

Bash Scripting

Report By: Sayush Khadka

Table of Contents

1. Task A.....	1
1.1. Introduction:.....	1
1.2. Script:	2
1.3. Testing:	8
1.3.1. Test 1: Run without username	8
1.3.2. Test 2: run with username and id	9
1.3.3. Test 3: run incorrect password 3 times	10
1.3.4. Test 4: run correct password	12
1.3.5. Test 5: band name	13
1.3.6. Test 6: incorrect band CODE	15
1.3.7. Test 7: correct band CODE	17
1.3.8. Test 8: pick 4 member names	18
1.3.9. Test 9: pick same member name	20
1.3.10. Test 10: wrong user id. (Parameter validation).....	22
1.3.11. Test 11: right user id	23
1.3.12. Test 12: NO External File of member (except 3 profile player that you have made) or Invalid Player ID.	24
1.3.13. Test 13: EXIT YES or Do not PLAY AGAIN.....	26
1.3.14. Test 14: EXIT NO or PLAY AGAIN.....	27
1.4. Content on 3 files:	28
1.4.1. John Lennon (JL):	28
1.4.2. Angus Young (AY):	28
1.4.3. Kurt Cobain (KC):	29
1.4.4. Content of the band Nirvana (NIR):	29
1.5. Conclusion:.....	30
2. Task B.....	31
2.1. Introduction:	31
2.2. Aims and Objectives (Process management with the help of CPU):	32
2.3. Background:	32
2.3.1. Process Architecture.....	32
2.3.2. Process control blocks:.....	33
2.3.3. Process States:.....	34
2.3.4. Process hierarchies:	35

2.3.5. Implementation of process:.....	37
2.4. Conclusion:.....	39
Bibliography	40
A. Appendix.....	42
A.1. Appendix- A (Glossary)	42
A.2. Appendix- B (Process Scheduling).....	43
A.3. Appendix- C (Priority scheduling)	44

Table of tables:

Table 1 Run without username	8
Table 2 Run with username and id.....	9
Table 3 Run incorrect password 3 times	11
Table 4 Run correct password	12
Table 5 Display band names.....	14
Table 6 Incorrect band code.....	15
Table 7 Correct band code.....	17
Table 8 Pick 4 members.....	18
Table 9 Pick same member.....	20
Table 10 Wrong user id (user id)	22
Table 11 Right user id	23
Table 12 NO External File of member (except 3 profile players that have been made) or Invalid Player ID.....	25
Table 13 EXIT YES or Do not PLAY AGAIN	26
Table 14 EXIT NO or PLAY AGAIN.	27

Table of figures:

Figure 1 Run without username	8
Figure 2 Run with username and id.....	9
Figure 3 Run incorrect password 3 times	11
Figure 4 Run correct password	13
Figure 5 Display band names.....	14
Figure 6 Incorrect band code	16
Figure 7 Correct band code.....	17
Figure 8 Pick 4 members	19
Figure 9 Pick same member	21
Figure 10 Wrong user id (user id).....	22
Figure 11 Wrong user id (username).....	22
Figure 12 Right user id	23
Figure 13 NO External File of member (except 3 profile player that you have made) or Invalid Player ID.	25
Figure 14 EXIT YES or Do not PLAY AGAIN	26
Figure 15 EXIT NO or PLAY AGAIN.	27

1. Task A

1.1. Introduction:

Bash is a shell that allows you to communicate with your computer and give commands. A script is simply a set of instructions for the computer to follow to carry out various tasks. A script allows you to automate specific procedures while also getting results faster than the traditional method. Normally, when you put a basic or complicated bash command into the terminal, it runs right away. With bash scripts, you may issue many instructions or commands at once, and the computer will only execute them all when you run the script. In a nutshell, you can run a single bash command in the terminal, but if you want to run many commands at once, you'll need to create a bash script (Malik, 2020).

WHY bash scripting?

If you want more control over your operating system and to do a variety of OS tasks, Bash is the way to go. When we mention bash, we're referring to both the bash programming language and the Linux operating system's utilities. Every tool on Linux has a specific purpose to fulfil, and they all do it differently. When you need to link all those tools together in such a way that they all work together to perform a task that would otherwise be difficult to do, Bash comes in useful. Anything linked to the Linux operating system, for example, maybe done in other programming languages such as Python or Perl, although many OS-related tasks are difficult to do. Bash is a simple, black-and-white command-line shell for the Linux operating system. Anyone who must use Linux OS tools (such as ls, cd, cat, touch, grep, and so on) should learn bash instead of any other programming language (Malik, 2020).

1.2. Script:

```
#!/bin/bash
#asking the user to enter username and id as a parameter(just kind of command
line argument.)
username=$1
id=$2

#playAgain function (asks user to play again or not)
playAgain()
{
    echo
    echo -e "Do you want to play again? Input (yes/y) to play again and (no/n)
to exit: \c"
    read againValue
    #checking empty string
    if [ -z $againValue ]
    then
        echo -e "Do not leave it empty."
        playAgain
    else
        #converting to upper case
        againValue=${againValue^^}
    fi

    if [ $againValue == "YES" ] || [ $againValue == "Y" ]
    then
        codeForBands
    elif [ $againValue == "NO" ] || [ $againValue == "N" ]
    then
        exit
    else
        echo -e "Choose only y/n or yes/no."
        playAgain
    fi
}

#select the members and displly the content of the file selected.
menuOfMembers()
{
    PS3="Choose your favourite member from these options: "
    select memberBand in $bandMemberOne $bandMemberTwo $bandMemberThree
    do
        case $memberBand in
            KC | AY | JL)
                #opens file of the option selected
            ;;
        esac
    done
}
```

```

        cat $memberBand
        break
    ;;

*)
    echo -e "Sorry. Wrong input."
    echo
    ;;
esac
done
playAgain
}

#function to display band members
codeForMembers()
{
    echo
    echo -e "-----"
    echo -e "Codes for members: "
    echo -e "-----"
    echo -e "John Lennon:    JL"
    echo -e "Angus Young:    AY"
    echo -e "Freddie Mercury: FM"
    echo -e "Debbie Harry:   DH"
    echo -e "Kurt Cobain:    KC"
    echo -e "-----"

    echo
    echo -e "Choose only 3 members from the given options: \c"
    read bandMemberOne bandMemberTwo bandMemberThree bandMemberFour

    choiceForMembers
}

#function to choose band members
choiceForMembers()
{
    #checking empty string
    if [ -z $bandMemberOne ] || [ -z $bandMemberTwo ] || [ -z $bandMemberThree ]
    then
        echo -e "Error!!!Do not leave it empty."
        codeForMembers

    #to restrict for 3 members
    elif [ $bandMemberFour ]
    then

```



```

        echo -e "Error!!!Can only select 3 members."
        codeForMembers

        #checking if all the input or same or not
        elif [ "$bandMemberOne" == "$bandMemberTwo" ] || [ "$bandMemberTwo" ==
"$bandMemberThree" ] || [ "$bandMemberThree" == "$bandMemberOne" ]
        then
            echo -e "Error!!!The members cannot be the same. The members must be
different at every input."
            codeForMembers

            #checking if the input is from the given options or not
            elif [ "$bandMemberOne" != "JL" ] && [ "$bandMemberOne" != "AY" ] && [
"$bandMemberOne" != "FM" ] && [ "$bandMemberOne" != "DH" ] && [
"$bandMemberOne" != "KC" ]
            then
                echo -e "Error!!!Select members from the given options."
                codeForMembers

                #checking if the input is from the given options or not
                elif [ "$bandMemberTwo" != "JL" ] && [ "$bandMemberTwo" != "AY" ] && [
"$bandMemberTwo" != "FM" ] && [ "$bandMemberTwo" != "DH" ] && [
"$bandMemberTwo" != "KC" ]
                then
                    echo -e "Error!!!Select members from the given options."
                    codeForMembers

                    #checking if the input is from the given options or not
                    elif [ "$bandMemberThree" != "JL" ] && [ "$bandMemberThree" != "AY" ] && [
"$bandMemberThree" != "FM" ] && [ "$bandMemberThree" != "DH" ] && [
"$bandMemberThree" != "KC" ]
                    then
                        echo -e "Error!!!Select members from the given options."
                        codeForMembers

                else
                    menuOfMembers

            fi
        }

#function to choose a band
choiceForBands()
{
    if [ -z $bandCode ]
    then
        echo -e "Error!!!Do not leave it empty.Please choose a band."
    fi
}

```

```

        codeForBands

        #band to be selected is nirvana and the code is NIR
        elif [ $bandCode == "NIR" ]
        then
            #opens file NIR to display the contents in it

            echo -e "
Nirvana was an American rock band formed in Aberdeen, Washington, in 1987.
Founded by lead singer and guitarist Kurt Cobain and bassist Krist Novoselic,
the band went through a succession of drummers, most notably Chad Channing,
before recruiting Dave Grohl in 1990.
Nirvana's success popularized alternative rock, and they were often referenced
as the figurehead band of Generation X.
Their music maintains a popular following and continues to influence modern
rock culture.
"

            codeForMembers

        else
            echo "Error!!!Incorrect input."
            codeForBands

        fi
    }

#function to display band codes
codeForBands()
{
    echo
    echo -e "-----"
    echo -e "Codes for bands: "
    echo -e "-----"
    echo -e "Beatles: BEA"
    echo -e "AC/DC:   AD"
    echo -e "Queen:   QUE"
    echo -e "Blondie: BLO"
    echo -e "Nirvana: NIR"
    echo -e "-----"
    echo

    echo -e "Choose the band from the given options: \c"
    read bandCode

    #calling choiceforBands function
    choiceForBands
}

```

```

# welcome fucntion to print username and id of the user and date after access
through secret key
welcome()
{
    echo
    echo -e "-----"
    echo -e "ID:          $id"
    echo -e "Username:       $username"
    echo -e "Date and Time:  "$(date)
    echo -e "-----"

    codeForBands
}

#validating username and id
#checking empty or not
if [ -z $username ] || [ -z $id ]
then
    echo -e "Error!!! Username or ID is missing."
#username = sayush, id = 1
elif [ $username != sayush ] || [ $id != 1 ]
then
    echo -e "Error!!!Wrong username or user id."
else

    attempt=0
    secretKey=0
    #secret key = 1234
    while [ $secretKey != 1234 ]
    do
        #runs the Loop 3 times
        if [ $attempt -le 3 ]
        then
            echo
            echo -e "Enter the secret key: \c"
            read -s secretKey #to hide the value of secretKey
            echo

            #checking if the secret key is empty
            if [ -z $secretKey ]
            then
                echo "Error!!!Do not leave the secret key empty."
                attempt=$(( $attempt+1 ))
                #secret key value set to 0 again
                secretKey=0
            fi
        fi
    done
}

```

```
        elif [ $secretKey == 1234 ]
        then
            break

        else
            echo -e "Error!!!The entered secret key does not match the
real secret key."
            attempt=$(( $attempt+1 ))
        fi

        else
            echo
            echo "The program is closed."
            exit
        fi

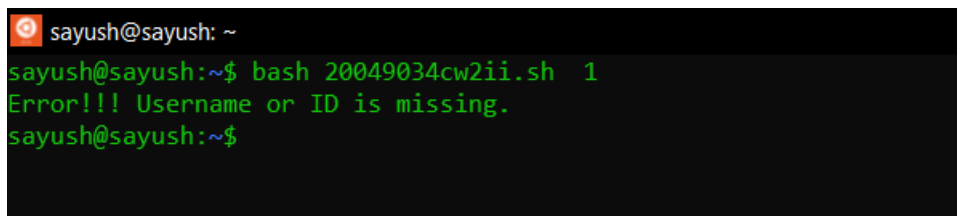
    done
    #calling welcome function
    welcome
fi
```

1.3. Testing:

1.3.1. Test 1: Run without username

Test No:	1
Objective:	To run without the username.
Action:	The program was run without the username.
Expected result:	The program would display error message. "Error!!! Username or ID is missing."
Actual result:	The program displayed error message. "Error!!! Username or ID is missing."
Conclusion:	Test successful.

Table 1 Run without username



```
sayush@sayush: ~  
sayush@sayush:~$ bash 20049034cw2ii.sh 1  
Error!!! Username or ID is missing.  
sayush@sayush:~$
```

Figure 1 Run without username

1.3.2. Test 2: run with username and id

Test No: 2	
Objective:	To run with username and id.
Action:	The program was run with username and id.
Expected result:	The program would run successfully and ask the user for the secret key.
Actual result:	The program run successfully and asked the user for the secret key.
Conclusion:	Test successful.

Table 2 Run with username and id

```
sayush@sayush: ~  
sayush@sayush:~$ bash 20049034cw2ii.sh 1  
Error!!! Username or ID is missing.  
sayush@sayush:~$ bash 20049034cw2ii.sh sayush 1  
Enter the secret key:
```

Figure 2 Run with username and id

1.3.3. Test 3: run incorrect password 3 times

Test No: 3	
Objective:	To run incorrect password 3 times.
Action:	Incorrect password was entered 3 times again after asking for the password initially.
Expected result:	<ul style="list-style-type: none">• The program would display error message. "Error!!!The entered secret key does not match the real secret key."• And the program would display a message. "The program is closed."• The program would close.
Actual result:	<ul style="list-style-type: none">• The program displayed error message. "Error!!!The entered secret key does not match the real secret key."• And the program displayed a message. "The program is closed."• The program closed.
Conclusion:	Test successful.

--	--

Table 3 Run incorrect password 3 times

```
sayush@sayush:~$ bash 20049034cw2ii.sh sayush 1
Enter the secret key:
Error!!!The entered secret key does not match the real secret key.

Enter the secret key:
Error!!!The entered secret key does not match the real secret key.

Enter the secret key:
Error!!!The entered secret key does not match the real secret key.

Enter the secret key:
Error!!!The entered secret key does not match the real secret key.

The program is closed.
sayush@sayush:~$
```

Figure 3 Run incorrect password 3 times

1.3.4. Test 4: run correct password

Test No: 4	
Objective:	To run correct password.
Action:	Correct password was entered.
Expected result:	<ul style="list-style-type: none">• The program would display the id, username, and date and time.• The program would display the codes for the bands.
Actual result:	<ul style="list-style-type: none">• The program displayed the id, username, and date and time.• The program displayed the codes for the bands.
Conclusion:	Test successful.

Table 4 Run correct password

```
sayush@sayush:~$ bash 20049034cw2ii.sh sayush 1

Enter the secret key:

-----
ID:          1
Username:    sayush
Date and Time: Sun Apr 24 02:02:13 +0545 2022
-----

Codes for bands:
-----
Beatles: BEA
AC/DC:   AD
Queen:   QUE
Blondie: BLO
Nirvana: NIR
-----

Choose the band from the given options:
```

Figure 4 Run correct password

1.3.5. Test 5: band name

Test No: 5	
Objective:	To display band names and their codes after entering correct password.
Action:	Correct password was entered.
Expected result:	The program would display band names and their codes.
Actual result:	The program displayed band names and their codes.

Conclusion:	Test successful.

Table 5 Display band names

```
sayush@sayush:~$ bash 20049034cw2ii.sh sayush 1

Enter the secret key:

-----
ID:          1
Username:    sayush
Date and Time: Sun Apr 24 02:02:13 +0545 2022
-----

-----
Codes for bands:
-----
Beatles: BEA
AC/DC:   AD
Queen:   QUE
Blondie: BLO
Nirvana: NIR
-----

Choose the band from the given options:
```

Figure 5 Display band names

1.3.6. Test 6: incorrect band CODE

Test No: 6	
Objective:	To enter incorrect band code.
Action:	Incorrect band code was entered.
Expected result:	<ul style="list-style-type: none">• The program would display an error message. "Error!!!!Incorrect input."• The program would ask again to enter the band code until the entered band code would be correct.
Actual result:	<ul style="list-style-type: none">• The program displayed an error message. "Error!!!!Incorrect input."• The program asked again to enter the band code until the entered band code was correct.
Conclusion:	Test successful.

Table 6 Incorrect band code

```
sayush@sayush:~$ bash 20049034cw2ii.sh sayush 1

Enter the secret key:

-----
ID:          1
Username:    sayush
Date and Time: Sun Apr 24 02:14:55 +0545 2022
-----

-----
Codes for bands:
-----
Beatles: BEA
AC/DC:   AD
Queen:   QUE
Blondie: BLO
Nirvana: NIR
-----

Choose the band from the given options: AD
Error!!!Incorrect input.

-----
Codes for bands:
-----
Beatles: BEA
AC/DC:   AD
Queen:   QUE
Blondie: BLO
Nirvana: NIR
-----

Choose the band from the given options:
```

Figure 6 Incorrect band code

1.3.7. Test 7: correct band CODE

Test No: 7	
Objective:	To enter correct band code.
Action:	Correct band code was entered (NIR).
Expected result:	<ul style="list-style-type: none"> The program would display information about the band. The program would display codes for members.
Actual result:	<ul style="list-style-type: none"> The program would display information about the band. The program would display codes for members.
Conclusion:	Test successful.

Table 7 Correct band code

```

-----
Codes for bands:
-----
Beatles: BEA
AC/DC:  AD
Queen:  QUE
Blondie: BLO
Nirvana: NIR
-----

Choose the band from the given options: NIR

Nirvana was an American rock band formed in Aberdeen, Washington, in 1987.
Founded by lead singer and guitarist Kurt Cobain and bassist Krist Novoselic, the band went through a succession of drummers, most notably Chad Channing, before recruiting Dave Grohl in 1990.
Nirvana's success popularized alternative rock, and they were often referenced as the figurehead band of Generation X.
Their music maintains a popular following and continues to influence modern rock culture.

-----
Codes for members:
-----
John Lennon:  JL
Bogus Young:  BY
Freddie Mercury: FM
Bobbie Harry:  BH
Kurt Cobain:  KC
-----

Choose members from the given options:

```

Figure 7 Correct band code

1.3.8. Test 8: pick 4 member names

Test No: 8	
Objective:	To pick 4 members.
Action:	4 members were picked.
Expected result:	<ul style="list-style-type: none">• The program would display an error message. "Error!!!Can only select 3 members."• The program would ask again to pick 3 members until the user would enter 3 members only from the options.
Actual result:	<ul style="list-style-type: none">• The program would display an error message. Error!!!Can only select 3 members.• The program would ask again to pick 3 members until the user would enter 3 members only from the options.
Conclusion:	Test successful.

Table 8 Pick 4 members

```
-----  
Codes for members:  
-----  
John Lennon:    JL  
Angus Young:    AY  
Freddie Mercury: FM  
Debbie Harry:   DH  
Kurt Cobain:    KC  
-----  
  
Choose only 3 members from the given options: JL FM DH KC  
Error!!!Can only select 3 members.  
  
-----  
Codes for members:  
-----  
John Lennon:    JL  
Angus Young:    AY  
Freddie Mercury: FM  
Debbie Harry:   DH  
Kurt Cobain:    KC  
-----  
  
Choose only 3 members from the given options:
```

Figure 8 Pick 4 members

1.3.9. Test 9: pick same member name

Test No: 9	
Objective:	To pick same member name.
Action:	Same member code was entered.
Expected result:	<ul style="list-style-type: none">• The program would display an error message. “Error!!!The members cannot be the same. The members must be different at every input.”• The program would ask again to pick 3 members until the user would enter 3 different members only from the options.
Actual result:	<ul style="list-style-type: none">• The program would display an error message. “Error!!!The members cannot be the same. The members must be different at every input.”• The program would ask again to pick 3 members until the user would enter 3 different members only from the options.
Conclusion:	Test successful.

Table 9 Pick same member

```
-----  
Codes for members:  
-----  
John Lennon:    JL  
Angus Young:    AY  
Freddie Mercury: FM  
Debbie Harry:   DH  
Kurt Cobain:    KC  
-----  
  
Choose only 3 members from the given options: KC KC KC  
Error!!!The members cannot be the same. The members must be different at every input.  
  
-----  
Codes for members:  
-----  
John Lennon:    JL  
Angus Young:    AY  
Freddie Mercury: FM  
Debbie Harry:   DH  
Kurt Cobain:    KC  
-----  
  
Choose only 3 members from the given options:
```

Figure 9 Pick same member

1.3.10. Test 10: wrong user id. (Parameter validation)

Test No:	10
Objective:	To enter wrong user id.
Action:	Wrong user id was entered.
Expected result:	The program would display an error message. "Error!!!Wrong username or user id."
Actual result:	The program displayed an error message. "Error!!!Wrong username or user id."
Conclusion:	Test successful.

Table 10 Wrong user id (user id)

```
sayush@sayush:~$ bash 20049034cw2ii.sh sayush 101
Error!!!Wrong username or user id.
sayush@sayush:~$
```

Figure 10 Wrong user id (user id)

```
sayush@sayush:~$ bash 20049034cw2ii.sh khadka 1
Error!!!Wrong username or user id.
sayush@sayush:~$
```

Figure 11 Wrong user id (username)

1.3.11. Test 11: right user id

Test No: 11	
Objective:	To enter right user id.
Action:	Right user id was entered.
Expected result:	The program would run and would ask for the secret key.
Actual result:	The program runs and asks for the secret key.
Conclusion:	Test successful.

Table 11 Right user id

```
sayush@sayush:~$ bash 20049034cw2ii.sh sayush 1
Enter the secret key:
```

Figure 12 Right user id

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

Test No: 12	
Objective:	NO External File of member (except 3 profile players that have been made) or Invalid Player ID.
Action:	<p>Note: Members whose external files are there are JL, AY and KC.</p> <ul style="list-style-type: none">• a is entered as invalid input.• 50 is entered as invalid input.• 7 is entered as invalid input.• Option (1) was entered as FM.• Option (2) was entered as DH.• Option (3) was entered as valid input as KC.
Expected result:	<ul style="list-style-type: none">• The program would display an error message.• The program would again ask to choose the option until the correct member among the 3 is entered.• The program would display information of the valid member entered.
Actual result:	<ul style="list-style-type: none">• The program displayed an error message.• The program again asks to choose the option until the correct member among the 3 is entered.

	<ul style="list-style-type: none"> The program displayed the information of the valid member entered.
Conclusion:	Test successful.

Table 12 NO External File of member (except 3 profile players that have been made) or Invalid Player ID.

```

-----
Codes for members:
-----
John Lennon:    JL
Angus Young:    AY
Freddie Mercury: FM
Debbie Harry:   DH
Kurt Cobain:    KC
-----

Choose only 3 members from the given options: FM DH KC
1) FM
2) DH
3) KC
Choose your favourite member from these options: a
Sorry. Wrong input.

Choose your favourite member from these options: 50
Sorry. Wrong input.

Choose your favourite member from these options: 7
Sorry. Wrong input.

Choose your favourite member from these options: 1
Sorry. Wrong input.

Choose your favourite member from these options: 2
Sorry. Wrong input.

Choose your favourite member from these options: 3

Kurt Donald Cobain (February 20, 1967 – c. April 5, 1994) was an American singer, songwriter and artist.
He was the guitarist, lead vocalist and primary songwriter of the rock band Nirvana.
Through his angst-fueled songwriting and anti-establishment persona, Cobain's compositions widened the thematic conventions of mainstream rock.
He was heralded as a spokesman of Generation X and is considered one of the most influential musicians in the history of alternative rock.

Do you want to play again? Input (yes/y) to play again and (no/n) to exit:

```

Figure 13 NO External File of member (except 3 profile player that you have made) or Invalid Player ID.

1.3.13. Test 13: EXIT YES or Do not PLAY AGAIN

Test No: 13	
Objective:	To exit the program or not to play again.
Action:	n is entered.
Expected result:	The program would exit.
Actual result:	The program exits.
Conclusion:	Test successful.

Table 13 EXIT YES or Do not PLAY AGAIN

```
Choose your favourite member from these options: 3
Kurt Donald Cobain (February 20, 1967 – c. April 5, 1994) was an American singer, songwriter and artist.
He was the guitarist, lead vocalist and primary songwriter of the rock band Nirvana.
Through his angst-fueled songwriting and anti-establishment persona, Cobain's compositions widened the thematic conventions of mainstream rock.
He was heralded as a spokesman of Generation X and is considered one of the most influential musicians in the history of alternative rock.

Do you want to play again? Input (yes/y) to play again and (no/n) to exit: n
sayush@sayush:~$
```

Figure 14 EXIT YES or Do not PLAY AGAIN

1.3.14. Test 14: EXIT NO or PLAY AGAIN

Test No: 14	
Objective:	EXIT NO or PLAY AGAIN.
Action:	y was entered.
Expected result:	The program would start again from choosing the code of the bands.
Actual result:	The program started again from choosing the code of the bands.
Conclusion:	Test successful.

Table 14 EXIT NO or PLAY AGAIN.

```

Choose only 3 members from the given options: JL AV FM
1) JL
2) AV
3) FM
Choose your favourite member from these options: 1

John Winston Ono Lennon(born John Winston Lennon; 9 October 1940 – 8 December 1980).
Lennon was an English singer, songwriter, musician and peace activist who achieved worldwide fame as the founder, co-songwriter, co-lead vocalist and rhythm guitarist of the Beatles.
Lennon was characterised by the rebellious nature and acerbic wit in his music, writing and drawings, on film, and in interviews.
His songwriting partnership with Paul McCartney remains the most successful in history.

Do you want to play again? Input (yes/y) to play again and (no/n) to exit: y

-----
Codes for bands:
-----
Beatles: BEA
AC/DC:  AD
Queen:  QUE
Blondie: BLO
Nirvana: NIR
-----

Choose the band from the given options:

```

Figure 15 EXIT NO or PLAY AGAIN.

1.4. Content on 3 files:

1.4.1. John Lennon (JL):

John Winston Ono Lennon (born John Winston Lennon; 9 October 1940 – 8 December 1980).

Lennon was an English singer, songwriter, musician and peace activist who achieved worldwide fame as the founder, co-songwriter, co-lead vocalist and rhythm guitarist of the Beatles.

Lennon was characterised by the rebellious nature and acerbic wit in his music, writing and drawings, on film, and in interviews.

His songwriting partnership with Paul McCartney remains the most successful in history.

1.4.2. Angus Young (AY):

Angus McKinnon Young (born 31 March 1955) is an Australian musician, best known as the co-founder, lead guitarist, songwriter, and sole constant original member of the Australian hard rock band AC/DC.

He is known for his energetic performances, schoolboy-uniform stage outfits, and his own version of Chuck Berry's duckwalk.

Young was ranked 24th in Rolling Stone magazine's 100 greatest guitarists of all-time list In 2003,

Young and the other members of AC/DC were inducted into the Rock and Roll Hall of Fame.

1.4.3. Kurt Cobain (KC):

Kurt Donald Cobain (February 20, 1967 – c. April 5, 1994) was an American singer, songwriter and artist.

He was the guitarist, lead vocalist and primary songwriter of the rock band Nirvana.

Through his angst-fueled songwriting and anti-establishment persona, Cobain's compositions widened the thematic conventions of mainstream rock.

He was heralded as a spokesman of Generation X and is considered one of the most influential musicians in the history of alternative rock.

1.4.4. Content of the band Nirvana (NIR):

Nirvana was an American rock band formed in Aberdeen, Washington, in 1987.

Founded by lead singer and guitarist Kurt Cobain and bassist Krist Novoselic, the band went through a succession of drummers, most notably Chad Channing, before recruiting Dave Grohl in 1990.

Nirvana's success popularized alternative rock, and they were often referenced as the figurehead band of Generation X.

Their music maintains a popular following and continues to influence modern rock culture.

1.5. Conclusion:

The above task or the coursework is related to bash scripting. BASH is a powerful command and script execution environment. To automate regular duties and commands, we may utilize a BASH script. Many commands make up a BASH script. A BASH file's end is indicated by the `.sh` suffix.

The coursework was designed and commanded to create a user-friendly environment between the user and the program. The program was structured and implemented where the user had to run the program by entering the correct username and user id. After the user enters the correct username and user id the program asks the user to enter a secret key. When the user enters the correct secret key, the program runs further. In terms of a user entering any wrong input that isn't desired by the program, the program displays a meaningful error message and takes the action accordingly as per the code implementation of the program. After successful user login, the program displays the user id, username of the user, and date. Then it, similarly, asks the user to select a band from 5 different options (NIR being the correct option). When the user enters the correct option then the program displays information about the band and similarly displays error messages when the user gives wrong inputs. The program further runs after correct input asking the user to choose only 3 members from 5 options. After the user provides the input, the program asks the user to choose their favourite member amongst the 3 members entered by the user. When the correct input (JL AY KC) is given (single input), the program displays the information of the member selected from their respective external files. At the very end, the program asks the user to run the program or not. And if the user selects (y/yes), the program runs by asking the user to select the band and if the user selects (n/no), the program terminates.

The program has been implemented with loops, functions, conditional statements, validating user input, and nested conditional statements. The overall program and the task were informative and assistive in the step of learning to code and interact with the UNIX environment.

2. Task B

2.1. Introduction:

The operating system is a crucial software component that must work correctly and regularly. It's exceedingly tough to utilize and control hardware. It will have to be in binary machine code. Programming at this level is a difficult endeavour that needs much experience. As a result, operating systems are designed to operate in collaboration with hardware to solve problems and simplify users' lives (Khan, 2008).

Process management is crucial to the functionality of an operating system because of its significant relationship with the processor. The process management equation includes state switching, resource management, and scheduling (CPU scheduling, I/O scheduling, and so on). Process management displays the life cycle of a process using process state diagrams (Khan, 2008).

Process Management is key to the performance of an operating system because of its strong link with the processor. Although the most generally quoted description is "a program in execution" [1, 2, 3, 4, 5], there is no universally accepted definition of a process. To build a process, you'll need memory address space, a process identity, and certain system resources for code and data components. A process passes through various phases during its life cycle. For a better understanding of how processes work and operate, the process state model is useful. Operating systems have progressed throughout time. State models have been presented in the same manner that operating systems have historically been related to the computer architecture on which they operate (Khan, 2008).

2.2. Aims and Objectives (Process management with the help of CPU):

- i) For the CPUs to schedule processes and threads (javaTpoint, n.d.).
- ii) For the process to be paused and resumed (javaTpoint, n.d.).
- iii) To provide mechanisms for synchronization of the process (javaTpoint, n.d.).
- iv) Process scheduling's main aims are to keep the CPU busy and to guarantee that all processes respond quickly enough (Das, n.d.).
- v) To provide communication means for processes (javaTpoint, n.d.).
- vi) For the user and system processes both to be created and deleted (javaTpoint, n.d.).

2.3. Background:

2.3.1. Process Architecture

Process architecture is the hierarchical design of processes and systems that convert inputs into outputs (Tallyfy, n.d.).

The process architecture is divided into 4 sections:

- i) **Stack**
The process stack stores temporary data such as method/function parameters, return address, and local variables (tutorialspoint, n.d.).
- ii) **Heap**
Heap allocates memory that can be used by the application during its execution (Williams, 2022).
- iii) **Data**
This section covers both global and static variables (tutorialspoint, n.d.).
- iv) **Text**

The value of the program counter in the text section reflects the current activity (Williams, 2022).

2.3.2. Process control blocks:

The process control block in the operating system represents a process. A task control block is another name for a process control block. It's a database of data on a certain operation (Vaishnav, 2022).

A Process Control Block (PCB) is a data structure that stores information about a process (Vaishnav, 2022).

Structure or components of process blocks:

- i) **Process ID:** When a user establishes a new process, the operating system gives it a process ID, which is a unique identifier. This ID is used to differentiate the process from the rest of the system's processes. The number of processes that the operating system can handle at any given time is limited; for example, OS can only handle N processes at a time (Vaishnav, 2022).
- ii) **Process State:** A process goes through various stages from creation to conclusion. A process can be in one of five states while it is being executed which are (Vaishnav, 2022):
 - a) New
 - b) Ready
 - c) Running
 - d) Block or wait
 - e) Termination
- iii) **Program Counter:** The program counter is a pointer to the next instruction in the program to be executed. This PCB property stores the address of the next instruction to be performed in the process (Vaishnav, 2022).
- iv) **CPU registers:** CPU registers contains accumulators, index, and general-purpose registers, as well as condition code information (Williams, 2022). A

CPU register is a small space supplied to the CPU for rapid access. These registers are stored in virtual memory (RAM) (Vaishnav, 2022).

- v) CPU scheduling information: CPU scheduling information contains a process priority, pointers for scheduling queues, and other scheduling settings (Williams, 2022).
- vi) Accounting and business information: This property includes information on the resources used by the process over its lifetime. CPU time, connection time, and so forth are examples (Vaishnav, 2022).
- vii) Memory- management information: This data includes the information of the base and limit registers, as well as the page and segment tables. The memory system used for the operating system does influence this (Williams, 2022).
- viii) Process I/O information: This field comprises a list of all the input/output devices required by the process throughout its execution (Vaishnav, 2022).

2.3.3. Process States:

The state of a process at a certain moment in time is characterized as a process state. The state of a process determines its status. This enables us to collect additional data about our process at a certain point in time. There are seven major stages of the process in an operating system (Shukla, 2017). The process includes the following seven states:

- i) New: When a program is summoned from secondary memory or the hard disk to primary memory or RAM, a new process is started. In a nutshell, it determines when a process begins (Shukla, 2017).
- ii) Ready: During this time frame, the state of the process is loaded into the main memory and prepared for execution (Shukla, 2017).
- iii) Waiting: In this condition, the process is put on pause, enabling other processes to start running. In other terms, it refers to the amount of time a

process spends waiting for CPU time and other resources to be assigned to it for execution (Shukla, 2017).

- iv) Executing: This is a process's principal state, and it is in this state that the process is active. In other words, it is the length of time that a process is carried out by the CPU (Shukla, 2017).
- v) Blocked: It denotes the amount of time a process spends waiting for something to happen, such as input/output activities (Shukla, 2017).
- vi) Suspended: It refers to the state of a process when it is ready to execute but has not yet been added to the operating system's ready queue (Shukla, 2017).
- vii) Terminated: It denotes the point at which a process is terminated or completed, and all the process's resources and memory are released (Shukla, 2017).

2.3.4. Process hierarchies:

In a computer system, we need to run several processes at once, and some of them need to spawn more processes while they're running. When a process creates a child process, the parent and child processes tend to associate in specific ways. The child process can also produce more processes if needed. The name given to this parent-child structure of processes is Process Hierarchy (Pal, 2020).

In UNIX, a process group consists of a process and all its descendants and grandchildren. When a user sends a signal from the keyboard, it is forwarded to all members of the process group to which the keyboard is currently connected. Each process can intercept the signal and do anything it wants with it (Pal, 2020).

A hierarchical structure does not exist in Windows. Each process receives equal attention. It is only distantly related to the hierarchy in that it assigns a handle to a

parent process. This parent process, on the other hand, may pass the handle to another process, rendering the hierarchical system worthless (Pal, 2020).

To create a new process there are different ways. They are:

- i) Execution: The parent process either performs the child process in parallel or waits until all the children have been terminated before continuing (Priya, 2021).
- ii) Sharing: The parent and kid processes may share all resources, such as memory and files, or the children process may share only a subset of the parent's resources, or the parent and children processes may not share any resources at all (Priya, 2021).

2.3.5. Implementation of process:

To implement the Process Model and keep track of all process data, the Process Table and Process Control Block are utilized. The operating system allots memory to a new process, loads a process code into it, and creates data space for it when it is formed (TutorialsSpace.com, n.d.).

The process's status in its PCB is retained as 'new,' and when it moves to the ready state, the process's state in the PCB changes as well. When a running process must wait for input/output devices, its status is modified to 'blocked.' The various queues used for this were implemented as linked lists (TutorialsSpace.com, n.d.).

The following queues are listed below:

- i) Read queue: This queue is where ready-to-run processes are stored (TutorialsSpace.com, n.d.).
- ii) Blocked queue: It's used to keep track of processes that need the use of an input/output device or resource (TutorialsSpace.com, n.d.).
- iii) Suspended queue: It's used to maintain track of any processes that have been blocked or halted (TutorialsSpace.com, n.d.).
- iv) Free process queue: It's used to keep track of where there's free RAM in the memory that may be utilized to make a new PCB (TutorialsSpace.com, n.d.).

The next PCB is indicated with a pointer on each PCB. There is a header for each queue type. The header contains information about the first and last PCBs in the queue (TutorialsSpace.com, n.d.).

There is another header that gives information on the operating process; however, because the system only has one process, there is no queue of ongoing processes (TutorialsSpace.com, n.d.).

In addition to changing the status of a process, PCB saves its location for future continuation where it left off. The process program counter, program status word, registers, and other data are first stored on the stack. The stack information is then recorded in the relevant PCB for the process. The technique of preserving the status

of a running process in its PCB is known as context saving. The relevant event handling procedure is invoked when the context of a process has been stored. If an input-output device is in the wait state, for instance, it is sent to the blocked processes queue. Because the current process has been interrupted, a new one must be sent to the CPU. Therefore, a Scheduler is used to schedule a job from the ready queue. When a process is chosen from the ready queue, its PCB is loaded and delivered to the processor for execution (TutorialsSpace.com, n.d.).

By storing their context on PCBs and enabling state adjustment, the processes are implemented in this way (TutorialsSpace.com, n.d.).

2.4. Conclusion:

Task B of coursework 2 included information on process management. Process management encompasses a variety of tasks, including process development, process scheduling, deadlock management, and process termination. The operating system's job is to manage the system's ongoing processes. The operating system oversees managing processes and performing tasks such as resource allocation and process scheduling. The RAM and CPU of a computer are utilized while a process executes. The operating system is also in charge of keeping all the computer's functions in sync.

Overall, with the help of the operating system, process management plays a vital role in the proper functioning of the computer and its tasks.

Bibliography

Das, V., n.d. *enjoyalgorithms*. [Online]

Available at: <https://www.enjoyalgorithms.com/blog/process-management-in-operating-system>
[Accessed 23 April 2022].

javaTpoint, n.d. *javaTpoint*. [Online]

Available at: <https://www.javatpoint.com/process-management-in-os>
[Accessed 23 April 2022].

Khan, Y. I., 2008. *Formal Modeling of Process Management System using Automata, UML and VDM-SL*. Bannu, s.n.

Malik, T. S., 2020. *linuxhint*. [Online]

Available at: https://linuxhint.com/what_is_bash_script/
[Accessed 23 April 2022].

Pal, T., 2020. *technobyte*. [Online]

Available at: <https://technobyte.org/processes-in-operating-systems/#:~:text=Process%20Hierarchy&text=When%20a%20process%20creates%20another,a%20hierarchy%2C%20called%20Process%20Hierarchy.>
[Accessed 24 April 2022].

Priya, B., 2021. *tutorialspoint*. [Online]

Available at: <https://www.tutorialspoint.com/what-is-a-process-hierarchy>
[Accessed 24 April 2022].

Shukla, A., 2017. *includehelp.com*. [Online]

Available at: <https://www.includehelp.com/operating-systems/process-management-of-operating-system.aspx>
[Accessed 24 April 2022].

Tallyfy, n.d. *Tallyfy*. [Online]

Available at: <https://tallyfy.com/process-architecture/>
[Accessed 23 April 2022].

tutorialspoint, n.d. *tutorialspoint*. [Online]

Available at: https://www.tutorialspoint.com/operating_system/os_processes.htm
[Accessed 23 April 2022].

TutorialsSpace.com, n.d. *TutorialsSpace.com*. [Online]

Available at: <http://www.tutorialsspace.com/Operating-System/15-Processes-Implementation-Of-Processes.aspx#:~:text=Process%20Model%20is%20implemented%20by,setup%20data%20space%20for%20it%20.>
[Accessed 24 April 2022].

Vaishnav, N., 2022. *Scaler Topics*. [Online]

Available at: <https://www.scaler.com/topics/operating-system/process-control-block-in-os/>
[Accessed 23 April 2022].

Williams, L., 2022. *Guru99*. [Online]
Available at: <https://www.guru99.com/process-management-pcb.html>
[Accessed 23 April 2022].

A. Appendix

A.1. Appendix- A (Glossary)

- a) bash - is a shell that allows you to communicate with the computer and give commands
- b) script - A script is simply a set of instructions for the computer to follow to carry out various tasks
- c) endeavour - try hard to do or achieve something.
- d) threads - a thread is a single sequential flow of activities being executed in a process; it is also known as a thread of execution or a thread of control.
- e) PCB – Process control blocks

A.2. Appendix- B (Process Scheduling)

Process scheduling is an important aspect or element of process management that affects the efficiency of the processor. As a result, excellent queue management and delay reduction are required, allowing the processor to make the most of its time.

Process scheduling may be divided into four categories:

- i) High level or long-term scheduling
- ii) Medium-term scheduling
- iii) Low level or short-term scheduling
- iv) I/O scheduling

The medium-term scheduler oversees switching, whereas the long-term scheduler oversees allocating programs for execution. To move a process from main memory to virtual memory, the swapping function is frequently employed. When the main memory needs to make room for additional processes, this occurs. The dispatcher function of the short-term scheduler is to choose the next process to run.

A.3. Appendix- C (Priority scheduling)

Priority Scheduling is a technique for scheduling processes that are prioritized. This approach is used by the scheduler to determine which jobs should be prioritized. Works with greater priority should be finished first, followed by jobs with equal priority in a round-robin or FCFS pattern. Memory restrictions, time constraints, and other considerations all play a role in prioritization. There are 2 types of priority scheduling. They are:

i) Pre-emptive scheduling:

Pre-emptive scheduling assigns tasks based on their priority. Even though the lower priority activity is continuing, it is occasionally important to complete a higher priority duty first. The lower priority job is placed on wait for a period, then resumed after the higher priority activity is finished.

ii) Non-pre-emptive scheduling:

In this scheduling system, the CPU has been assigned to a certain process. To free up the CPU, the process that is keeping it busy will switch context or terminate. It's the only solution that works on a wide range of hardware. This is because, unlike pre-emptive scheduling, it does not need any special hardware (such as a timer).