the user entered a command line parameter.

a. If yes, set it as the filename.

b. If no, prompt the user to enter the source folder and destination folder.

i. Check if the source folder exists.

ii. If it doesn't exist, display an error message and exit.

iii. Check if the destination folder exists.

iv. If it doesn't exist, create it

v. Copy the contents of the source folder to the destination folder.

vi. If successful, display a success message. Otherwise, display an error message and exit.

1. Check if the file specified by the filename exists.

a. If it doesn't exist, display an error message and exit.

1. Read each line of the text file and perform the following tasks for each line:

a. Extract the source folder and destination folder from the line.

b. Check if the source folder exists.

i. If it doesn't exist, continue to the next line.

c. Check if the destination folder exists.

i. If it exists, prompt the user to confirm overwriting it.

ii. If the user does not confirm overwriting it, continue to the next line.

iii. If the user confirms overwriting it, remove the destination folder.

d. Copy the contents of the source folder to the destination folder.

e. If successful, display a success message. Otherwise, display an error message.

1. End the program

**CODE FOR MS-DOS**

@echo off

setlocal EnableDelayedExpansion

REM Check if a command line parameter was provided

if [%1] == [] (

set /p sourceFolder="Please enter the source folder: "

if exist !sourceFolder! (

set /p destinationFolder="Please enter the destination folder: "

if exist !destinationFolder! (

echo Directory already exist. Overwrite directory?

CHOICE /C YN /M "Press Y for Yes, N for No"

if "!ERRORLEVEL!"=="1" (

xcopy !sourceFolder! !destinationFolder! /e /y

exit /b 1

) else if "!ERRORLEVEL!"=="2" (

echo Copy terminated

exit /b 1

)

) else (

xcopy !sourceFolder! !destinationFolder! /i /e

exit /b 1

)

) else (

echo The source folder !sourceFolder! does not exist

exit /b 1

)

) else (

set filename=%1

)

REM Check if more than one file

if not [%2] == [] (

echo You have entered more than one file

exit /b 1

)

REM Check if the file exists

if not exist %filename% (

echo The file %filename% does not exist.

exit /b 1

)

REM Read each line of the text file and perform a copy task

for /f "usebackq tokens=1,2" %%a in ("!filename!") do (

REM Check if the source folder does not exists

if not exist %%a (

echo The source folder %%a does not exist.

) else (

REM Check if the destination folder exists

if exist %%b (

set /p choice="The destination folder %%b already exists. Do you want to overwrite it? (y/n) "

if /i "!choice!"=="y" (

REM Copy from source folder to destination folder

xcopy /e /i /y %%a %%b

echo Copy task successful

) else (

echo The folder was not copied

)

) else (

REM Copy from source folder to destination folder

xcopy /e /i %%a %%b

echo Copy task successful

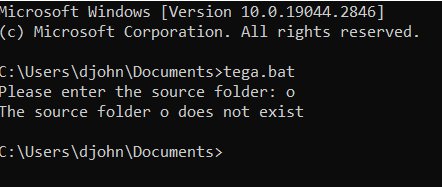
)

)

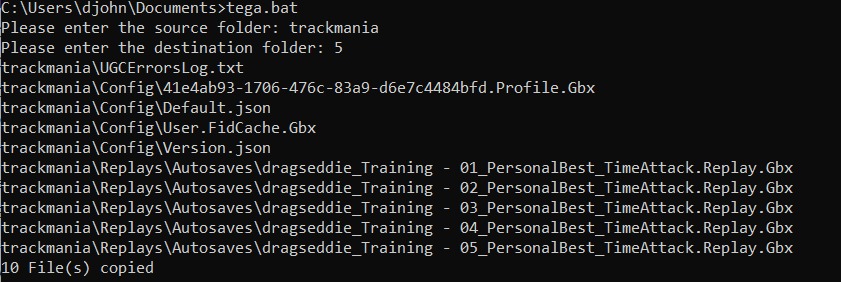
)

**TEST CASES FOR THE MS-DOS CODE**

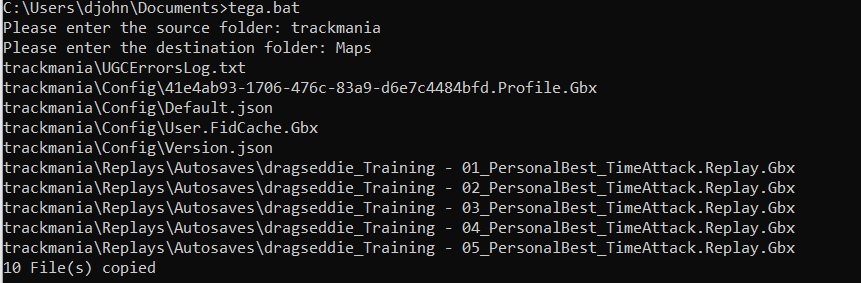
WHEN SOURCE FOLDER DOES NOT EXIST



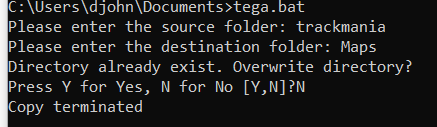
WHEN SOURCE FOLDER EXISTS BUT DESTINATION FOLDER DOES NOT EXIST



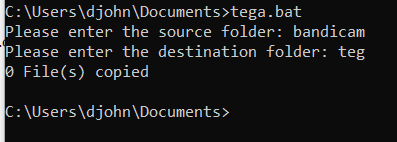
WHEN SOURCE AND DESTINATION FOLDERS EXIST

****

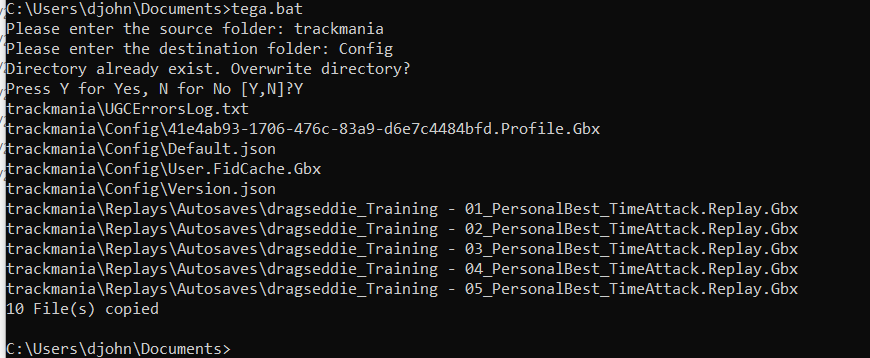
WHEN SOURCE AND DESTINATION FOLDER EXIST AND THE CASE IS NO TO OVERWRITE A COPY



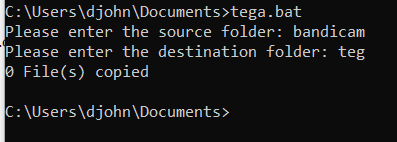
WHEN SOURCE FOLDER EXIST BUT DESTINATION DOES NOT AND THERE ARE 0 FILES THAT CAN BE COPIED



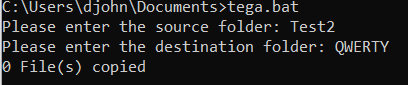
WHEN SOURCE AND DESTINATION FOLDER EXIST AND THE CASE IS NO TO OVERWRITE A COPY



WHEN SOURCE AND DESTINATION FOLDERS EXIST BUT DESTINATION FOLDER IS EMPTY



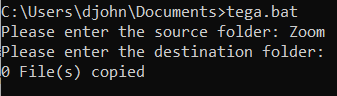
WHEN SOURCE FOLDER IS EMPTY AND DESTINATION FOLDER DOES NOT EXIST



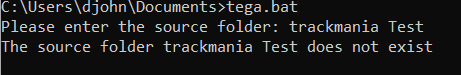
WHEN THERE IS NO PARAMETER ENTERED AT THE SOURCE FOLDER FIELD

**C:\Users\djohn\Documents\2179610\when there is no parameter entered at the source folder.png**

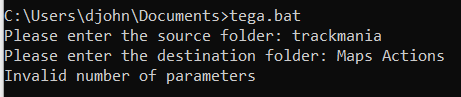
WHEN THERE IS NO PARAMETER ENTERED AT THE DESTINATION FOLDER FIELD



WHEN MORE THAN ONE PARAMETER IS ENTERED AT THE SOURCE FOLDER FIELD



WHEN MORE THAN ONE PARAMETER IS ENTERED AT THE DESTINATION FOLDER FIELD



**PROGRAM FLOW FOR MS-DOS**

1. Start the program.
2. Check if a command line parameter was provided.
3. If no command line parameter was provided:

a. Ask the user to enter the source folder.

b. If the source folder exists:

i. Ask the user to enter the destination folder.

ii. If the destination folder exists:

1. Ask the user if they want to overwrite the destination folder.

2. If the user chooses to overwrite the folder, copy the contents of the source folder to the destination folder.

3. If the user chooses not to overwrite the folder, exit the program.

iii. If the destination folder does not exist, copy the contents of the source folder to the destination folder.

c. If the source folder does not exist, exit the program.

1. If a command line parameter was provided:

a. Check if more than one file was entered.

b. Check if the file exists.

c. Read each line of the text file and perform a copy task.

1. End the program.

Question 1

What is meant by starvation in process management? What will be the consequence of starvation?  
How can starvation be avoided?

ANSWERS:

Starvation in an operating system occurs when a process is unable to access the necessary resources for an extended period due to those resources being allocated to other processes. In other words, it is a resource management problem that results in a process being deprived of the resources it needs for an extended period.(Takur, 2022)

If higher priority processes continually dominate the processor, lower priority processes may have to wait indefinitely during starvation. However, starvation cannot cause a deadlock since low-priority programs are not interacting with any resources. (Takur, 2022)

STEPS TAKEN TO AVOID STARVATION

* A freelance manager handling the distribution of resources to ensure that they are allocated evenly..
* Taking into account factors such as aging when establishing the priority criteria for resource allocation is necessary to prevent starvation.
* Randomly selecting processes or allocating processors for resource allocation should be avoided as it can contribute to the occurrence of starvation. (Takur, 2022)

Question 2

In the context of memory management, what is a page fault? Describe the actions that follow after a  
page fault occurs.

ANSWERS:

A page fault is a type of interruption that takes place when a software program tries to access a memory block that is not presently available in the RAM of the system.

When a page fault occurs, an exception is raised to inform the operating system that it needs to fetch the required pages from virtual memory to allow the program to continue running. After the data is loaded into physical memory, the program can continue its execution as usual.(Noida, 2021)

Question 3

My email system gives me a choice of POP3 or IMAP. I am currently using POP3. Why might I want to  
switch to IMAP? Explain both POP3 and IMAP and give justifications whether I should switch to IMAP.

ANSWERS:

The term POP3 refers to Post Office Protocol version 3, which is a protocol used to access email inboxes stored on a mail server. It enables users to download and remove messages from the server. When a user connects to the mail server using POP3, all the messages in the mailbox are retrieved and saved on their local computer. The messages are then deleted from the remote server.

IMAP, short for Internet Message Access Protocol, is a protocol that allows users to access and manage their email messages on a mail server. It provides several features such as message organization into folders, permanent message deletion, and efficient message search. Additionally, it enables users to set or remove email flags and selectively retrieve email attributes. Unlike POP3, messages are not automatically deleted from the server with IMAP, and they remain there until explicitly deleted by the user.

While the POP3 protocol presumes that your email is being accessed only from one application, IMAP allows concurrent access by various users. This is why IMAP is more appropriate for you, if you're going to access your email from different locations or if your messages are managed by various users.

Question 4

In an operating system what does it mean when a computer is said to be “thrashing”? How can it be  
prevented?

ANSWERS:

The computer is said to be thrashing if the CPU spends more time serving page faults than executing the pages. This leads to low CPU utilization and the Operating System in return tries to intensify the level of multiprogramming.

To prevent thrashing, it is crucial to allocate enough frames in physical memory to each process to meet its requirements while it is running. This way, the process can execute without experiencing frequent page faults, which can cause the CPU to spend more time handling them than executing actual instructions.(Kumari, 2022)

Question 5

In the bash shell what does the ampersand (“&”) symbol mean when used in a command? If I used an  
& in a command why might I need to use the command “ps”?

ANSWERS:

The ampersand character serves two purposes in Bash. Firstly, it marks the end of a command. Secondly, it executes the command asynchronously in the background. To verify the status of a background process and ensure that it is functioning correctly, the "ps" command can be used.

Refenrence List

Thakur, S. (2022) *Competitions, quizzes, hackathons, scholarships and internships for students and corporates*, *Unstop*. Available at: https://unstop.com/blog/starvation-in-os (Accessed: April 23, 2023).

*What is a page fault?* (2021) *Computer Hope*. Available at: https://www.computerhope.com/jargon/p/pagefaul.htm#:~:text=A%20page%20fault%20is%20an,SSD%20or%20HD)%20to%20RAM. (Accessed: April 23, 2023).

*Page fault handling in operating system - javatpoint* (2021) *www.javatpoint.com*. Available at: https://www.javatpoint.com/page-fault-handling-in-operating-system (Accessed: April 23, 2023).

*What are email protocols - POP3, SMTP and IMAP* (2023) *SiteGround Knowledge Resources*. Available at: https://www.siteground.co.uk/tutorials/email/protocols-pop3-smtp-imap/#:~:text=While%20the%20POP3%20protocol%20assumes,are%20managed%20by%20multiple%20users. (Accessed: April 23, 2023).

Kumari, R. (2021) *Thrashing in operating system: What is thrashing in os*, *Scaler Topics*. Scaler Topics. Available at: https://www.scaler.com/topics/thrashing-in-os/ (Accessed: April 23, 2023).

Top of Form