Module Code & Module Title

CS5001NA Networks and Operating System

Assessment Weightage & Type

20% Individual Coursework

Year and Semester

2020-21 Autumn

Student Name: Rhythm Sapkota

London Met ID: 20049417

College ID: np01cp4s210220

Assignment Due Date: 25th April, 2022

Assignment Submission Date: Week 24

Word Count (TASK B):

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Table of Contents

Task A

Introduction

After successfully creating a software in Linux, the current report was completed. Linux, being an open source project, has a large number of users who work on it and contribute to its development. Each and every server, PC, and smart phone runs on Linux. Linux is a versatile operating system that may be used both at home and away from home. We utilized Ubuntu for our subsequent development since Ubuntu is a particular version of Linux.

Unix is a multi-tasking, operating system developed by AT&T, and as an operating system, it can only be used by its copywriters during the coding process. It's mostly found in workstations and servers.

The program has some requirements; for example, it requires the user to provide two proper parameters: username and id; if these two parameters are not provided, the program will not let you in. It also requires a password, after which you will be asked to choose the best band, after which you will be given a short description about the band you choose, you will be allowed to choose three band members codes, which include the numbering of the band members, and after selecting any one number, you will be given a description about that band member, and finally the program will ask if you want to continue or not, and if you say yes, the program will repeat itself and if not then the program will terminate.

All of these tasks were completed by the help of lecture videos, slides, and other types of websites. This was a good program that taught us a lot of new stuff as well.

Script

```
1.  #!/bin/bash
2.
3.  # Creating the secret key
4.  secretKey="kurt1234"
5.  passKey="passkey"
6.  passTry=0
7.
8.
9.  until [ $secretKey = $passKey ]
10. do
11.   if [ $passTry -ge 3 ]
12.    then
13.             exit 1
14.   fi
15.   echo -e "Enter your Password: \c"
16.   read -s passKey
17.   echo
18.   (( passTry++ ))
19.
20.   if [ -z $passKey ]
21.    then
22.             passKey="passKey"
23.             echo "!!!Incorrect password. Please Retry!!!"
24.             echo
25.   else
26.             if [ $secretKey != $passKey ]
27.              then
28.                       echo "!!!Incorrect Password!!!"
29.                       echo
30.             else
31.                       echo "Password matched!"
32.                       echo
33.             fi
34.   fi
35. done
36.
37. # making the function for displaying the list of five band names
38. listBand () {
39.   echo
40.   echo
41.   echo "The names and codes of  five bands are listed below:"
42.   echo "                  _____________________________"
43.   echo "                 |    Band Name    |     Code     |"
44.   echo "                 |-----------------|--------------|"
45.   echo "                 |     Beatles     |     BEA      |"
46.   echo "                 |     AC/DC       |     AD       |"
47.   echo "                 |     Queen       |     QUE      |"
48.   echo "                 |     Blondie     |     BLO      |"
49.   echo "                 |     Nirvana     |     NIR      |"
50.   echo "                 |_________________|______________|"
51.   echo
52. }
53.
54. # function for validating band name
55. validateBand () {
56.   while [ true ]
57.   do
58.             echo -e "Please guess required best band >> \c"
59.             read bandName
60.             if [ -z $bandName ]
61.              then
62.                       echo "!!!Please enter the name of your best band!!!"
63.             else
64.                       if [[ $bandName = "BEA" || $bandName = "AD" || $bandName = "QUE" ||
    $bandName = "BLO" || $bandName = "NIR" ]]
65.                        then
```

```
                                if [ $bandName = "NIR" ]
                                 then
                                        echo
                                        echo "Correct Guess!!!"
                                        echo -e "Nirvana was a rock band from the United States that formed in 1987 in Aberdeen, Washington. Founded by Kurt Cobain, lead vocalist and guitarist, and Krist Novoselic, bassist, the band went through a number of drummers, most notably Chad Channing, until bringing in Dave Grohl in 1990. Nirvana's popularity popularized alternative rock, and they were sometimes referred to as Generation X's flagship band. Their music has a devoted fanbase and continues to have an impact on current rock culture."
                                        break
                                else
                                        echo
                                        echo "Wrong Guess!!! Try again!!!"
                                fi
                        else
                                echo
                                echo "!!!Invalid band code. Please choose the valid band code!!!"
                                continue
                        fi
                fi
  done
}

#making the function to list members
listMember () {
  echo
  echo
  echo "The names and codes of five band members are listed below:"
  echo "                  _____________________________________"
  echo "                 |     Player                |     Code    |"
  echo "                 |---------------------------|-------------|"
  echo "                 |     John Lennon           |     JL      |"
  echo "                 |     Angus Young           |     AY      |"
  echo "                 |     Freddie Mercury       |     FM      |"
  echo "                 |     Debbie Harry          |     DH      |"
  echo "                 |     Kurt Cobain           |     KC      |"
  echo "                 |___________________________|_____________|"
  echo
          echo "Please select any three band members code each separated by a space."
  }


  #making the function to read the three band members input from user
  chooseMember () {
          choose=false
          while [ $choose = "false" ]
          do
                  echo -e "Choose  any three members code from band members code. >> \c"
                  read -a members

                  if [ ${#members[@]} -eq 3 ]
                   then
                          for member in "${members[@]}";
                          do
                                  if [[ $member = "JL" || $member = "AY" || $member = "FM" || $member = "DH" || $member = "KC" ]]
                                   then
                                          valid=true
                                          continue
                                  else
                                          valid=false
                                          break
                                  fi
                          done

                          if [ $valid = "false" ]
```

```
127.                                 then
128.                                         echo "Invalid Code, Please choose the valid
    band members code."
129.                                 else
130.                                         choose=true
131.                                 fi
132.                         else
133.                                 echo
134.                                 echo "Please enter only 3 band members."
135.                         fi
136.             done
137.   }
138.
139.   #making the method for creating the menu
140.   memberMenu () {
141.             chooseMember
142.             echo
143.             PS3="Please select a number of specific band member from the menu above to
    see the details. >> "
144.             select file in "${members[@]}"
145.             do
146.                         if [ -z $file ]
147.                          then
148.                                 echo
149.                                 echo "Please select a band member number.!!!"
150.                         else
151.                                 echo
152.                                 case $file in
153.                                         JL) if [ -f JL ]
154.                                              then
155.                                                   cat JL
156.                                                   break
157.                                             else
158.                                                   echo "This is not a file"
159.                                             fi;;
160.                                         AY) if [ -f AY ]
161.                                              then
162.                                                   cat AY
163.                                                   break
164.                                             else
165.                                                   echo "This is not a file"
166.                                             fi;;
167.                                         FM) if [ -f FM ]
168.                                              then
169.                                                   cat FM
170.                                                   break
171.                                             else
172.                                                   echo "This is not a file"
173.                                             fi;;
174.                                         DH) if [ -f DH ]
175.                                              then
176.                                                   cat DH
177.                                                   break
178.                                             else
179.                                                   echo "This is not a file"
180.                                             fi;;
181.                                         KC) if [ -f KC ]
182.                                              then
183.                                                   cat KC
184.                                                   break
185.                                             else
186.                                                   echo "This is not a file"
187.                                             fi;;
188.                                         *) echo "Selected file not understandable";;

189.                                 esac
190.                         fi
191.             done
192.   }
193.
```

```
194.  if [ $# -eq 2 ]
195.   then
196.           if [[ $1 =~ [Aa-Zz]+$ && $2 =~ [0-9]+$ ]]
197.            then
198.                    echo " _____________________________________________ "
199.                    echo "|                                             |"
200.                    echo "    London Met Id: $2"
201.                    echo "    Name: $1"
202.                    echo "    Date and Time: " $(date +%F) $(date +%T)
203.                    echo "|_____________________________________________|"
204.
205.                    continue=true
206.
207.                    while [ $continue = "true" ]
208.                    do
209.                            listBand
210.                            validateBand
211.                            listMember
212.                            memberMenu
213.                            echo
214.
215.                            will=1
216.                            until [[ $will = 'y' || $will = 'Y' || $will = 'n' ||
    $will = 'N' ]]
217.                            do
218.                                    echo -e "Please Enter Y to continue the program
    and N to exit >> \c"
219.                                    read will
220.                            done
221.
222.                            case $will in
223.                                    y|Y) continue;;
224.                                    n|N) echo "!!! Thank u for coming have a good
    day !!!"
225.                                         break;;
226.                                    *) echo "invalid option. please retry!!!"
227.                            esac
228.                    done
229.           else
230.                    echo "Please enter in correct order. First Name and then enter ID"
231.           fi
232.  elif [ $# -gt 2 ]
233.   then
234.           echo "Please enter only your Name and ID as a parameter."
235.           echo
236.  else
237.           echo "Enter your Name and your ID."
238.           echo
239.  fi
240.
```

Testing

Test 1: Run without username:

Test No.	1
Input	The program was run without parameters.
Expected Output	The program will display error when parameters are not given.
Actual Output	The program displayed error and asked user to re-run the program with correct parameters.
Test Result	The test was successful after keeping correct parameters.

Table 1: Run without username

Screenshots

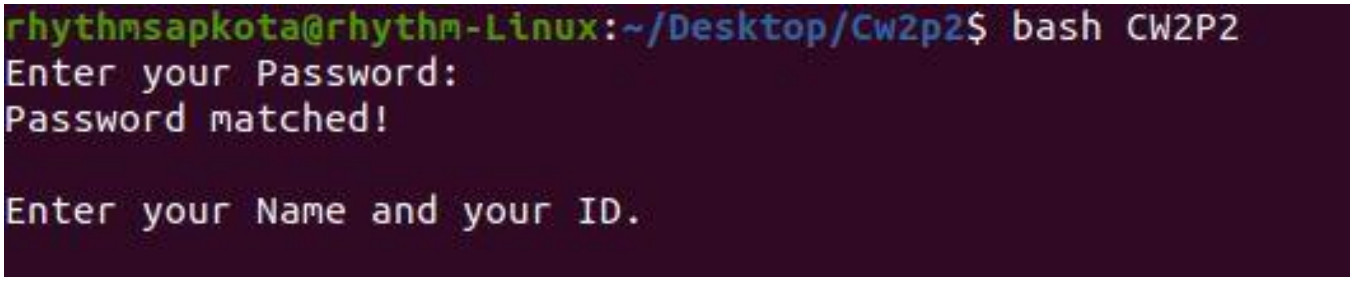


Figure 1: Error message was shown saying enter your name and ID.

Test 2: Run with username and id

Test No.	2
Input	The program was run with correct parameters.
Expected Output	The program will run when correct parameter is given.
Actual Output	The program successfully ran with keeping correct parameters.
Test Result	The test was successful after the correct parameters

Table 2: Run with username and id

Screenshots

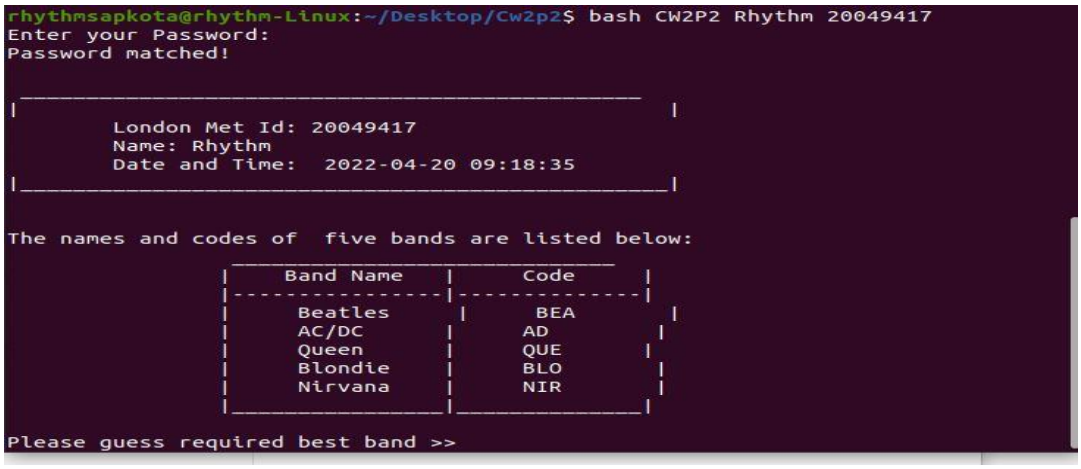


Figure 2: Program ran successfully with correct parameters.

Test 3: run incorrect password 3 times

Test No.	3
Input	Wrong passcode was entered for Three times.
Expected Output	The program will ask up to 3 times for correct password and will terminate if it is still wrong.
Actual Output	The program asked 3 times to re enter the passcode and then terminated.
Test Result	The test was successful.

Table 3: run incorrect password 3 times

Screenshots

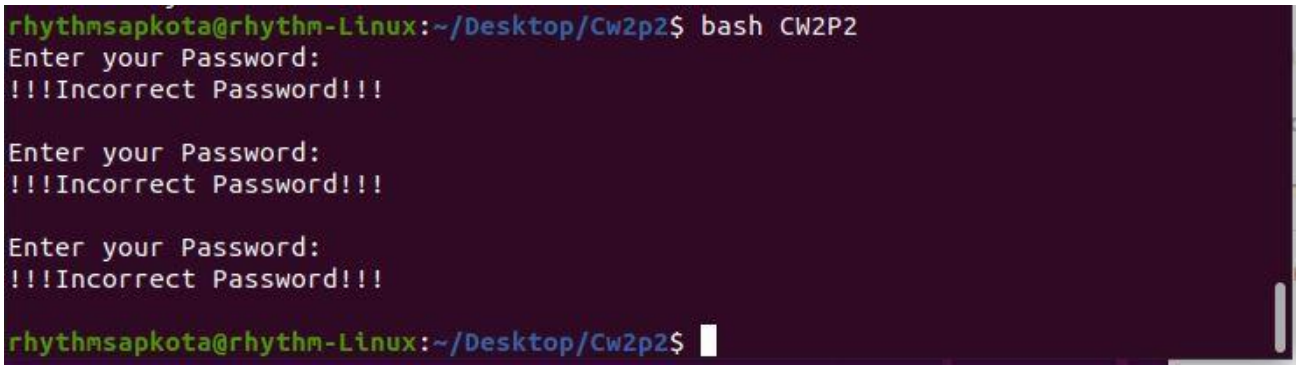


Figure 3: Incorrect password was given 3 times and program was terminated.

Test 4: run correct password

Test No.	4
Input	To run program with correct password
Expected Output	The program will ask for password, after entering correct password the program will continue.
Actual Output	The program asked for password and after correct password, the program continues.
Test Result	Test was successful.

Table 4: run correct password

Screenshots:

Figure 4: Correct password was given and program ran.

Test 5: Run the program with full Band name.

Test No.	5
Input	Run the program with full Band name.
Expected Output	The program was run with full Band name, and was unsuccessful.
Actual Output	The program displayed invalid option after running with full band name.
Test Result	The test was successful.

Table 5: Run the program with full Band name.

Screenshots:

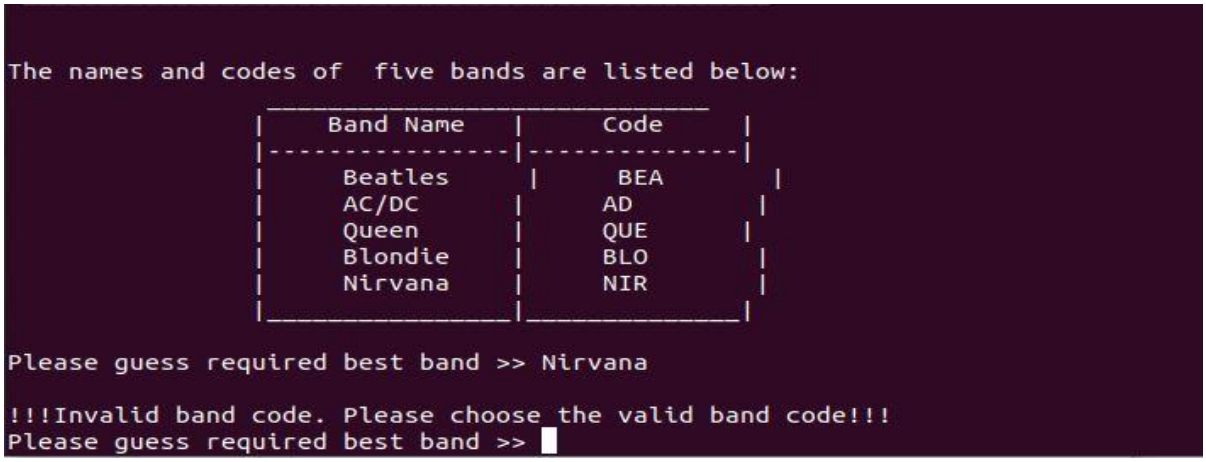


Figure 5: error message was shown after entering the full name of the band.

Test 6: incorrect band CODE

Test No.	6
Input	Run program with incorrect Band code.
Expected Output	The program will run with incorrect band code , displaying error message.
Actual Output	The program was run with incorrect band code , which displayed error message.
Test Result	The test was successful.

Table 6: incorrect band CODE

Screenshots:

Figure 6: error message was shown after guessing the wrong band code.

Test 7: correct band CODE

Test No.	7
Input	Run program with correct Band code.
Expected Output	The program needs to run with correct band code.
Actual Output	The program was run with correct band code displaying some information.
Test Result	The test was successful.

Table 7 correct band CODE

Screenshots:

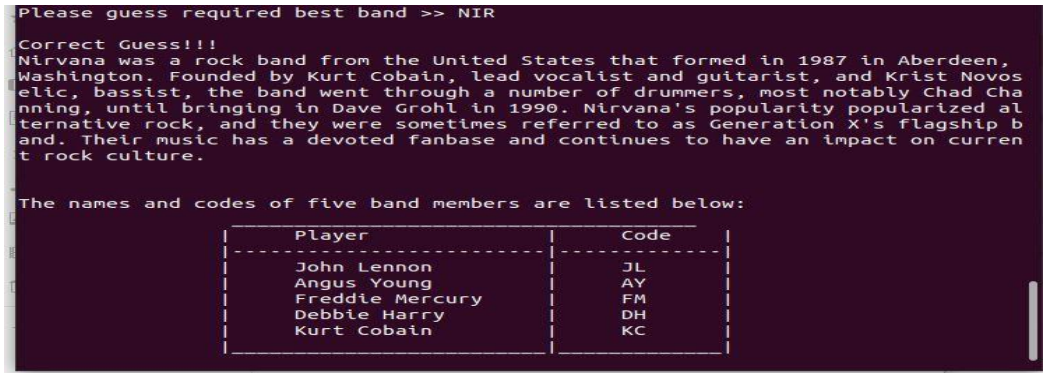


Figure 7: After the correct guess A small description of the band was shown and messgare was shown.

Test 8: pick 4 band member Codes.

Test No.	8
Input	To pick 4 band member's codes.
Expected Output	The 4 members code was serially taken and then error messaged was displayed.
Actual Output	The error message was obtained after entering 4 band members code.
Test Result	The test was successful.

Table 8: pick 4 band member Codes.

Screenshots:

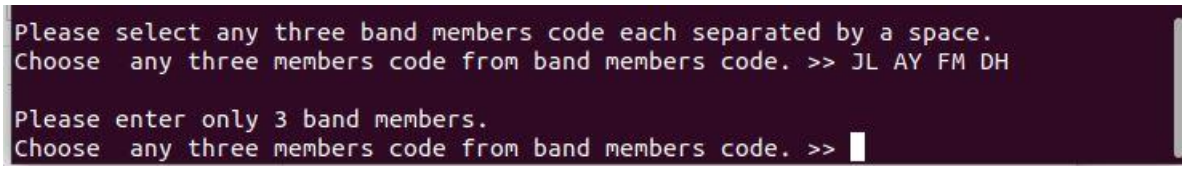


Figure 8: 4 band members code were given and error message was shown.

Test 9: Pick a random code that is not in the band member table.

Test No.	10
Input	To pick a random code which is not listed and not associated.
Expected Output	Different band members code is to be given and error message to be shown.
Actual Output	Different band members code was given and error message was shown.
Test Result	The test was successful.

Table 9: Pick a random code that is not in the band member table.

Screenshots

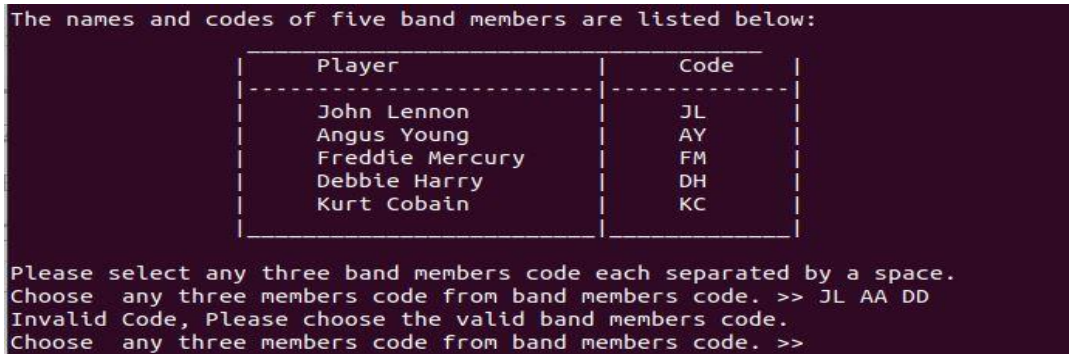


Figure 9: error message was shown after entering a random code.

Test 10: wrong user id.(parameter validation)

Test No.	10
Input	To run the wrong username and user id.
Expected Output	Wrong parameter was given and the error message was displayed.
Actual Output	Wrong username and id were given which resulted in error message.
Test Result	The test was successful.

Table 10: wrong user id.(parameter validation)

Screenshots:

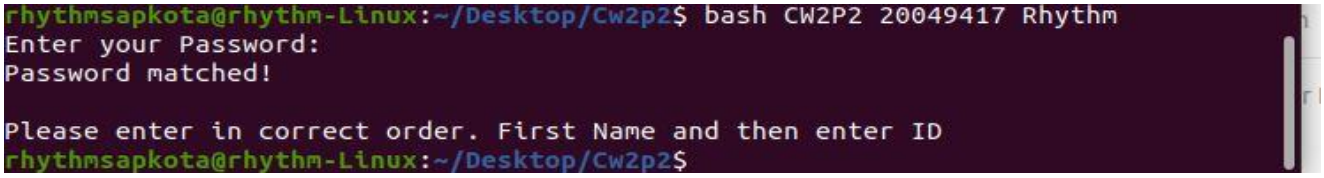


Figure 10: Error message was shown.

Test 11: right user id without username

Test No.	11
Input	Right user id was given without username
Expected Output	The program was run by giving only one parameter which would result in error message.
Actual Output	The error message was shown while giving only one parameter.
Test Result	The test was successful.

Table 11: right user id without username

Screenshots:

```
rhythmsapkota@rhythm-Linux:~/Desktop/Cw2p2$ bash CW2P2 20049417
Enter your Password:
Password matched!

Enter your Name and your ID.
```

Figure 11: Error message was shown.

Test 12: NO External File of member(except 3 profile members that you have made) or Invalid member ID.

Test No.	12
Input	NO External File of member or Invalid member ID.
Expected Output	The program should not have extra information except 3 three band members.
Actual Output	Other than the three members extra members are not valid.
Test Result	The test was successful.

Table 12: NO External File of member(except 3 profile members that you have made)

or Invalid member ID.

Screenshots

```
Please select any three band members code each separated by a space.
Choose  any three members code from band members code. >> JL AY FM

1) JL
2) AY
3) FM
Please select a number of specific band member from the menu above to see the de
tails. >> 1

John Winston Ono Lennon (born John Winston Lennon; 9 October 1940 - 8 December 1
980) was an English singer, songwriter, musician, and peace campaigner who rose
to international prominence as the Beatles' founder, co-composer, co-lead vocali
st, and rhythm guitarist. In his music, writing, and paintings, as well as on fi
lm and in interviews, Lennon was known for his rebellious attitude and biting hu
mor. His songwriting collaboration with Paul McCartney is still one of the most
successful in history. [3]

As a teenager, Lennon became embroiled in the skiffle craze in Liverpool. He est
ablished the Quarrymen in 1956, which later became the Beatles in 1960. He was k
nown as "the intellectual Beatle" and served as the group's de facto leader unti
l McCartney took over.
(reference- wikipidia)
```

Figure 12: valid members description is shown through their file.

```
Please select any three band members code each separated by a space.
Choose  any three members code from band members code. >> JL DH KC

1) JL
2) DH
3) KC
Please select a number of specific band member from the menu above to see the de
tails. >> 2

This is not a file
Please select a number of specific band member from the menu above to see the de
tails. >> 3

This is not a file
```

Figure 13: Extra members were not valid

TEST 13: EXIT YES

Test No.	13
Input	To run the program again press Y and N to exit: yes, in program.
Expected Output	yes, would again repeat the program,
Actual Output	The run was successful.
Test Result	The test is successful.

Table 13: TEST 13: EXIT YES

Screenshots:

```
Please Enter Y to continue the program and N to exit >> Y


The names and codes of  five bands are listed below:
                _______________________________
               |    Band Name    |     Code     |
               |-----------------|--------------|
               |     Beatles     |     BEA      |
               |     AC/DC       |     AD       |
               |     Queen       |     QUE      |
               |     Blondie     |     BLO      |
               |     Nirvana     |     NIR      |
               |_________________|______________|

Please guess required best band >>
```

Figure 14: The program was repeated after entering Y.

TEST 14: EXIT NO

Test No.	14
Input	To run the program again press Y and N to exit: NO, in program.
Expected Output	The exit: no command was typed and the program would be terminated
Actual Output	The program was terminated after this command.
Test Result	The test is successful.

Table 14: EXIT NO COMMAND

Screenshots:

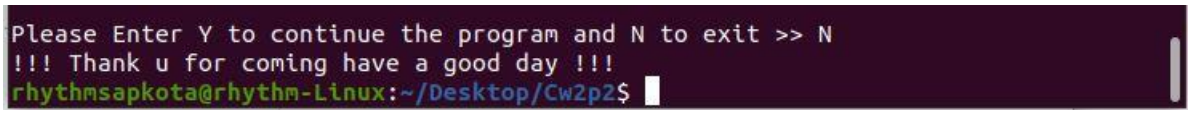


Figure 15: After pressing N the program was terminated showing a thank you message.

Contents of three files: (TEXTS)

JL

John Winston Ono Lennon (born John Winston Lennon; 9 October 1940 – 8 December 1980) was an English singer, songwriter, musician, and peace campaigner who rose to international prominence as the Beatles' founder, co-composer, co-lead vocalist, and rhythm guitarist. In his music, writing, and paintings, as well as on film and in interviews, Lennon was known for his rebellious attitude and biting humor. His songwriting collaboration with Paul McCartney is still one of the most successful in history.

As a teenager, Lennon became embroiled in the skiffle craze in Liverpool. He established the Quarrymen in 1956, which later became the Beatles in 1960. He was known as "the intellectual Beatle" and served as the group's de facto leader until McCartney took over.

(reference- wikipidia)

.

AY

Angus McKinnon Young (born March 31, 1955) is an Australian musician best known for co-founding, leading guitarist, composer, and being the only consistent original member of the Australian hard rock band AC/DC. He's noted for his high-energy performances, stage attire inspired by schoolboy uniforms, and his own take on Chuck Berry's duckwalk. Young was voted 24th among the 100 best guitarists of all time by Rolling Stone magazine.

(reference-wikipedia)

FM

Freddie Mercury (born Farrokh Bulsara; September 5, 1946 – November 24, 1991) was a British singer-songwriter best known for being the main vocalist of the rock band Queen. He was recognized for his flamboyant stage demeanor and four-octave vocal range, and was regarded as one of the best vocalists in the history of rock music. Mercury disregarded rock leader traditions with his theatrical manner, which influenced Queen's aesthetic direction.

(Reference- Wikipedia)

Conclusion

From this coursework We found shell scripting to be quite useful, and we were able to learn a lot from it. Shell scripting includes the bash shell, which is essential for coding. Bash is used in almost every shell and is also the default on most Linux systems. The coding portion was a little difficult, but with the support of our renowned teachers and lecturers, we were able to overcome it. Shell scripting was a completely new concept for us, but it became simpler as time went on. As IT students, we consider ourselves extremely fortunate to be able to participate in such amazing projects since they broaden our ideas and provide us with information on new topics. If we are interested in Linux, shell scripting is a very significant topic. Never the less, I'd like to thank Islington College for offering this form of platform for us; this type of assignment topic is really beneficial to us because it broadens our minds and especially assists us in the future. Mr. Nabin Acharaya, our tutor, has gone above and beyond to make our work simpler and he has also put in a lot of effort during our coursework session, and his suggestions have proven to be much more beneficial to us.

The coding section included a secret key, and after entering into the program, it was about selecting the best band and then going on to selecting three band members. A description was displayed, and the user was asked if they wanted to continue or not.

Task B

Introduction

A process is a running program. When we construct a C or C++ program, for example, the compiler generates binary code. Both the original and binary codes are programs. When we run the binary code, it turns into a process. When a single program is launched numerous times, it can generate multiple processes; for example, when we open a.exe or binary file multiple times, multiple instances are formed (multiple processes are created). (GeeksforGeeks, 2021)

Process management includes a variety of tasks, including process creation, scheduling, termination, and a deadlock. A process is a running program that is an important component of modern operating systems. The operating system must distribute resources to allow processes to interact and exchange data. It also secures each process' resources from other methods and allows for process synchronization (Guru99, 2022). Processes are managed by the operating system, which performs tasks like as resource allocation and process scheduling. When a process runs on a computer, the computer's Primary Memory and CPU are used (IncludeHelp, 2020).

In terms of Process Management, the operating system is responsible of the following functions:
Processes and threads on the CPUs are scheduled by operating system.

1) Both user and system processes can be created and deleted by Operating system.
2) Processes can be paused and resumed by the OS.
3) OS Provides synchronization methods for processes.
4) OS Provides communication mechanisms for processes.
   (Point, 2021)

Background

A process is basically an executing program. The execution of a process must be done in a specific order. A process is an object that represents the fundamental unit of work that must be implemented in the system.

At the early stage of Computer generation Hand programmed machine with simple software were avialiable. Then later with the development of OS different process management strategies and programs were developed. First batch processing was developed but it had so many issues and in 1965 multiprogramming CPU were developed which kept several jobs in main memory and multiplex CPU between jobs. It was in use for about 15-20 years and then Time sharing was developed.

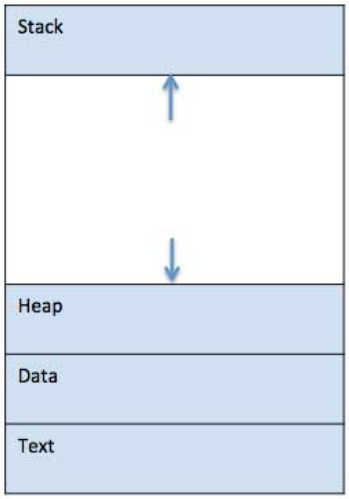
It was just a matter of time before time sharing became a reality. Only one program could be run at a time on early computers. It was just a matter of time before time sharing became a reality. This required greater management as well as increased compartmentalization. Despite all of these advancements, a computer still has a single CPU and a single program counter. A process is a running application that has its own virtual CPU. The real-time CPU alternates between processes.

But as of present day context process management has taken a huge step and is now more efficient and convenient. Modern day Process management involves a variety of activities, including process development, scheduling, termination, and a deadlock. A process is a running program that is an important component of current operating systems. The operating system must distribute resources to allow processes to interact and exchange data. Many functions are required in a modern multi-programming, time-sharing operating system.

Process Architecture

A program may be separated into four sections when it is loaded into memory and becomes a process: stack, heap, text, and data. The diagram below is a simplified representation of a process in main memory.

The following are some of the architecture designs of the Process:



- Stack's architecture: Temporary data such as function parameters, return addresses, and local variables are stored on the Stack.
- Heap: It allocates memory that could be processed during the program's runtime.
- Data: The variable is contained in the data.
- Text: The current activity is represented by the value of the Program Counter in the Text Section.

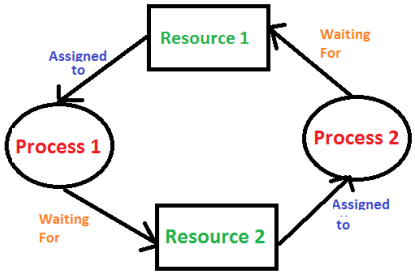
Process Hierarchies

When a process creates a child process, the parent and child processes are still linked in some ways. A process hierarchy is formed when a child process creates more processes.

A process group is formed by a process and all of its child processes and descendants in UNIX OS. When a user delivers a signal via the keyboard, it is sent to all members of the process group with whom the keyboard is currently attached. But Windows has no concept of a process hierarchy. Every process is the same. When a process is formed, the parent is given a specific token (called a handle) that it may use to manage the child, which is the only sign of a process hierarchy. It is, however, free to send this token to another process, thereby nullifying the hierarchy. In UNIX, processes cannot disinherit their children. (TANENBAUM, 2015)

Deadlock

A deadlock occurs when a group of processes is stalled because each process is holding a resource and waiting for another process to obtain it. (GeeksforGeeks, 2021)



For example:

In the figure, Process 1 is holding Resource 1 while Process 2 acquires Resource 2, while Process 2 is also waiting for Resource 1.

Some of the methods for handling deadlock are as follows:

1) Deadlock avoidance
2) Deadlock detection and recovery
3) Ignore the problem and reboot the system.(Both Windows and Unix take this approach.)

Process scheduling

Process Scheduling is a task in the operating system that schedules processes in various stages such as ready, waiting, and running. Process scheduling allows the operating system to assign each process a time period for CPU execution. Another significant advantage of employing a process scheduling system is that it keeps the CPU active at all times. (Guru99, 2022)

Process scheduling is an essential part of a Multiprogramming operating systems. Multiple processes can be loaded into executable memory at the same time in such operating systems, and the loaded processes share the CPU utilizing temporal multiplexing. (Tutorialspoint, 2021)

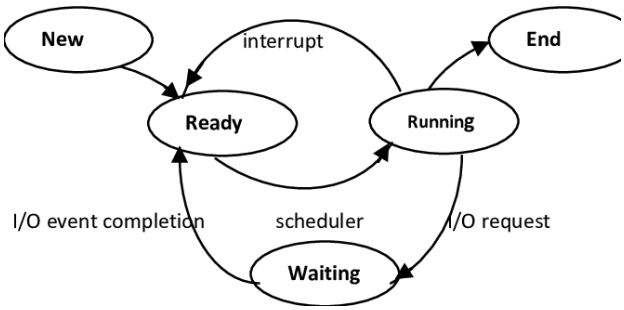
Process States

Despite the fact that each process is independent, with its own program counter and internal state, processes frequently interact with one another.

In the figure, we can see a state diagram showing the three states a process may be in:

They are:

a) Ready
b) Running
c) Waiting or Blocked



The first two states are logically related. The process is willing to execute in both circumstances; but, in the second, there is temporarily no CPU available for it. The process cannot run in the third state, which is fundamentally different from the previous two in that the CPU is idle and has nothing else to perform.

Here are the four transitions form the above figure:

1 Process blocks for input

2. Scheduler picks another process

3. Scheduler picks this process

4. Input becomes available

As illustrated, there are four possible transitions between these three phases. When the operating system realizes that a process can't continue right now, it enters Transition 1. In certain systems, the process can enter a blocked state by issuing a system call, such as pause. When a process reads from a pipe or special file (e.g., a terminal) when there is no input accessible on other systems, including UNIX, the process is immediately halted.

The following are examples of several types of process state models:

Two-state process Model:

There are two states in the two-state process block: not running and running. A two-state can be produced whether a process is being performed or not. When a process is created for the first time, it is in the not-running state. The process enters the running state when it is executed by the CPU. If a greater important process is detected, the current operating process becomes inactive, while the higher significance process becomes active. Every process that isn't currently running is placed in a queue. Because the dispatcher's process may still be waiting for an event or an I/O request to occur, using a two-state design can degrade performance.

Three state process Model:

In the three-state process paradigm, the not running stage is divided into two parts: ready state and waiting or blocked state. A process in the ready state is ready to run, whereas one in the delayed or blocked state is waiting for something to happen. The operating system has a separate queue for each state. However, in today's world, programs are so large that loading all processes into main memory is unfeasible. This problem is supposed to be solved using a five-state paradigm.

Five state process model:

In the five-state process paradigm, two new states emerge: new state and terminated state. The procedure is still in its infancy and hasn't been moved to main memory. A process has either been aborted or completed its execution while it is in the terminated state. One big downside of this paradigm is that if the processor executes at such a fast pace, all processes may move to the waiting stage, and no process will be ready. It's possible that this will result in minimal CPU utilization.

Six state process model:

The six-stage process model now has a sixth state: suspend. In the suspend state, the process in the waiting or block state is suspended and moved to secondary memory. Every process in the suspend state has been queued. With the aid of this state, the CPU can relocate another process to main memory. The CPU may not know which process in the suspension queue is ready for execution, which is a key downside of this paradigm.

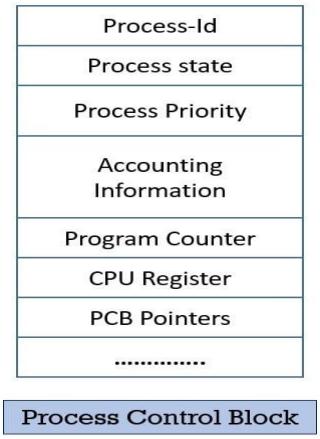
Seven state process Model:

In the seven-state process paradigm, the suspend stage is divided into two states: Blocked Suspended and Ready Suspended. In blocked suspend, the process in secondary memory is waiting and not ready to execute, but in ready suspended, the process is ready to execute.

Process Control Block (PCB)

A Process Control Block is a data structure that includes process-related information. The process control block is sometimes referred to as a task control block, process table entry, and so on. Process data structure is done in terms of the PCB, therefore it's critical for process management. It also specifies the operating system's current status.

Many data elements are stored in the process control for effective process management. Some of these data sets are presented using the diagram provided. (TutorialsPoint, 2021)



Policies and Algorithm

The various types of Policies and Algorithms in process Scheduling are as follows:

1 First come First Serve (FCFS)

First-Come-First-Served algorithm is the simplest scheduling algorithm is the simplest scheduling algorithm. Processes are dispatched in the order in which they appear in the ready queue. Because it is a non-pre-emptive discipline, once a process gets a CPU, it will run until it is finished. One of the scheme's primary flaws is that the average time is frequently rather long.

2 Shortest job first (SJF) scheduling

SJF is a non-pre-emptive discipline in which the waiting task (or process) with the shortest anticipated run-time-to-completion is executed first. To put it another way, when CPU becomes available, it is allocated to the process with the smallest next CPU burst. The obvious flaw in the SJF approach is that it necessitates accurate knowledge of how long a work or process will take, which is rarely accessible.

3 Priority Scheduling

Each process is given a priority, and the processes with the highest priority are permitted to execute. FCFS order is used to schedule Equal-Priority processes. A specific instance of the general priority scheduling method is the shortest-job-first (SJF) algorithm. Lower priority processes may starve and be postponed indefinitely if high priority processes use a lot of CPU time. Starvation refers to a condition in which a procedure is never scheduled to run.

4 Round Robin

Round robin is one of the oldest, simplest, fairest, and most extensively used algorithms (RR). Round Robin Scheduling is beneficial in time-sharing scenarios where the system has to guarantee adequate response times for interactive users since it is pre-emptive (at the end of the time-slice).

5 Shortest Remaining time

Shortest remaining time, commonly known as shortest remaining time first (SRTF), is a scheduling algorithm that is a pre-emptive form of shortest job next scheduling. The procedure with the shortest length of time till completion is chosen to be executed in this scheduling technique.

6 Multilevel Queue Scheduling

The ready queue has been divided into seven different queues using the multilevel queue scheduling algorithm. These processes are permanently allocated to one queue based on their priority, such as memory size, process priority, or process kind. Each queue has its own method for scheduling. Some queues are utilized for the foreground process, while others are used for the background process.

(Ques10, 2017)

Inter-Process Communication

The operating system's technique for allowing processes to interact with one another is known as interprocess communication. This communication might include a process informing another process of an occurrence or data being transferred from one process to another.

The figure below shows the inter-process Communication:

Interprocess communication necessitates the use of synchronization. It's either handled by the communication processes or given via the interprocess control mechanism.

Some methods to provide synchronisation in Interprocess Communication are as follows:

- Semaphore
- Mutual Exclusion
- Barrier
- Spinlock

Implementation of Process

The operating system uses a table (an array of structures) called the process table to implement the process model, with one entry per process. (These items are also known as process control blocks by some writers.) This entry contains critical information about the process's state, such as the program counter, stack pointer, memory allocation, the status of open files, accounting and scheduling information, and anything else that must be saved when the process is switched from running to ready or blocked so that it can be restarted as if it had never been stopped.

Process Table and Process Control Block are used to implement Process Model and keep track of all process information. When a new process is created, the operating system allocates memory for it, loads process code into that memory, and creates data space for it.

Conclusion

To summarize, a process is a program that is being run. It is a crucial component of the operating system. The computer can accomplish activities efficiently and at a fast speed thanks to process management. The process control block assists in keeping track of the state of the process execution. Process scheduling aids in the removal of a running process from the CPU and the selection of a new or ready process to run according to the plan.

Bibliography

GeeksforGeeks. (2021, june 28). *Introduction Of Process Management*. Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/introduction-of-process-management/

Guru99. (2022, April 09). *Process Management in OS*. Retrieved from Guru99: https://www.guru99.com/process-management-pcb.html#:~:text=Process%20management%20involves%20various%20tasks,to%20share%20and%20exchange%20information.

IncludeHelp. (2020). *Process management of operating system*. Retrieved from IncludeHelp: https://www.includehelp.com/operating-systems/process-management-of-operating-system.aspx

Point, J. (2021). *Process management in OS*. Retrieved from JavaTpoint: https://www.javatpoint.com/process-management-in-os

Ques10. (2017). *What is scheduling? Give different scheduling policies and their comparision.* Retrieved from Ques10: https://www.ques10.com/p/4433/what-is-scheduling-give-different-scheduling-polic/?

TANENBAUM, A. S. (2015). Process Hierarchies. In A. S. TANENBAUM, *MODERN OPERATING SYSTEMS.* ANDREW S. TANENBAUM: PEARSON.

Tutorialspoint. (2021). *Operating System - Process Scheduling*. Retrieved from TutorialsPoint: https://www.tutorialspoint.com/operating_system/os_process_scheduling.htm

TutorialsPoint. (2021). *What is Process Control Block (PCB)?* Retrieved from TutorialsPoint: https://www.tutorialspoint.com/what-is-process-control-block-pcb

Appendix

Appendix - A(Glossary)

Appendix - B (Process Scheduling)

Appendix -C(Priority Scheduling)