Module Code & Module Title

CS5001NA Networks and Operating System

Assessment Weightage & Type

20% Individual Coursework

Year and Semester

2019-20 Autumn

Student Name: Anshul Agarwal

London Met ID: 19031269

College ID: 190006

Assignment Due Date:11th April 2021

Assignment Submission Date: 11th April 2021

Word Count (TASK B):2000

Acknowledgement

I would like to express my gratitude to London Metropolitan University and Islington College for giving me an opportunity for providing us this course in our syllabus. I would also like to thank Mr. Puranjan Acharya our tutor teacher and Mr. Dpeshor Silwal out module leader teacher for providing us this project report which has enabled us to widen our scope of knowledge.

I would also like to express my gratitude to all my colleagues and people who have helped me by providing their valuable assistance and time for completion of project work.

Table of Contents

Task A

Introduction

A shell script is a programming program that runs in the Unix/Linux shell and can be either of the following:

- The Bourne Shell is a fictional character created by John Bourne.
- The Shell in C
- The Shell of Korn
- The Bourne-Again Shell (GNU Bourne)

A shell is a command-line interpreter, and shell scripts commonly execute file editing, programme execution, and text printing. Shell scripts must have multiple mandatory structures that inform the shell environment what to do and when. Most scripts, though, are more complicated than the one above.After all, the shell is a full-fledged programming language with variables, control structures, and so on. A script is always just a series of commands executed in order, no matter how complex it becomes. (Tutorialspoint).A shell is a special user application that allows users to interact with operating system resources. Shell accepts human-readable commands from the user and converts them into kernel-friendly commands. It's a command language interpreter that can read instructions from keyboards or files and execute them. When a user signs in or launches a terminal, the shell is launched. (greeks, 2017).

Bash (also known as Bourne Again Shell) is an interpreter for shell commands.it was written by Brian Fox and was released in 1889. A shell interpreter takes plain text commands and uses them to invoke Operating System utilities. The is command, for example, displays a directory's files and directories. Bash is a better clone of Sh (Bourne Shell). Shell scripting is the process of writing a programme that the shell will run, and a shell script is the file or programme that the shell will run.

You may have used commands like mv to transfer or rename a file, touch to make a file, or nano to modify a file if you're a programmer. These commands are entered into a console, which is the shell interpreter's user interface. In and by itself, a shell script is a full-fledged programming language. It allows one to describe variables, functions, and conditional execution of shell commands. Having a terminal at your disposal will save you time and your operating system's GUI cannot always have the tools you need to execute tasks like executing a binary file with options. Working in a terminal, on the other hand, makes you look like a nerd, if that's your thing. (Hiwarale, 2019).

This coursework was given to us to develop the script with the help of the UNIX environment and make it run. Here we have developed the program by asking the username and id. We made the script in such a way that the user cannot keep alphabets in its id and if he tries to enter the alphabets the program will tell him to enter the integer value not the alphabet. We have kept the secret password so that the authorized person can only get the access or no other individual can access it. There is only four tries to keep the right password if the user tries more than four times the program will get exit. We have made the program of the favourite team and player where we have given their unique code so that it does not match with others the user have to keep only the code if the user tries to keep the favourite country then it will tell again to keep the country code. After the country is selected then we have to choose the favourite three players from best five players and type their code and the list will appear and from that three player whose information u want to read we have to choose serial number and last it will ask do you want to continue the program if we write yes the code run and again or if we write no the program will end and exit.

Script

```
#!/bin/bash

if [ $# = 2 ]  #checking number of parameters

then

if [[ $2 != [0-9]* ]]

then

echo

echo "Intiger value required in 2nd parameter"

exit

else

echo
```

```
fi

else

echo "Enter two Parameter username and id"

exit

fi

secret() {

echo -e "Enter Secret key:\c"

read Secret

if [ "$Secret" != "pass" ]

then

echo "Enter correct KEy"

count=$(( $count + 1 ))
if [ $count -gt 3 ]

then

echo "exit"

exit

else

secret

fi

else

echo

fi

}
```

```
secret

echo "Welcome prac1"

echo -e "FirstName:$1"

echo -e "Lomdon ID:$2"

date

countries() {

echo -e "S.N Country Name \tcode"

echo -e "1.  Brazil \t\tBRZ"

echo -e "2.  Argentina \t\tARG"

echo -e "3.  Nepal \t\tNEP"

echo -e "4.  China \t\tCHI"

echo -e "5.  England \t\tENG"
echo -e "Enter Country Code:\c"

read Country

if [[ $Country != NEP ]]

then

echo "this is not fav country"

countries

else

echo "this is fav contry"

fi

}

countries
```

```
choose() {

echo -e "S.N Player Name \tcode"

echo -e "1.  Lional Messi \tLM"

echo -e "2.  Neymar Junior \tNJ"

echo -e "3.  Kiran Chemjong \tKC"

echo -e "4.  Zheng Zhi \t\tZZ"

echo -e "5.  Harry Kane \t\tHK"

echo -e "Enter three player Code:\c"

read player1 player2 player3

best=($player1 $player2 $player3)

        if [[ ${#best[@]} -ne 3 ]]

                then
                echo "Enter 3 codes"

                choose

        else

                if [ $player1 = "LM" ] || [ $player1 = "NJ" ] || [ $player1 = "KC" ] || [ $player1 = "ZZ" ] || [ $player1 = "HK" ] &&

                [ $player2 = "LM" ] || [ $player2 = "NJ" ] || [ $player2 = "KC" ] || [ $player2 = "ZZ" ] || [ $player2 = "HK" ] &&

                [ $player3 = "LM" ] || [ $player3 = "NJ" ] || [ $player3 = "KC" ] || [ $player3 = "ZZ" ] || [ $player3 = "HK" ]

                    then

                    if [ $player1 = $player2 ] && [ $player2 = $player3 ] && [ $player1 = $player3 ]

                        then

                        echo "Enter Different code"
```

```
                    choose
                else
echo "list"
                fi
            else
                echo "player code is incorrect"
                choose
            fi
        fi
}
choose
```

```
selects() {

    PS3="select one player from list:"

    play="$player1 $player2 $player3"

    select i in $play

    do

    if [[ -n "$i" ]]

        then

        case $i in

        "LM")

        cat LM

        break

        ;;
        "NJ")

        cat NJ

        break

;;

        "KC")

        cat KC

        break

        ;;

        "ZZ")

        echo -e "mising!!"

        countries
        break

        ;;

        "HK")

        echo -e "mising"

        countries

        break

        ;;

        esac

    else

        echo "Enter a correct number"

        echo "wrong choice, try again "

    fi
    done

}

selects

restart() {

echo -e "DO U WANT TO REEXECUTE\c"

read dodey

if [[ $dodey = yes ]]

then

countries

choose

selects
```

```
restart

elif [[ $dodey = no ]]

then

echo "Exit"

exit

else

echo "Enter yes or no"

restart

fi

}

restart
```

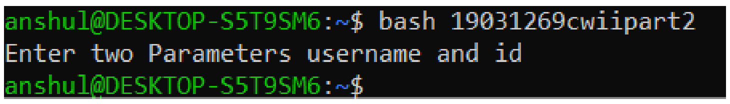
3 .Testing

3.1 Test 1

Test No.	1
Input	bash 19031269cwiipart2
Expected Output	Program should run
Actual Output	Program runned

Table 1: run without username and id

Screenshots



```
anshul@DESKTOP-S5T9SM6:~$ bash 19031269cwiipart2
Enter two Parameters username and id
anshul@DESKTOP-S5T9SM6:~$
```

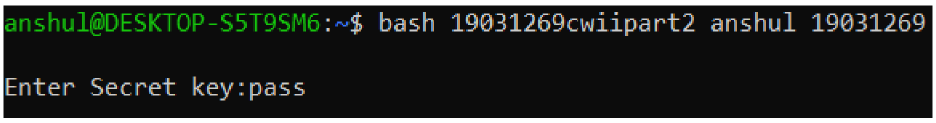
Figure 1: run without username and id

3.2 Test 2

Test No.	2
Input	bash 19031269cwiipart2 anshul 19031269
Expected Output	Program asked for username and id
Actual Output	Program started and asked for username and id

Table 2: Run with correct username and id

Screenshots:



```
anshul@DESKTOP-S5T9SM6:~$ bash 19031269cwiipart2 anshul 19031269

Enter Secret key:pass
```

Figure 2: Run with correct username and id

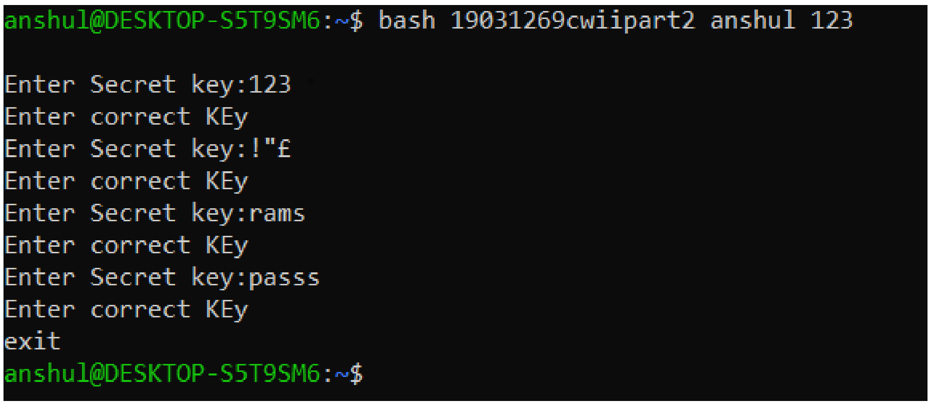
Here we have asked to run the program with the username and id and we have stated the program.

3.3 Test 3

Test No	3
Input	Enter the incorrect password for 4 times
Expected Output	Program should be exit after the 4 times incorrect password
Actual Output	Program got terminated.

Table 3: run with incorrect password 4 times

Screenshots:



```
anshul@DESKTOP-S5T9SM6:~$ bash 19031269cwiipart2 anshul 123

Enter Secret key:123
Enter correct KEy
Enter Secret key:!"£
Enter correct KEy
Enter Secret key:rams
Enter correct KEy
Enter Secret key:passs
Enter correct KEy
exit
anshul@DESKTOP-S5T9SM6:~$
```

Figure 3: run with incorrect password 4 times

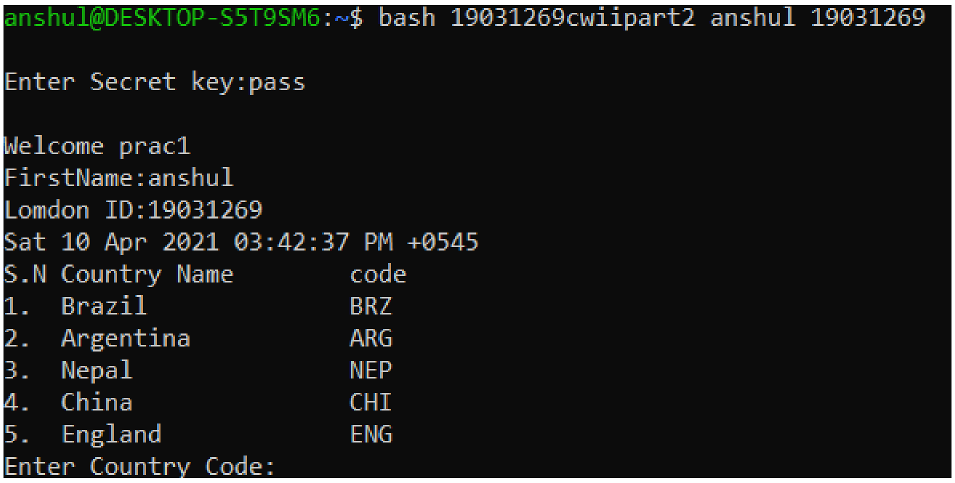
Here we have asked to run the program to enter the four time the wrong secret key and to know what will happen if the user enter the wrong password for four times.

3.4 Test 4

Test No	4
Input	Enter the password pass
Expected Output	Program should run and asked for favourite football team.
Actual Output	Program got successfully.

Table 4: run with the correct password

Screenshots:



```
anshul@DESKTOP-S5T9SM6:~$ bash 19031269cwiipart2 anshul 19031269

Enter Secret key:pass

Welcome prac1
FirstName:anshul
Lomdon ID:19031269
Sat 10 Apr 2021 03:42:37 PM +0545
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:
```

Figure 4: run with the correct password

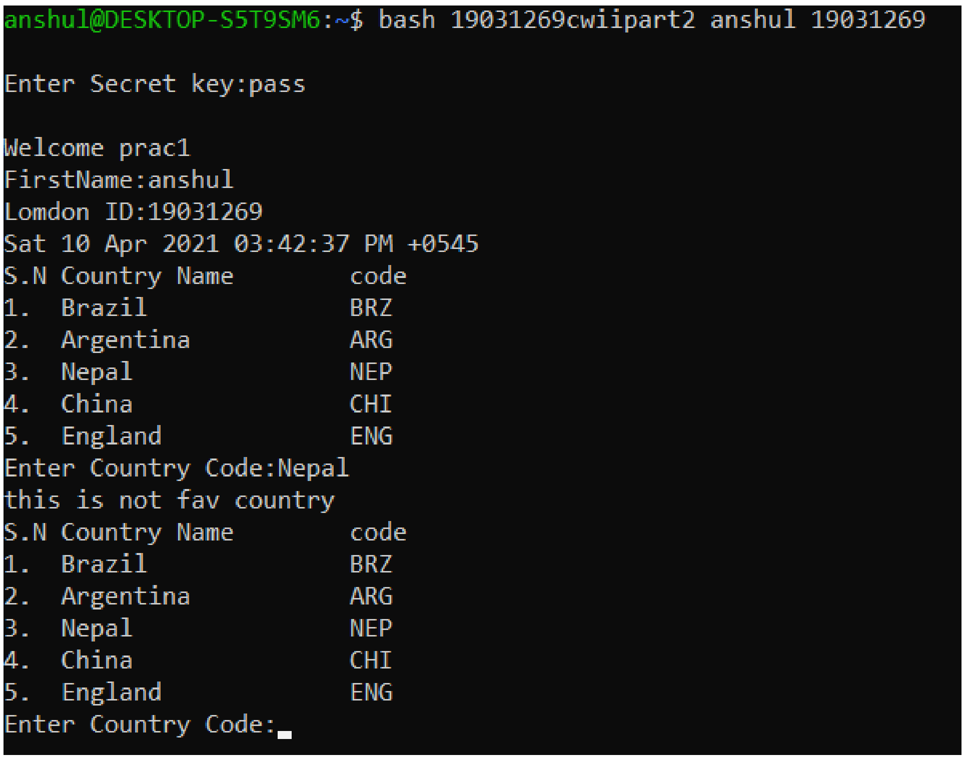
Here we have asked to run the program with the help of correct secret key from which the user is able to see the date and time , country name and their code.

3.5 Test 5

Test No	5
Input	Enter the full country name
Expected Output	Program will say this is not your favourite team and ask to enter again.
Actual Output	Try again

Table 5: run with the full country name

Screenshots:



```
anshul@DESKTOP-S5T9SM6:~$ bash 19031269cwiipart2 anshul 19031269

Enter Secret key:pass

Welcome prac1
FirstName:anshul
Lomdon ID:19031269
Sat 10 Apr 2021 03:42:37 PM +0545
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:Nepal
this is not fav country
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:_
```

Figure 5: run with the full country name

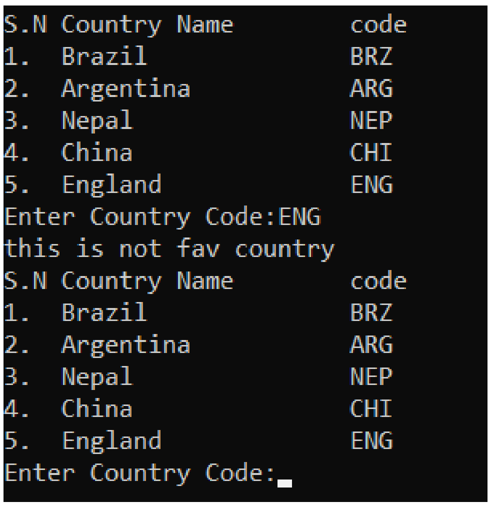
Here we have asked to run the program and keep the country name instead of the country to check what will happen.

3.6 Test 6

Test No	6
Input	Enter the incorrect code ENG
Expected Output	Program will say this is not your favourite team and ask to enter the code again
Actual Output	Try again

Table 6: incorrect country code

Screenshots:



```
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:ENG
this is not fav country
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:_
```

Figure 6: incorrect country code

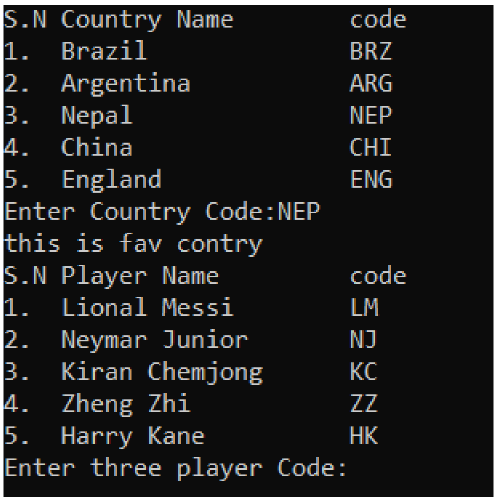
Here we have asked to run the program enter the wrong country code to check what will happen next.

3.7 Test 7

Test No	7
Input	Enter the correct code NEP
Expected Output	Program will run and ask about the favourite player in that team.
Actual Output	Success output.

Table 7: Correct country code

Screenshots:



```
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:NEP
this is fav contry
S.N Player Name         code
1.  Lional Messi        LM
2.  Neymar Junior       NJ
3.  Kiran Chemjong      KC
4.  Zheng Zhi           ZZ
5.  Harry Kane          HK
Enter three player Code:
```

Figure 7: Correct country code

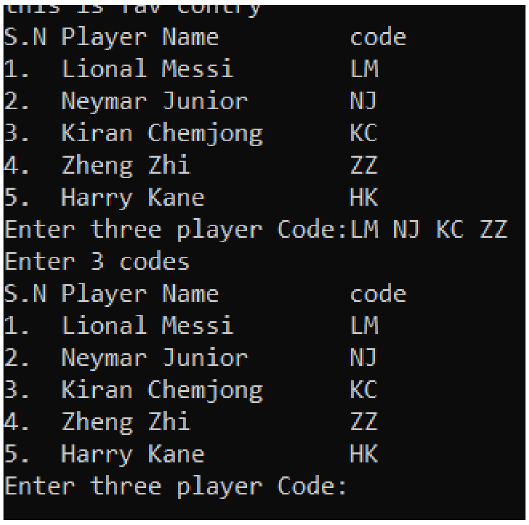
Here we have asked to run the program and enter the correct country code and see what next process the user have to do.

3.8 Test 8

Test No	8
Input	Enter four player name
Expected Output	Program will only accept three player
Actual Output	Asked to enter only three player.

Table 8: pick up four player name

Screenshots:



```
S.N Player Name         code
1.  Lional Messi        LM
2.  Neymar Junior       NJ
3.  Kiran Chemjong      KC
4.  Zheng Zhi           ZZ
5.  Harry Kane          HK
Enter three player Code:LM NJ KC ZZ
Enter 3 codes
S.N Player Name         code
1.  Lional Messi        LM
2.  Neymar Junior       NJ
3.  Kiran Chemjong      KC
4.  Zheng Zhi           ZZ
5.  Harry Kane          HK
Enter three player Code:
```

Figure 8: pick up four player name

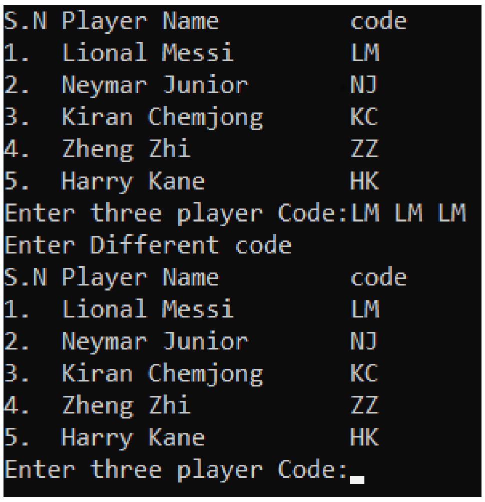
Here we have asked to run the program and choose the four player code and see what will happen because we have the limit of the three codes only and what will happen if we choose four.

3.9 Test 9

Test No	9
Input	Enter same player name for 3 times
Expected Output	Program will ask for to enter the three different code
Actual Output	Asked to enter three correct player code.

Table 9: pick same player name

Screenshots:



```
S.N Player Name         code
1.  Lional Messi        LM
2.  Neymar Junior       NJ
3.  Kiran Chemjong      KC
4.  Zheng Zhi           ZZ
5.  Harry Kane          HK
Enter three player Code:LM LM LM
Enter Different code
S.N Player Name         code
1.  Lional Messi        LM
2.  Neymar Junior       NJ
3.  Kiran Chemjong      KC
4.  Zheng Zhi           ZZ
5.  Harry Kane          HK
Enter three player Code:_
```

Figure 9: pick same player name

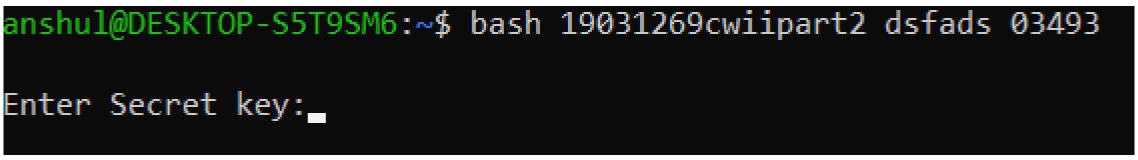
Here we have asked to run the program and write the same player code for the three time and see what will happen if we keep same code for three times.

3.10 Test 10

Test No	10
Input	Enter wrong username and id
Expected Output	Program will run ask for secret key
Actual Output	Successful

Table 10: run with the wrong username and id (parameter validation)

Screenshots:



```
anshul@DESKTOP-S5T9SM6:~$ bash 19031269cwiipart2 dsfads 03493

Enter Secret key:
```

Figure 10: run with the wrong username and id (parameter validation)

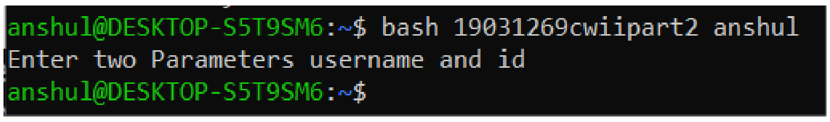
Here we have asked to run the program to enter the wrong username and id to check whether the program will run or not.

3.11 Test 11

Test No	11
Input	Enter only one parameter
Expected Output	Program will not run and tell to enter the two parameter which is username and id
Actual Output	Program was not able to execute.

Table 11: run by giving only one parameter

Screenshots:



```
anshul@DESKTOP-S5T9SM6:~$ bash 19031269cwiipart2 anshul
Enter two Parameters username and id
anshul@DESKTOP-S5T9SM6:~$
```

Figure 11: run by giving only one parameter

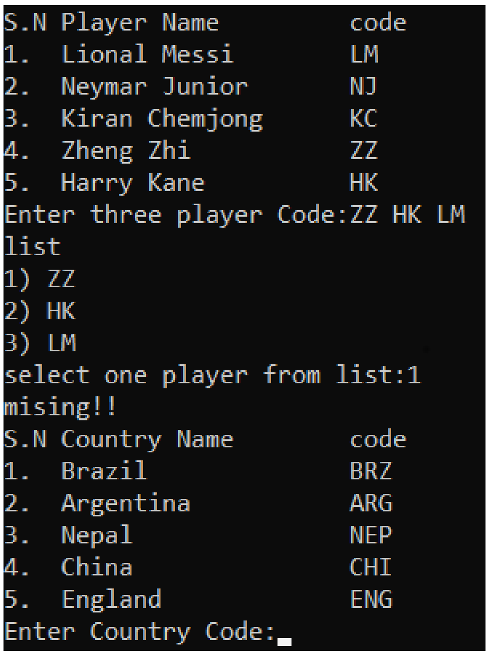
Here we have asked to run the program and the user to keep only the username and not keep the id and check the whether the program will run or not.

3.12 Test 12

Test No	12
Input	Enter wrong player id whose file has not been made
Expected Output	Program will not run and tell to that the file is missing
Actual Output	Program was not able to find the file.

Table 12: invalid player id

Screenshots:



```
S.N Player Name         code
1.  Lional Messi        LM
2.  Neymar Junior       NJ
3.  Kiran Chemjong      KC
4.  Zheng Zhi           ZZ
5.  Harry Kane          HK
Enter three player Code:ZZ HK LM
list
1) ZZ
2) HK
3) LM
select one player from list:1
mising!!
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:_
```

Figure 12: invalid player id

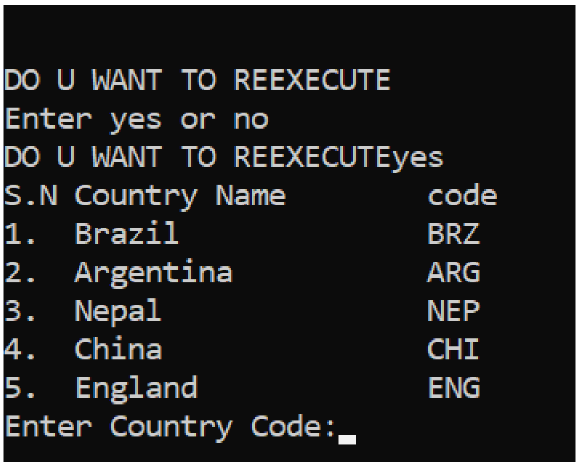
Here we have asked to run the program and choose one player from the list and the message is generated to enter the correct id.

3.13 Test 13

Test No	13
Input	Exit Yes
Expected Output	Program will start to run again
Actual Output	Program started to run again

Table 13: Exit Yes

Screenshots:



```
DO U WANT TO REEXECUTE
Enter yes or no
DO U WANT TO REEXECUTEyes
S.N Country Name        code
1.  Brazil              BRZ
2.  Argentina           ARG
3.  Nepal               NEP
4.  China               CHI
5.  England             ENG
Enter Country Code:_
```

Figure 13: Exit Yes

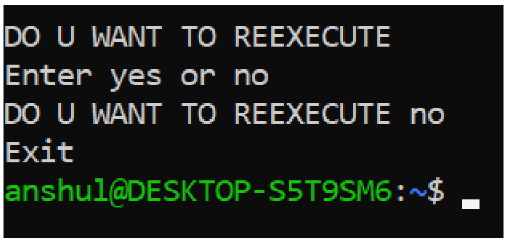
Here we have asked to run the program to enter YES on Exit and to run the program from the first again.

3.14 Test 14

Test No	14
Input	Exit NO
Expected Output	Program will get shutdown
Actual Output	Successful output.

Table 14: Exit No

Screenshots:



```
DO U WANT TO REEXECUTE
Enter yes or no
DO U WANT TO REEXECUTE no
Exit
anshul@DESKTOP-S5T9SM6:~$ _
```

Figure 14: Exit No

Here we have asked to run the program to enter NO the program will not run again and then it will get close.

Contents of three files: (TEXTS)

- LM

Lionel Messi, in full lionel Andres Messi also called Leo Messi (born on June 24 1987, Rosario Argentina), Argentine- born football (Soccer) player who was named Federation International de Football Association (FIFA) world player of the year five times (2009-12 and 2015).

.

- NJ

Neymar da Silva Santos Junior is a professional Brazilian footballer, who plays as a forward for the national team and French club Paris Saint Fermain. Commonly Known as Neymar, he was born 5 Febuary 1992 in Mogi das Cruzs Brazil. Neymar is regarded as one of the best players of his generation. The versatile attacker is dominate on the left wing but can play on the right wing and as a striker or second stiker as well. The Brazilian has 202 club goals from 346 appearances and 53 goals from 83 international caps.

- KC

Kiran Chemjong (born 1990 -03-20) is currently enrolled in goalkeeper positions from the Three Star Club and Nepali National Team. After graduating from ANFA Academy Chemjong joined Machhindra Football Club where he spend a year. After showing impressive performance between the sticks for Red Lions, Mega Three Club offered him a job in 2065 B.S. He was won many tournaments with Mega Three Star Cklub including British Gurkha Cup and Aaha Gold Cup Football Tournament.

Conclusion

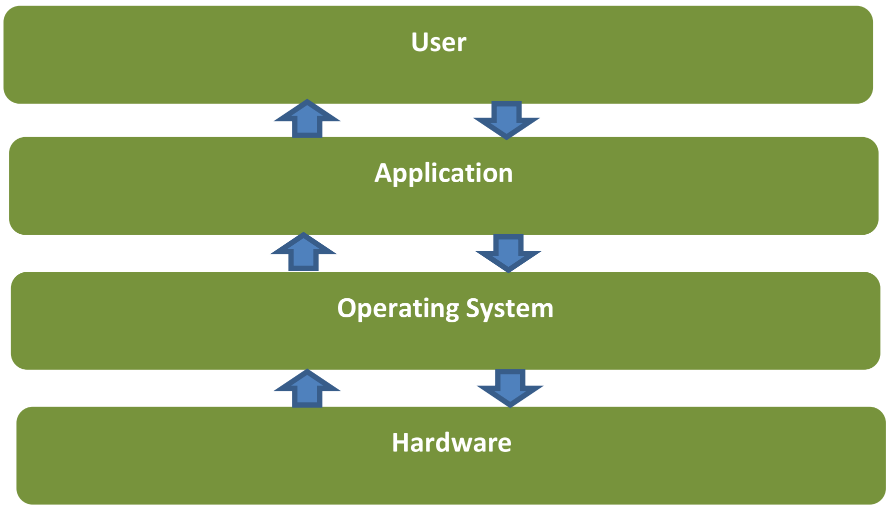
The first task is all about the UNIX commands, from the help of UNIX commands we developed a small program using small UNIX commands like shell, script, bash script. The UNIX command was run under the virtual machine. Linux is an open platform to that run on any kind of the platforms. All commands are written under the single Linux shell terminal.

This course helped me to improve my knowledge and skill to understand the UNIX command and the debian shell terminal. This course helped to understand the basic knowledge about the bash shell script. Doing this coursework was not easy I have to face my challenges to complete this coursework. After making lots of slip-up and blunder to overcome from this I took help from my teacher and colleagues due to their help I was able to complete this coursework time.

Task B

Introduction and Background

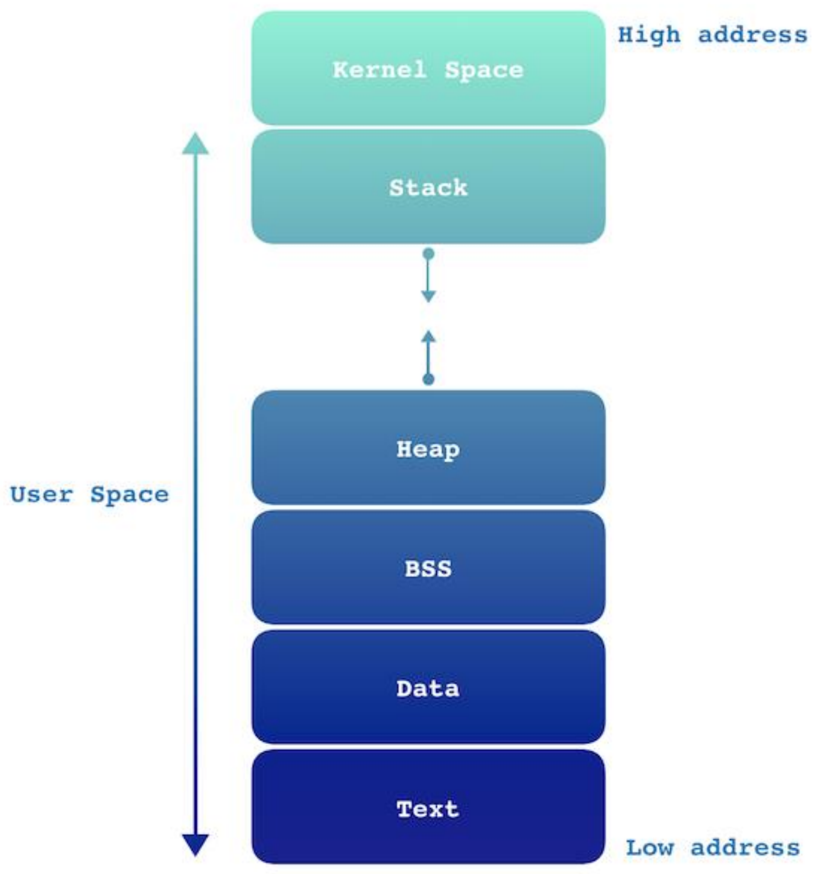
Operating System (OS): An operating system (OS) is a software programmer that links a computer user to the hardware.



Memory Organizer: Memory management is the method of managing and organizing computer memory by allocating parts of memory known as blocks to different operating programmers in order to improve the system's overall capacity. The primary role of an OS is to handle primary data. The maintenance of primary memory is the most essential feature of an operating system. It enables the transfer of processes between main memory and the execution disc. It supports the operating system in maintaining track of all memory locations, whether they are assigned to a process or remain unallocated.

Virtual address space is used by the multitasking OS. Memory addresses are reserved by 8 bytes in 64-bit systems, 4 bytes in 32-bit systems, and 2 bytes in 16-bit systems. The smallest unit addressable by the CPU is 1 byte, and this number is known as address size (8 bit). When the programme runs, the system divides the computation into two spaces: Kernel Space and User Space. The two computing spaces implicitly interfere with one another, and the programme processing continues.

Memory layout



Use of memory Management:

• It helps you to see how much memory needs to be distributed to the processes that specify which processors get memory when.

• Keeps track of whether product is released or unallocated. It can, according to it, refresh the status.

• It gives application routines their own space.

• It also ensures that these apps don't conflict with one another.

• It puts the programmes in memory, allowing memory to be used to its fullest capacity.

Types of memories:

Memory types: There are two types of computer memory: primary memory (RAM and ROM) and secondary memory (hard drive,CD,etc.). Read Only Memory (ROM) is primary-non-volatile memory, while Random Access Memory (RAM) is primary-volatile memory. It's also known as read-write memory, primary memory, or main memory.

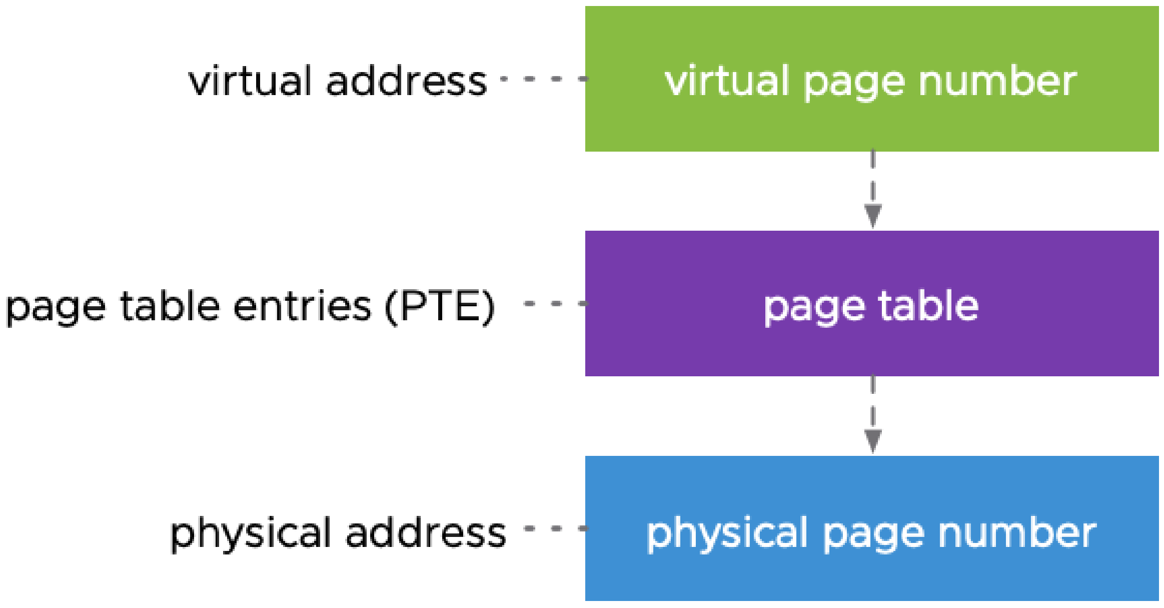
Memory is divided into two types: physical and abstract memory (internal storage of data). Physical memory can be used on chips (RAM memory) as well as storage devices and hard discs. A method must first load into RAM physical memory before it can be executed (also termed main memory).

When a device's physical memory is limited, it often causes system-wide delays or, in serious situations, a full system hang. This chapter explains how the operating system manages physical memory, how to recognise low-memory states, and how to resolve them.

Despite today's network response capability, network acquisition is still a worry. Owing to a lack of suitable or forensically sound software that would enable network retrieval, the incident responder and forensic analyst were previously limited to physical acquisition.

Inside a network, there are many factors to remember when it comes to rights and permissions. New protection features in Windows 7 and 64-bit operating systems can prevent responders from running their tools and capturing related data sets. . This is one reason why responders should have higher rights than regular users and, in the case of independent contractors, should collaborate as closely as possible with network administrators. For example, in Windows 7, the User Authentication Control (UAC) prevents the execution of applications that need elevated privileges to gather such data sets.

Data can be continuously shared between the hard disc and RAM memory of modern operating systems thanks to virtual memory. Data is exchanged via virtual memory using a method known as swapping. Since virtual memory enables the replication of the relocation of whole blocks of data, it seems like a machine has more RAM space. This allows programmes to run seamlessly and efficiently. Data is written to the hard disc rather than having to fit it into the often-limited volatile RAM memory. As a result, the amount of virtual memory is only restricted by the hard disk's size, or the space assigned to virtual memory on the hard disc. When information in RAM is required, the exchange mechanism quickly switches memory blocks (also known as pages of memory) between RAM and the hard disc.



Hardware memory management

At the hardware level, memory control is associated about the computing systems that currently contain data. RAM and memory caches are examples of this.

Operating system memory management

Memory must be assigned to user applications in the operating system, and then reclaimed by other programmes until it is no longer used. Digital memory systems allow the operating system to pretend that the device has more memory than it really does, as well as that each programme has all of the machine's memory to itself.

Application memory management

Application memory management entails allocating memory for a program's objects and data structures from the finite storage available, then recycling the memory until it is no longer used. Since application programmes can't anticipate how much memory they'll use in advance, they need additional code to cope with their changing memory requirements.

Application memory management combines two related tasks:

• Allocation: When a programme demands a block of memory, the memory manager must separate the block from the larger blocks the operating system has given. The allocator is the part of the memory manager that does this. Allocation can be done in a variety of ways, some of which are covered in Allocation strategies.

• Recycling: As memory blocks are distributed but the data they hold is no longer used by the software, the blocks may be recycled and reused. There are two methods to memory recycling: When memory should be reused, either the programmer or the user must specify (known as manual memory management)

• CPU overhead: The time it takes the memory manager to complete an operation and return control to the computer while the programme is running.

• Delay times: The time it takes the memory manager to complete an operation and return control to the programme.

Main memory has two partitions:

Low memory: The operating system is stored in this memory.

Large memory: This memory stores user processes.

Memory Management Techniques:

Single Contiguous Techniques: This is the simplest method of memory control. In this process, all forms of computer memory are usable for one operation, with the exception of a limited section reserved for the operating system. This is how the MS-DOS operating system, for example, allocates memory. A single programme often operates on an embedded device.

Partitioned Allocation: It splits primary memory into several memory partitions, most of which are adjacent memory zones. Any partition includes all of the data for a single task or work. This strategy entails allocating a partition to a job as it begins and leaving it unallocated when it stops.

Paged memory control: splits the primary memory of the device into fixed-size units known as page frames. This hardware memory management unit partitions pages into frames and allocates memory on a page-by-page basis.

Segmented Memory Management: it is the only memory management technique that does not supply the user's programme with a linear and contiguous address space.

Operating system uses following memory allocation mechanism:

Single-partition allocation: The relocation-register scheme is used to protect user processes from each other as well as from modifying operating-system code and data in this form of allocation. The value of the smallest physical address is stored in the relocation register, while the range of logical addresses is stored in the limit register. The limit register must be less than one logical address.

Multiple-partition allocation: This method of allocation divides main memory into a number of fixed-size partitions, each of which can only contain one operation. When a partition becomes available, a procedure is chosen from the input queue and loaded into it. The partition becomes available for another process until the process finishes. (Wilison) (Guru99)

Aims and Objectives

The main aim and objective of this coursework was giving to understand the details of the memory issues, how to allocate the memory and manage it. The aim and objective of memory is listed below:

Aims:

- The ambition is to possess innumerable processes from meddlesome with one another.
- To hoard the subsequent programme in memory so that it can be used to its thoroughgoing bulk.
- Accomplishes the virtual discourses planetary of the method, which is defined as a non- core dweller.

Objective:

- It taught us about most of the technical knowledge about the memory management.
- It taught us how to manage the memory and gave the knowledge about other memory allocation.
- Got the knowledge about the different types or memory and their function and how they help memory to play their role.

Background:

Memory Placement

Memory Placement Techniques Where should incoming processes be placed? Using a first-fit approach Process is inserted into the first hole that is large enough. Best-fit technique is simple and has a low overhead in terms of execution time. Process should be put in a hole with the least amount of available space surrounding it. More overhead in terms of execution time –worst-fit plan Process is situated in a hole that has a lot of empty space surrounding it. Another big void is created, making it more likely that another mechanism will be able to fit into it.

Memory Swapping and Multiprogramming Inactive systems do not need to be held in memory –Swapping Just placed the currently operating loop in main memory; some would be transferred to secondary storage for the time being. –Significant overhead when transferring processes –Maximizes usable memory Better still, retain several processes in mind at the same time – –Much quicker reaction times –Similar to paging –Less usable memory. Multiprogramming in a switching device where only one operation is in main memory at a time. 9.10 Memory Swapping and Multiprogramming.

Page coloring

Page colouring is a speed enhancement that makes the most of the processor cache when accessing contiguous pages in virtual memory. Cpu caches in ancient times (i.e. 10+ years ago) appeared to map internal memory rather than physical memory. This resulted in a slew of issues, including the need to clear the cache on any context swap in some situations and data aliasing issues in the cache. Modern processor caches specifically map physical memory to overcome these issues. This means that two adjacent pages in a process's address space might not equate to two adjacent pages in the cache. In reality, if you're not careful, side-by-side virtual memory pages might end up using the same processor cache page, causing cacheable data to be thrown away prematurely and lowering CPU efficiency. Also for multi-way set-associative caches, this is so (though the effect is mitigated somewhat).

Page colouring optimizations are implemented in FreeBSD's memory allocation code, which ensures that the memory allocation code can try to find free pages that are contiguous from the cache's perspective. The page colouring code would not allocate page 20 of physical memory to page 1 of a process's virtual memory if page 16 of physical memory is allocated to page 0 of a process's virtual memory and the cache will accommodate 4 pages. Instead, it will allocate physical memory page 21.

Since page 20 maps over the same cache memory as page 16, the page colouring code tries to prevent assigning it. This will result in less-than-optimal caching. As you would expect, this code brings a large amount of sophistication to the VM memory allocation subsystem, but the end product is well worth the effort. In terms of cache performance, page colouring renders VM memory as deterministic as physical memory.

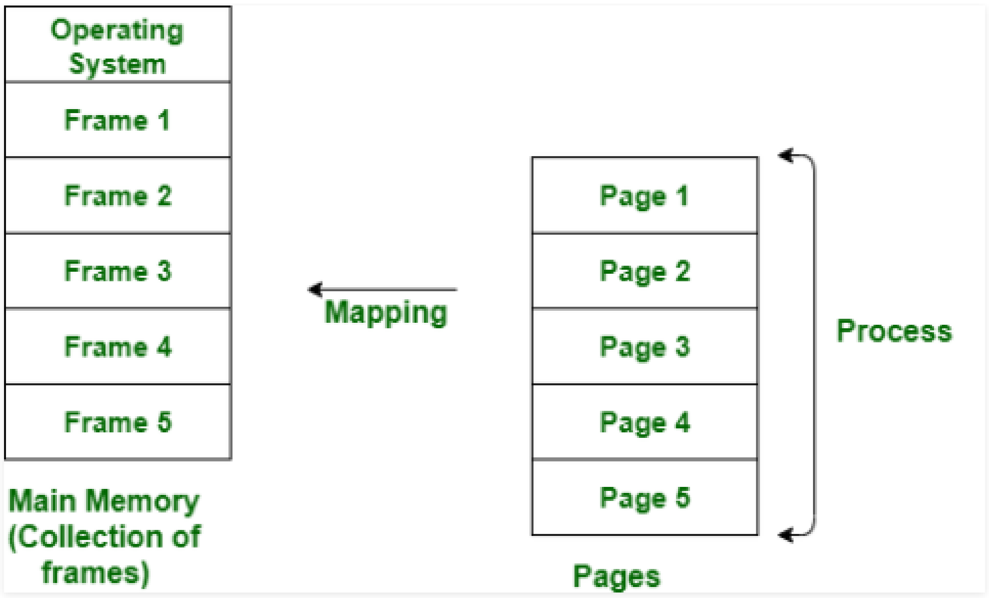
Description of Paging and Segmentation

This code adds a lot more complexity to the VM memory allocation subsystem, as you would imagine, but the end result is well worth the effort. Page colouring makes VM memory as deterministic as physical memory in terms of cache efficiency.

Paging

A method address space is divided into fixed-size blocks called pages in paging. A method address space is divided into portions of different sizes during segmentation. The memory is divided into pages by the operating system. Both primary and secondary memory are split into identical fixed-size partitions during paging. Pages and frames are the partitions of the secondary memory area unit and the primary memory area unit, respectively.

Paging is a memory management technique that uses pages to fetch processes from secondary memory into the primary memory. In paging, each method is divided into bits, each of which is the same size as the page. The last half's size may also be

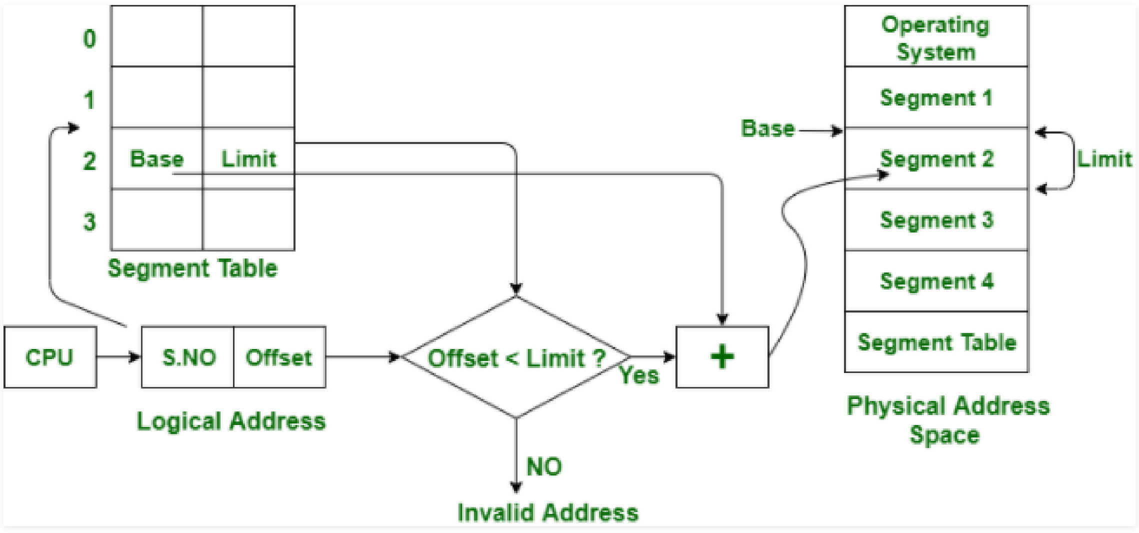


determined by the page size. The pages of the process area unit are held inside the main memory frames based on their usability.

Segmentation

Like paging, segmentation is a non-contiguous memory allocation scheme. Segmentation, like paging, does not break processes indiscriminately into mounted(fixed) size pages. It's a theme of variable partitioning sizes. Segmentation, like paging, does not split secondary and primary memory into equal-sized partitions. Segments are the partitions of the secondary memory area network. The data for each section are held in a table called the segmentation table. The Segment table includes two key pieces of information about segments: Base, which is the segment's bottom address, and Limit, which is the segment's length.

During segmentation, the CPU creates a logical address that includes the segment number and offset. If the section offset is less than the cap, the address is considered valid; otherwise, it results in a miscalculation since the address is invalid. (Yokoyama, 2018) (Player) (geeks g. f., 2020) (geeks G. f., 2020)



Page Size variation

There are some trade-offs between small and big page sizes.Internal fragmentation wastes less memory on small pages.Smaller page tables are used for large pages.The latency and seek times for disc access far outweigh the real data transfer times. This makes transferring one big page of data much easier than transferring two or three smaller pages with the same amount of data. Since we aren't dragging in data that isn't really required, smaller pages best fit locality.Smaller pages result in more page errors, resulting in more overhead.Page size can also be influenced by the physical hardware.It's difficult to say what the "ideal" page size is for any particular device. The current standards vary from 4K to 4M, with greater page sizes being more popular as time goes by. (jbell)

Conclusion

Memory management is the method of managing and organizing computer memory by allocating parts of memory known as blocks to different operating programmers in order to improve the system's overall capacity. Memory management include some different parts like RAM, Chips and many more. The main objective and purpose was to describe about the memory issues and clearly understand the newly topic and get the knowledge about that topic with the help of the research.

This whole report was about the memory issues and management. This report was completed with the help of the teacher and colleagues.

Bibliography

Geeks, G. f. (2019, 08 16). *virtual memory in operating system*. Retrieved 04 05, 2021, from Geeks for greeks : https://www.geeksforgeeks.org/virtual-memory-in-operating-system/

geeks, G. f. (2020, 02 04). *difference between paging and segmentation*. Retrieved 04 05, 2021, from Geeks for geeks: https://www.geeksforgeeks.org/difference-between-paging-and-segmentation/

geeks, g. f. (2020, 11 06). *partition Allocation method in memory management*. Retrieved 04 05, 2021, from greeks for geeks: https://www.geeksforgeeks.org/partition-allocation-methods-in-memory-management/

greeks, g. o. (2017, 09 26). *introduction to linux shell and shell scripting*. Retrieved 04 1, 2021, from greeks of geeks : https://www.geeksforgeeks.org/introduction-linux-shell-shell-scripting/

Guru99. (n.d.). *Guru99*. Retrieved 03 24, 02021, from https://www.guru99.com/os-memory-management.html#2

Hiwarale, U. (2019, 09 8). *Bash Scripting: Everything you need to know about the Bash-Shell programming*. Retrieved 04 01, 2021, from medium : https://medium.com/sysf/bash-scripting-everything-you-need-to-know-about-bash-shell-programming-cd08595f2fba

JAVAtpoint. (n.d.). *cache memory* . Retrieved 04 05, 2021, from javatpoint: https://www.javatpoint.com/cache-memory

jbell. (n.d.). *cs.uic.edu*. Retrieved 04 05, 2021, from virtual memeory: https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

Player, S. (n.d.). *memory placement strategies.* Retrieved 04 5, 2021, from Slide player : https://slideplayer.com/slide/5116714/

Tutorialspoint. (n.d.). *shell scripting tutorial*. Retrieved 04 1, 2021, from tutorialspoint: https://www.tutorialspoint.com/unix/shell_scripting.htm

utmel. (2020, 08 12). *what is a memory controller?* . Retrieved 04 05, 2021, from utmel: https://www.utmel.com/blog/categories/memory%20chip/what-is-a-memory-controller

Wilison. (n.d.). *Memory Management References*. Retrieved 3 23, 2021, from https://www.memorymanagement.org/mmref/begin.html

Yokoyama, S. (2018, 11 10). *understanding memory layout*. Retrieved 04 5, 2021, from medium: https://medium.com/@shoheiyokoyama/understanding-memory-layout-4ef452c2e709

Appendix

Memory Controller

The memory controller is a crucial component of a computer system that manages memory and facilitates data transfer between the memory and the CPU. The memory controller specifies the computer system's full memory power, the amount of memory banks, memory type and speed, memory particle data depth and distance, and other critical parameters. Since the memory controller controls the computer system's memory performance, it has a greater effect on the system's overall performance. There are different types of memory controller:

- Traditional memory controller.
- Integrated memory controller (utmel, 2020)

Virtual Memory

Virtual Memory is a data allocation structure that allows you to address secondary memory as if it were primary memory. The addresses used by a computer to refer to memory are distinct from the addresses used by the memory system to locate physical storage locations, and programme derived addresses are immediately converted to machine addresses.

The capacity of virtual storage is restricted by the computer system's addressing scheme, and the volume of secondary memory available is not limited by the number of main storage locations. It's a method that uses both hardware and software to work. It converts abstract addresses, which are used by programmes, into physical addresses in computer memory.

- Inside a loop, all memory references are logical addresses that are dynamically converted into physical addresses at run time. This ensures that during execution, a procedure can be swapped in and out of main memory, occupying various locations in main memory at different times.
- A procedure can be broken down into several bits, and these pieces do not need to be kept in main memory at all times during execution. This is possible

thanks to a combination of dynamic run-time address translation and the use of a page or section table. (Geeks, 2019)

Cache Memory

Cache memory is a form of high-speed memory that is smaller than main memory but faster (RAM). It can be accessed by the CPU much faster than main memory. As a result, it's used to synchronise with a fast CPU and boost its performance. Just the CPU has access to cache memory. That may be a section of main memory set aside for it or a storage unit external to the CPU. It stores the data and programmes that the CPU uses most. As a result, it ensures that the data is immediately accessible to the CPU anytime it is needed. To put it another way, if the CPU detects the data or instructions it needs in the cache memory, it doesn't need to enter the main memory (RAM). As a result, it improves processor stability by serving as a buffer between RAM and the CPU. (JAVAtpoint)