MODULE MOD003218:

OPERATING SYSTEMS ASSIGNMENT 2023/2024

SID NO:

2264684

PART1

PART1

LINUX JCL SCRIPT

PROGRAM CODE:

```
#!/bin/bash


#This batch script prompts a user for a domain name, and displays the corresponding IP address


#Prompt the user for one parameter
if [ "$1" = "" ]
then
    #Counter is set to 0
    choice=0


    #Loop through the menu till the user picks 2
```

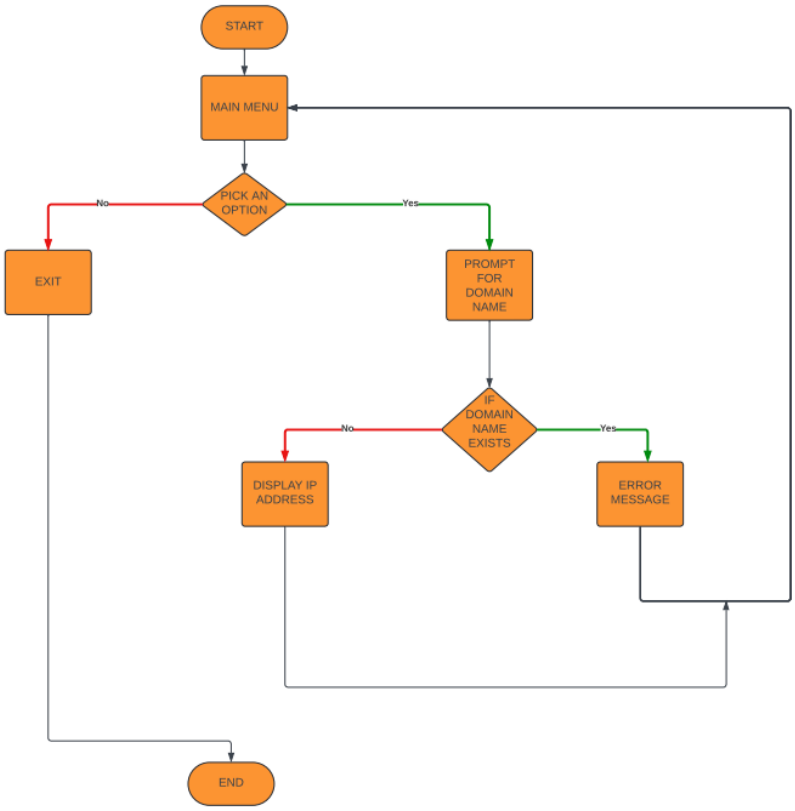
```
while [ "$choice" != 2 ]
do
  #Display the menu
  echo "Main Menu"
  echo "1.Input a domain name"
  echo "2.Quit from the program"
  echo "Please pick an option"

  #Prompt the user for a choice
  read choice

    #If the user picks option 1, prompt the user for a domian name
    if [ "$choice" = 1 ]
    then
      echo "Input a domain name: "
      read name

        #Check if domain name exists
#Check if domain name exists
```

```
#Using this command alongside the host command, desposes any unwanted output so that the check can be carried out without any interruptions in the terminal

#If domain name exists, display ip address

if host "$name" >/dev/null;

then


  #Create a variable to take in all that would be displayed when the host command is used alongside the domain name

  domain=$(host $name)


  #Use the grep command to filter the lines with "has address" of the output

  #Use the awk command to extract only the IPv4 address

  echo "$domain" | grep "has address" | awk '{print "IP address:", $4}'


  #Return to the main menu

  continue

#If domain name does not exist, print "domain name does not exist"

else
```

```
                echo "Domain name does not exist"
                #Return to main menu
                continue
            fi
        else
        #If user choice is 2 exit from the code
         if [ "$choice" = 2 ]
         then
            #Display exiting...
            echo "Exiting..."
            #Terminate code
            exit
#If user inputs a wrong value, print an error message
         else
            echo "Error"
            echo "Wrong input"
            #Return to main menu
            continue
         fi
        fi
```

```
    #Increase counter by 1
    ((choice++))
    #End loop
    done
#set variable to take one parameter
else
  name=$1
  choice=$1
fi

#delete variable
choice=""
name=""
domain=""
```

FLOWCHART



TEST CASES:

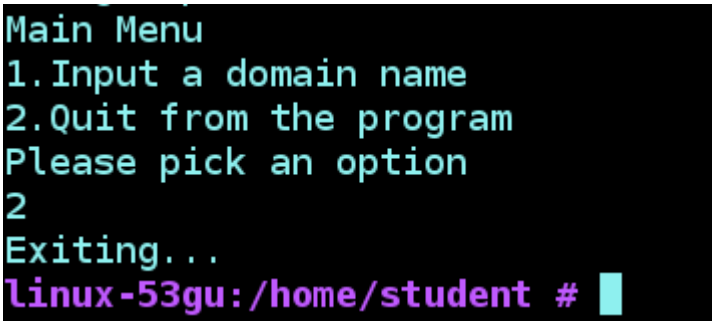
- The program starts by displaying the menu, and then prompts a user to make a choice amongst the options.
- IF USER CHOICE IS 1: The user is prompted for a domain name. The program checks if the domain name exists, and if it does, it displays the IPv4 address and returns to the menu.

```
linux-53gu:/home/student # ./projectos.sh
Main Menu
1.Input a domain name
2.Quit from the program
Please pick an option
1
Input a domain name:
cisco.com
IP address: 72.163.4.185
Main Menu
1.Input a domain name
2.Quit from the program
Please pick an option
```

- IF DOMAIN NAME DOES NOT EXIST: If the domain name doesn't exist ,the program prints an error message and returns back to the main menu

```
Main Menu
1.Input a domain name
2.Quit from the program
Please pick an option
tick.com
Error
Wrong input
Main Menu
1.Input a domain name
2.Quit from the program
Please pick an option
```

- IF USER CHOICE IS 2: The program will be terminated



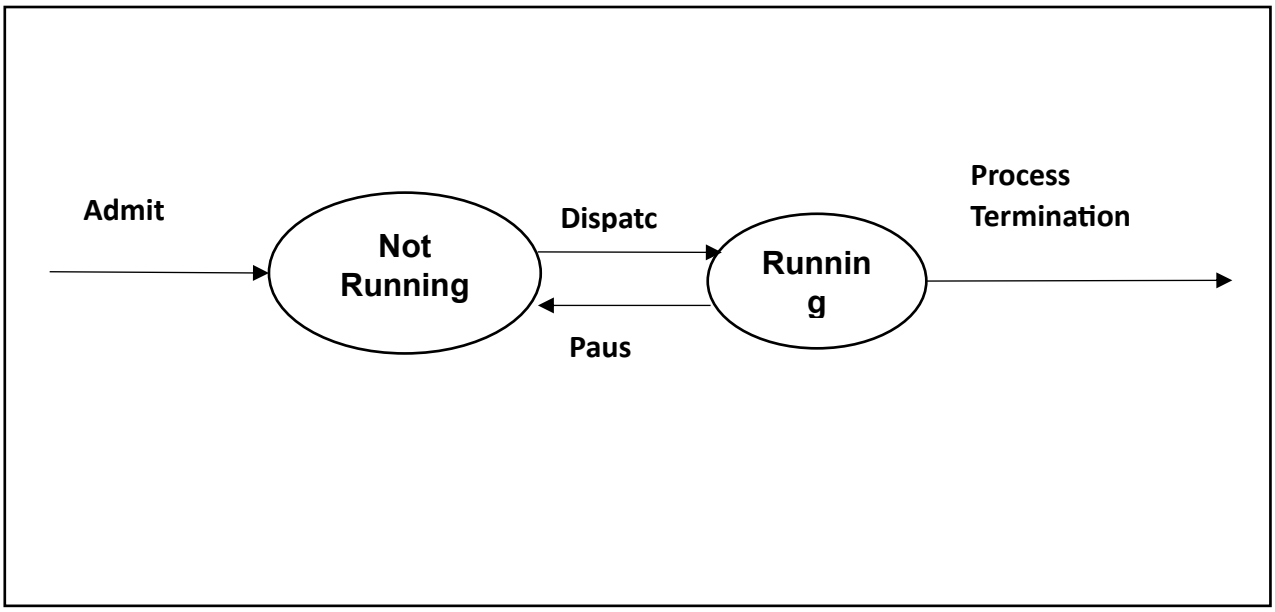
PART 2

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

   ANSWERS:

A process is a computer program that is being executed. When a process is being executed, it goes through many stages before it is aborted. These stages differ in various operating systems. These are the common process lifecycles;

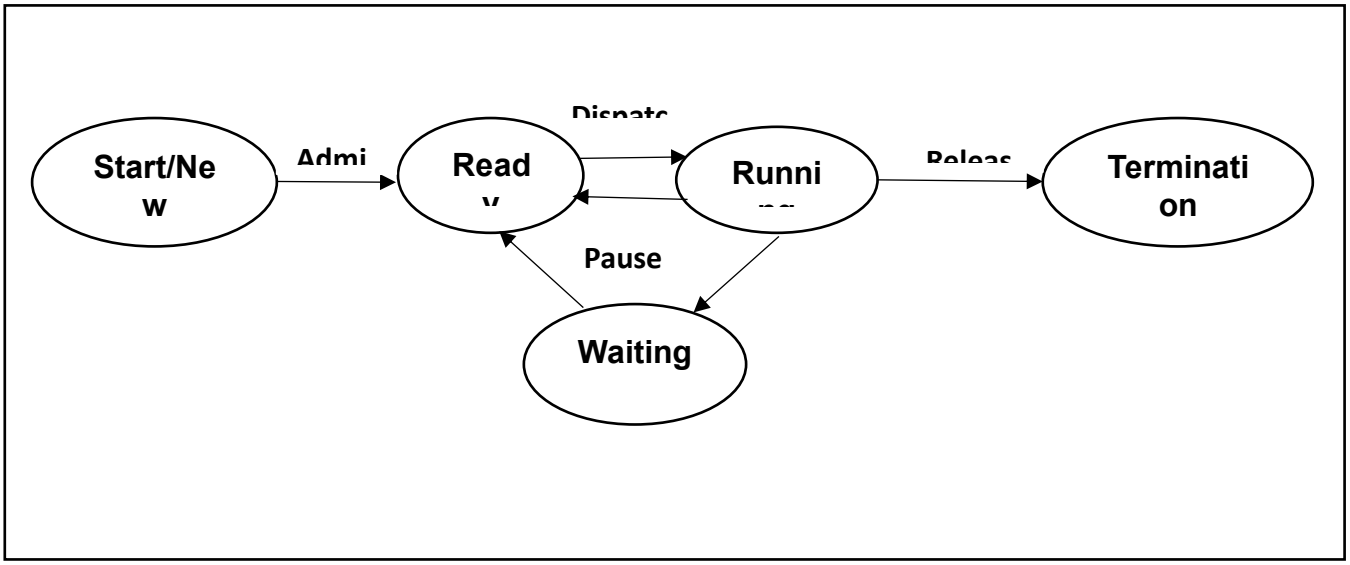
a) **The Two-State Model:** This process lifecycle consists of just two states; the running and not running state. So, its either the process is running or not running on the CPU. When a process is created, it goes into the not running state, then it is stored into a dispatcher. The dispatcher then sends the process to the CPU when the CPU is free ,and it goes into the running state once it uses the CPU.



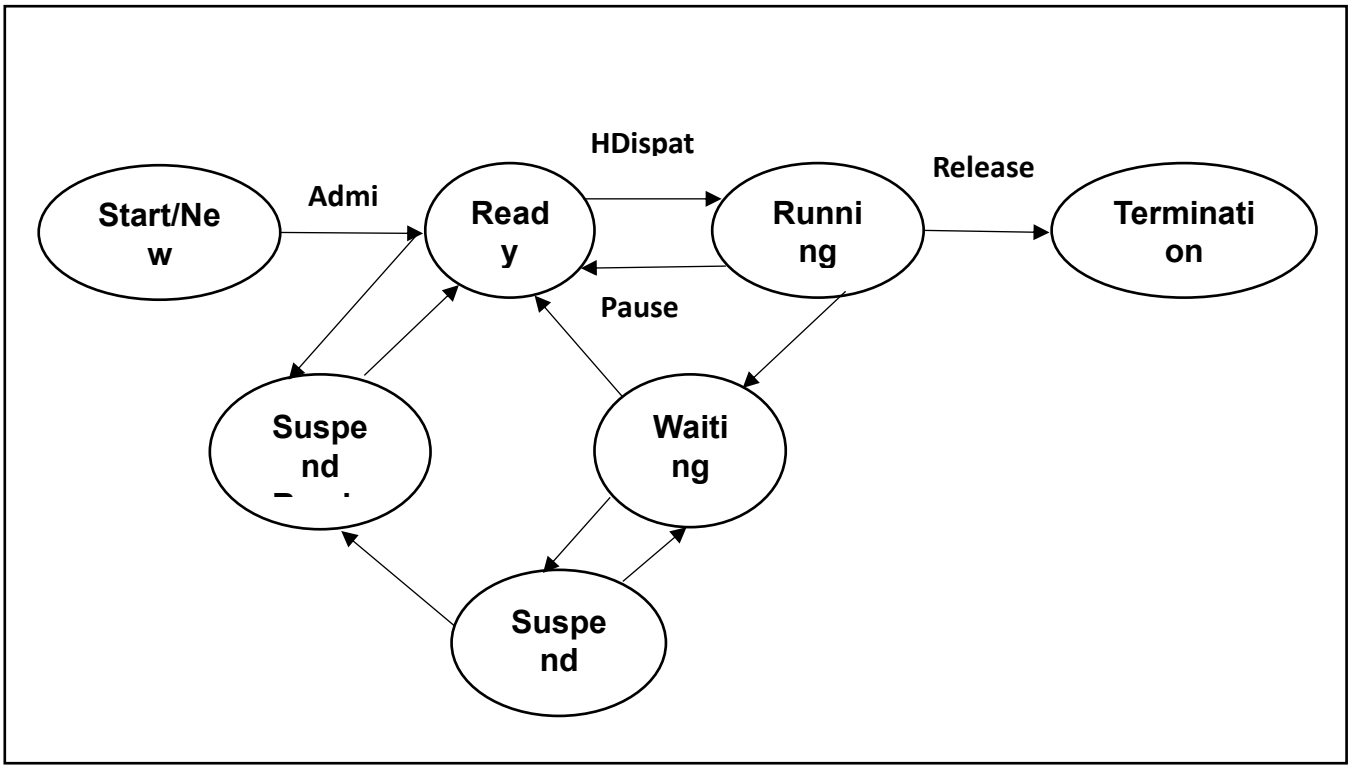
I

b) The Five-State Model: This process lifecycle consists of five states; the start/new state, the ready state, the running state, the waiting state and the termination state.

In the start/new state, the process is just being created. The process is stored in the dispatcher, waiting to be moved to the main memory. After the process has been moved to the main memory, it's state changes from start to ready. The process is in the ready state while it is waiting for the CPU on the main memory. The process is then moved into the CPU once it becomes free for further execution. After the execution has been completed (when a process is in the running state, it means it is being executed in the CPU) ,the process goes to the waiting state, while it waits for some I/O operations to get completed. The process goes to the ready state once the CPU is free again, then from the ready state, it goes to the running state . The process goes to the termination stage once it has completed its execution or the execution has been terminated.



c) **The Seven-State Model:** This process lifecycle consists of 7 states; the start/new state, the ready state, the running state, the waiting state, the suspend wait state, the suspend ready state and the termination state. Processes in the ready state may get exchanged from the main memory ,when this happens, the process goes into the suspend ready state, and then the CPU scheduler moves the process into an external storage. The process goes back to the ready state when it is moved back to the main memory. Sometimes, due to lack of main memory, processes in the waiting state are moved into the secondary storage. These processes then go into the suspend wait state. After this stage, it can either go into the waiting state or the suspend ready state.



i. **Conditions a process is switched from ready state to running state:** A process is switched from the ready state to the running state if it is chosen by the operating system for execution depending on different scheduling algorithms. The scheduling algorithm regulates which process is chosen to be executed from the ready state.

ii. **Conditions a process is switched from running state to ready state:** A process is switched from the running state to ready state if the time-out condition is met. The time-out condition requires that the workflow must wait for a definite amount of time or a specified date. A process is also switched from the running state to the ready state when some priority issues are allocated to

processes by the operating system. A process enters the system when it has a higher priority. This done by a process scheduler.

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.

ANSWERS:

i. Firstly, starvation occurs when a process of low priority ready for the CPU resources waits indefinitely to run. In this case, starvation is likely to occur on system B. This is because the processes created on system A are mostly short CPU burst processes. This means that there are a few long CPU burst processes. Long CPU burst processes are of low priority compared to the short CPU burst processes. This is because schedulers usually favour processes with a shorter burst time. Starvation will occur when all the long burst processes are held back, because of the short CPU burst processes that continuously keeps arriving, causing the long or longer CPU burst processes to wait indefinitely leading to a poor performance.
ii. If process starvation occurs on System A, the type of processes that are likely to be starved are the long CPU burst processes. The SJF scheduling algorithm prioritises based on execution time, also known as burst time. Processes with a smaller execution time (short burst time) are executed first before those with a longer execution time. Starvation will occur to the long CPU burst processes because of the incoming short CPU burst processes.

The processes created on system B are mostly very short burst processes. Since the SJF scheduling algorithm prioritizes processes with a shorter execution time compared to others, the type of processes that are likely to be

starved in system B are the processes with the longest execution time, because of the shorter burst that keep coming.

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.

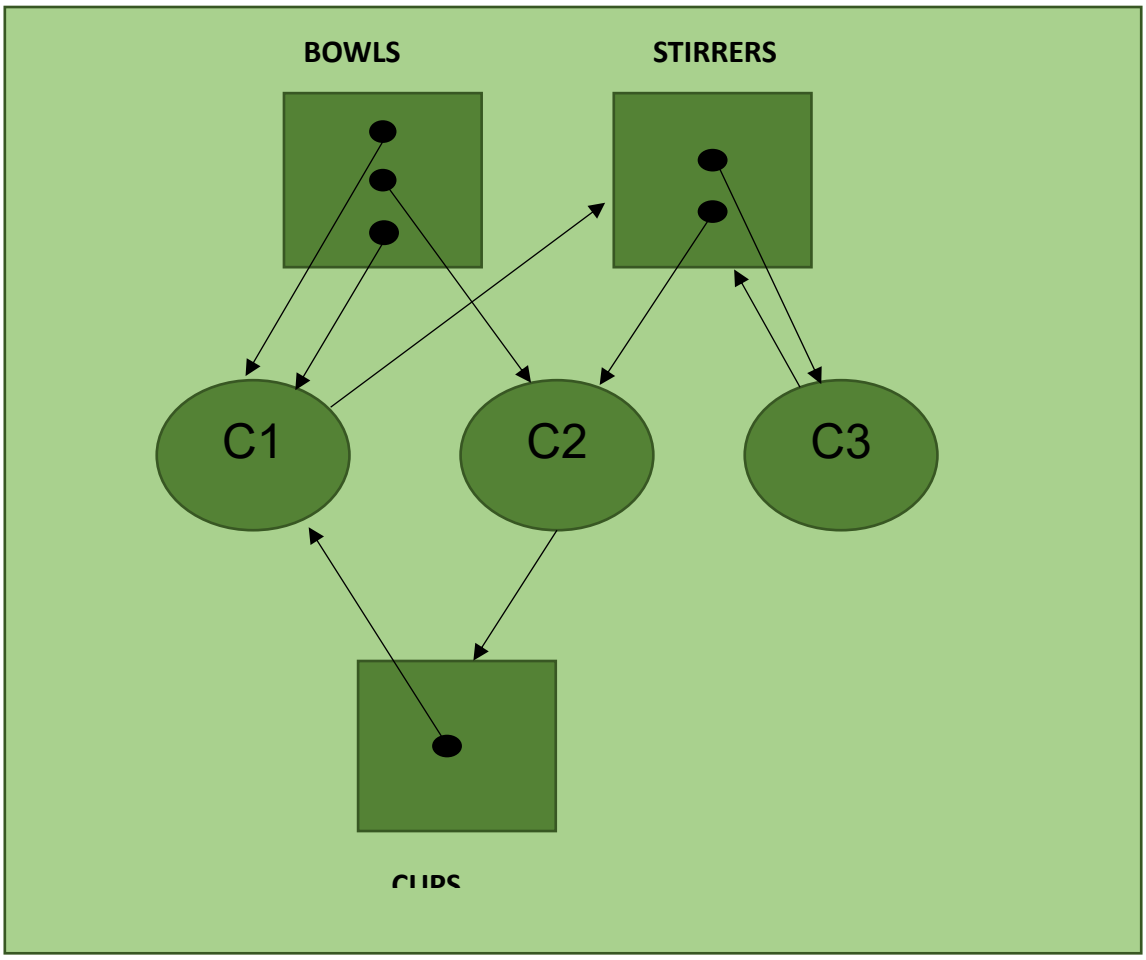
(i) There will be bad consequences when a deadlock occurs in the kitchen. Your boss is very concerned and ask you in theory the conditions for deadlock to occur. Answer your boss.
(ii) Draw a Resource Allocation Graph (RAG) to describe the resource allocation situation outlined above.
(iii) Based on 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.

ANSWERS:

i. The conditions for a deadlock to occur include;

- **Mutual Exclusion**: This means that a resource is held by only one process at a time.
- **Hold and Wait**: This means that a process can hold some number of resources at a time, and still request for resources being held by other processes at the same time.
- **No pre-emption**: This means that a resource cannot just be taken from a process, it has to be released by the process.
- **Circular wait**: This means that a deadlock can happen when processes wait for each other in a circular way.

ii.



iii. In the resource allocation graph above, a deadlock occurred. Deadlocks occur when processes share resources. In this situation, a chef (being the process) holds at least one resource(the bowls ,stirrers and cup), while it waits for another resource held by another chef. A resource cannot be taken from a chef, unless it is released by the chef holding the resource. The chef will have to wait indefinitely for the

resource that it needs until it is released by the chef holding it. Till then, he wont be able to proceed with his work.

iv. In a real computer system, the chefs are the processes, the bowls, cups and stirrers are the resources. Taken that ; the bowls are resource R1, the stirrers are resource R2 and the measuring cup is resource R3. The process C1 is holding two of resource R1, one of resource R3 and requesting for one of resource R2. Process C2 is holding one of resource R1, one of resource R2 and requesting for resource R3. Process C3 is holding one of resource R2 and requesting for one of resource R2. Process C1 won't proceed without resource R2 so it waits indefinitely for resource R2 to be released by the processes holding it, C2 and C3. Process C2 won't proceed without resource R3, and would wait indefinitely for it to be released by process C1, but process C1 won't release resource R3 until it has gotten resource R2. Process C3 won't proceed without resource R2, and would wait indefinitely for process C2 to release it, but C2 won't be able able to release it until it has gotten resource R3. While a process is waiting for a resource to be released, it remains in the waiting state and does not go into the running state .

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).

ANSWERS:

i. The performance metric that would be most relevant when selecting the most suitable CPU scheduling algorithm for the server, is the completion time.
   The completion time is the point at which a process is done executing and gets terminated. A low completion time increases the efficiency in a system. As there are over a thousand customers using the online computer booking system, it is expected that the web server would experience high traffic.
   - With a low completion time, processes are completed quickly. This means that customers will experience reduced latency.
   - The system would be able to make good use of available resources, because of the low completion time.
   - A low completion time, allows the system to process more requests within a given period of time than it normally would.
   - Lastly, considering the fact that the web server would be serving multiple clients at the same time, a low completion time allows for a fair distribution of CPU time among tasks.

ii. To support more customers, it would be more important to increase the size of the main memory. Considering the fact that over a thousand customers are expected to use the system at the same time, to support even more customers, the server is prone to have high traffic. This means that we need the server to be able to handle more tasks efficiently.

- Increasing the size of the main memory allows the web server to handle more users by providing more space for data caching. The web server caches more data leading to a quicker response time. This is becuase data that is frequently accessed, will be readily available in memory.
- Increasing the size of the memory allows the web server to manage more user connections simultaneously without running out of resources.
- Increasing the size of the main memory will improve the application performance. This is because they have more space to store and manage data being processed. This reduces the delay/latency in processing user requests.
- Increasing the size of the main memory allows the web server to improve its use of the system resources and allows the web server support more users.

The use of a more powerful CPU, would be a very useful upgrade, but factors like data caching, network bandwidth, memory may become hurdles. Just enhancing the software to use caching and reducing the need to read from or write to a disk, is enough to cause a significant impact on the performance of he web server without the need of a more powerful CPU, in some cases.

iii. Based on the article, web caching is a suitable solution to achieve a low completion time.

- Web caching involves downloading and storing shared page elements near the user so that the browser can collect the data from the storage location. This helps to decrease network traffic ,which reduces completion time, and thereby improves the performance of a web page.
- Web caching reduces load time which reduces latency. It does this by reducing the number of requests made to the origin server. This reduces completion time and therefore ,improves the efficiency of the web server.
- Web caching reduces network congestion as it creates more space and decreases the load on the origin server, which leads to low completion time. This is because with low completion time there is fair distribution of time among tasks. This speeds up operations in the web server as it helps non-cached content to be served quicker.

1

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. THE PROGRAM CODE:

@echo off

REM This batch scrpit prompts a user for the name of a batch file,and adds two commments the author/date and time to the start of the file

REM Prompt the user for the name of a batch file

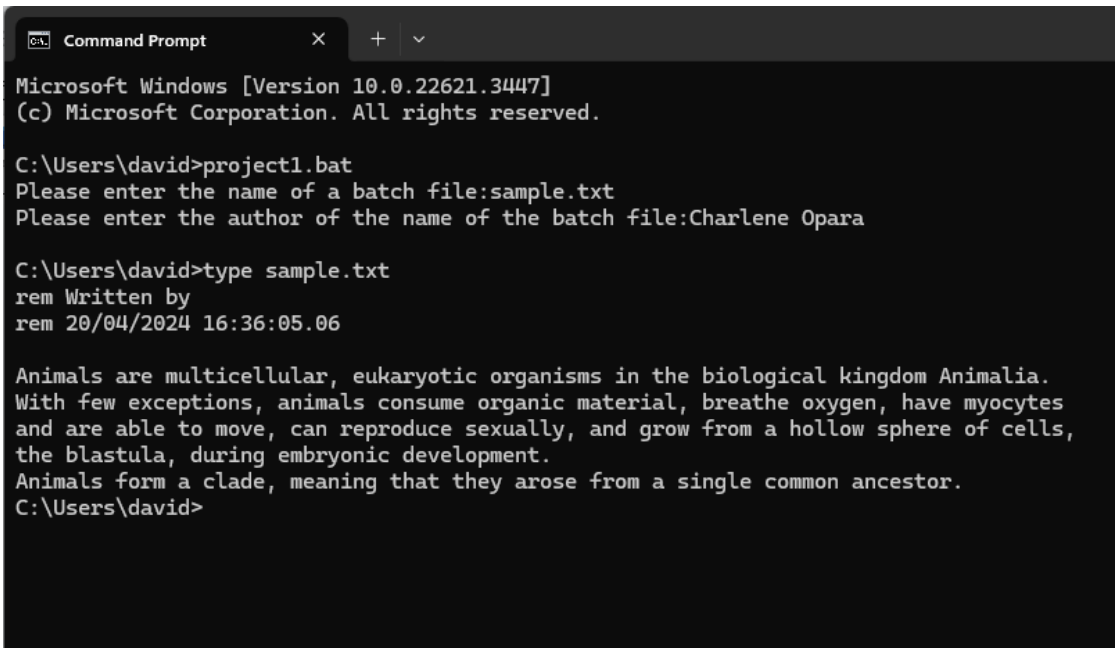
set /p name="Please enter the name of a batch file:"

```
REM Assign a tempoprary file to a variable
set file="temp.tmp"

REM Check if the file exists
if exist %name% (
    REM Prompt the user for the author of the name of the batch file
    set /p author="Please enter the author of the name of the batch file:"

    REM Overwrite the temporary file by redirecting the content in the inputted file to the temporary file
    type "%name%" > !file!

    REM Overwrite the content in the file with "Written by (the author's name)"
    echo rem Written by %author% > "%name%"

    REM Redirect the date and time to the file, this will make it appearon the second line after the first comment
    echo rem %date% %time% >> %name%
```

```
    REM Redirect the original contents of the file from the temporary file back
the the original file
    type "!file!" >> %name%


    REM Program is terminated
    exit /b


) else (
    REM If file doesnt exist, display an error message
    echo Error
    echo File does not exist
    exit /b
)


REM delete variables
file=
name=
```

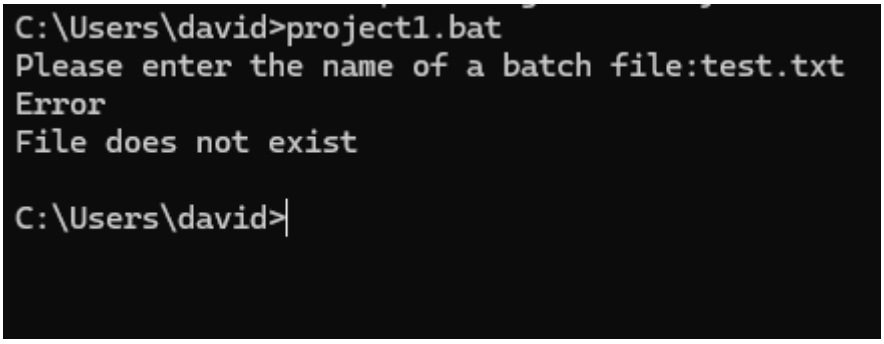
TEST CASES:

- IF FILE EXISTS: the program prompts the user for the name of a batch a file and checks if it exists. It goes further to prompt the user for the author of the name of the batch file, and reads the date and time, and then displays both as comments in the first two lines of the file.



- IF FILE DOES NOT EXIST

If the file does not exist, the program prints an error message and is then terminated



ii. PROGRAM CODE:

```
#!/bin/bash

#Counter for the loop is set to 0
choice=0

#Loop through the menu, till the user types "4"
while [ "$choice" != 4 ]
do
  #Display the menu
  echo "Menu:"
  echo "1.Copy file"
  echo "2. Rename file"
  echo "3.Create directory"
```

```
  echo "4.Quit"
  echo "Please pick an option"

  #Prompt the user to pick an option
  read choice

  #For each case the user picks, various operations will be carried out
  case $choice in
    #Case 1
    1)
      #Prompt the user for one parameter
      if [ "$1" = "" ]
      then
        #Prompt the user for the name of the file he wants to copy
        echo "Please enter the file you would like to copy:"
        read file1
#If file exists, prompt the user for the file he would like to copy to
        if [ -f $file1 ]
        then
           echo "Please enter the file you would like to copy $file1 to:"
           read file2
        else
           #If file does not exist, go back to the menu
           echo "File does not exist"
           continue
```

```
        fi


        #If file exist, copy file
        if [ -f $file2 ]
        then
          cp $file1 $file2
          echo "Copying..."
          echo "File has been successfully copied"
          continue
        else
          #If file does not exist, go back to the menu
          echo "File does not exist"
          continue
        fi
      #Set the variables to take one parameter
      else
        file1=$1
        file2=$1
      fi
    ;;
#Case 2
    2) #Prompt the user for one parameter
      if [ "$1" = "" ]
      then
        #Prompt the user for the name of a file he would like to rename
```

```
        echo "Please enter the file you would like to rename:"
        read file3

        #If the file exists, it prompts the user for the to rename to
        if [ -f $file3 ]
        then
          echo "Please enter the file you would like to rename $file3 to:"
          read file4
        #If file does not exist,it goes back to the menu
        else
          echo "File does not exist"
          continue
        fi

        #If the second file exists, it renames the file and goes back to the main
menu
        if [ -f $file4 ]
        then
          mv "$file3" "$file4"
          echo "Renaming..."
          echo "File has been renamed successfully"
          continue
        #If not ,it just goes back to the main menu
        else
          echo "File does not exist"
```

```
            continue
fi
        #Set the variables to take one parameter
        else
          file3=$1
          file4=$1
        fi
       ;;

      #Case 3
      3)
        #Prompt the user for one parameter
        if [ "$1" = "" ]
        then
           #Prompt the user for a directory name
           echo "Please enter the name of the directory you would like to create"
           read dirct

           #If directory already exists, it prints that it already exists and it goes
back to the menu
           if [ -d $dirct ]
           then
             echo "This directory already exists"
             continue
          #If is doesnt exist, it creates the directory and goes back to the menu
```

```
            else
              mkdir $dirct
             echo "Directory has been created"
             continue
            fi
        #Sets variable to take one parameter
        else
dirct=$1
        fi
    ;;

    #Case 4
    4)
      #Prints exiting and exits the program
      echo "Exiting..."
      exit
     ;;
  esac
((choice++))
done

# delete variables
file1=""
file2=""
file3=""
```

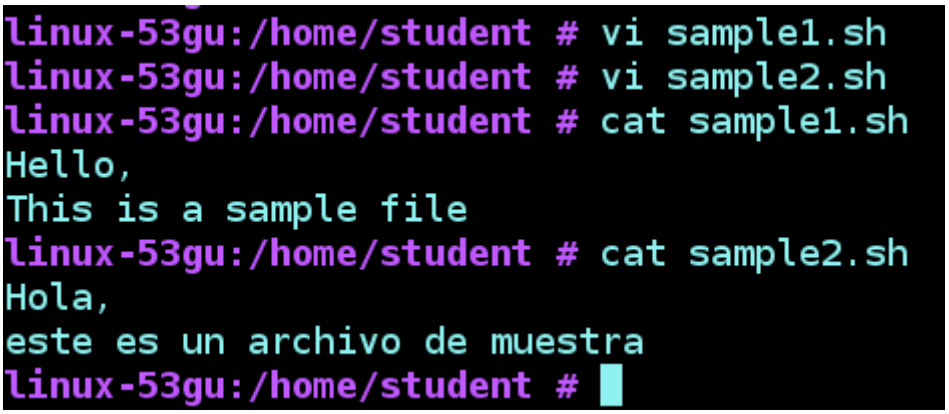
file4=""

choice=""

dirct=""

TEST CASES:

- In this illustration, we are going to be working with two sample files I created, ‘sample1.sh’ and ‘sample2.sh’. In the image below, i displayed the content of this two files.



- The program displays a menu of 4 options and loops through the menu till option is picked. In the image below, after it displays the menu and prompts the user to pick an option.
- If the user picks option 1, it prompts the user for a file to copy and a file to copy to and checks if they exist. If they exist, the program then copies the first file to the second file and goes back to the main menu. The content in the first file was copied to the second file

```
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
1
Please enter the file you would like to copy:
sample1.sh
Please enter the file you would like to copy sample1.sh to:
sample2.sh
Copying...
File has been successfully copied
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
```

```
linux-53gu:/home/student # cat sample1.sh
Hello,
This is a sample file
linux-53gu:/home/student # cat sample2.sh
Hello,
This is a sample file
linux-53gu:/home/student # ./project1.sh
```

- If the user picks option 2, it prompts the user for a file to rename and a file to rename the first file to, and checks if they exist. If they exist, the program then renames the first file to the second file and goes back to the main menu. This makes the first file to be non-existent as its name was changed to that of the second file

```
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
2
Please enter the file you would like to rename:
sample1.sh
Please enter the file you would like to rename sample1.sh to:
sample2.sh
Renaming...
File has been renamed successfully
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
linux-53gu:/home/student # cat sample1.sh
cat: sample1.sh: No such file or directory
linux-53gu:/home/student # cat sample2.sh
Hello,
This is a sample file
linux-53gu:/home/student #
```

- If the user picks option 3, it prompts the user for the name of a directory, and if it doesnt already exist, it creates a new directory.

```
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
3
Please enter the name of the directory you would like to create
test
Directory has been created
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
^C
linux-53gu:/home/student # 
drwxr-xr-x 2 root    root   4096 2024-04-22 03:46 test
-rw-r--r-- 1 root    root     30 2024-04-22 03:10 text2.sh
-rw-r--r-- 1 root    root     30 2024-04-22 02:59 text.sh
drwx------ 4 student users  4096 2012-12-30 17:49 .thumbnails
drwxr-xr-x 2 student users  4096 2012-12-30 17:39 usbmount_files
-rw-r--r-- 1 student users 41382 2012-12-30 17:39 usbmount.htm
drwxr-xr-x 2 student users  4096 2012-10-16 03:42 Videos
-rw-r--r-- 1 student users  1002 2012-10-16 04:39 .vimrc
-rw------- 1 student users   214 2024-04-22 02:37 .Xauthority
-rw-r--r-- 1 student users  1940 2012-10-16 04:39 .xim.template
drwxr-xr-x 2 student users  4096 2024-04-22 02:37 .xine
-rwxr-xr-x 1 student users  1446 2012-10-16 04:39 .xinitrc.template
-rw------- 1 student users 34182 2024-04-22 03:46 .xsession-errors
linux-53gu:/home/student # S
```

(the directory is on the first line)

- If the user chooses option 4, the program is terminated.

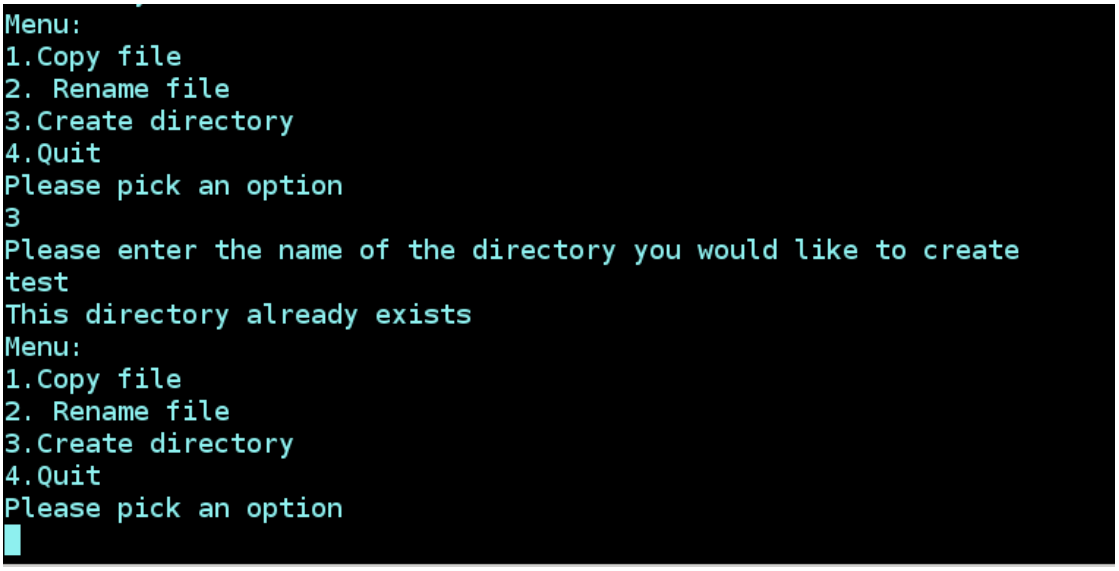
```
linux-53gu:/home/student # ./project1.sh
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
4
Exiting...
linux-53gu:/home/student #
```

- If the files doesn't exist, the program prints an error message and goes back to the main menu

```
linux-53gu:/home/student # ./project1.sh
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
1
Please enter the file you would like to copy:
smile.txt
File does not exist
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
1
Please enter the file you would like to copy:
sample2.sh
Please enter the file you would like to copy sample2.sh to:
file.sh
File does not exist
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
```

```
linux-53gu:/home/student # ./project1.sh
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
2
Please enter the file you would like to rename:
general.sh
File does not exist
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
2
Please enter the file you would like to rename:
sample2.sh
Please enter the file you would like to rename sample2.sh to:
pie.sh
File does not exist
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
```

- If directory exits ,the program prints “directory already exist” and goes back to the main menu


Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option
3
Please enter the name of the directory you would like to create
test
This directory already exists
Menu:
1.Copy file
2. Rename file
3.Create directory
4.Quit
Please pick an option

References:

i) Baeldung (2023) *Process Lifecycle*. Available at: https://www.baeldung.com/cs/process-lifecycle (Accessed:10/03/2024).

ii) Geeks for Geeks (2024) *States of a Process in Operating Systems.* Available at: https://www.geeksforgeeks.org/states-of-a-process-in-operating-systems/ (Accessed: 11/03/2024).

iii) Geeks for Geeks (2024) *Introduction of Deadlock in Operating System* Available at: https://www.geeksforgeeks.org/introduction-of-deadlock-in-operating-system/ (Accessed: 18/03/2024).

iv) Baeldung (2024) *Deadlock*. Available at: https://www.baeldung.com/cs/os-deadlock (Accessed:19/03/2024).

v) Geeks for Geeks (2024) *Starvation and Aging in Operating Systems.* Available at: https://www.geeksforgeeks.org/starvation-and-aging-in-operating-systems/ (Accessed: 21/03/2024).

vi) Baeldung (2024) *CPU Scheduling: Arrival, Burst, Completion, Turnaround, Waiting, and Response Time*. Available at: https://www.baeldung.com/cs/cpu-scheduling (Accessed: 21/03/2024).

vii) Geeks for Geeks (2023) *Program for Shortest Job First (or SJF) CPU Scheduling | Set 1 (Non-preemptive).* Available at: https://www.geeksforgeeks.org/program-for-shortest-job-first-or-sjf-cpu-scheduling-set-1-non-preemptive/ (Accessed: 21/03/2024).

viii) Baeldung (2024) *CPU Scheduling: Arrival, Burst, Completion, Turnaround, Waiting and Response Time.* Available at: https://www.baeldung.com/cs/cpu-scheduling (Accessed: 20/04/2024).

ix) Hostadvice (2023) *How to Determine the Correct Size and Type of a Web Server.* Available at: https://hostadvice.com/how-to-/web=hosting/how-to-determine-the-correct-size-and-type-of-a-web-server/ (Accessed: 21/04/2024).