

**Module Code & Module Title**

**CS5001NA Networks and Operating System**

**Assessment Weightage & Type**

**20% Individual Coursework**

**Year and Semester**

**2019-20 Autumn**

**Student Name: Himanshu pandey**

**London Met ID: 19031311**

**College ID:NP01NT4A190131**

**Assignment Due Date: 11 Apr 2021**

**Assignment Submission Date: 11 Apr 2021**

**Word Count (TASK B):2000**

*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.*

Contents

[Introduction 3](#_Toc69035751)

[Script 4](#_Toc69035752)

[Contents of three files: (TEXTS) 22](#_Toc69035753)

[LM 23](#_Toc69035754)

[23](#_Toc69035755)

[NJ 23](#_Toc69035756)

[23](#_Toc69035757)

[KC 23](#_Toc69035758)

[23](#_Toc69035759)

[Introduction 24](#_Toc69035760)

[Dynamic Storage Allocation 29](#_Toc69035761)

[**Segmentation** 35](#_Toc69035762)

[References 38](#_Toc69035763)

[Appendix 39](#_Toc69035764)

[**Memory Controller** 39](#_Toc69035765)

[**Virtual Memory** 39](#_Toc69035766)

**Table of figure**

[Table 1 passing one parameter while running program 15](#_Toc68986861)

[Table 2 passing valid parameters 15](#_Toc68986862)

[Table 3entering incorrect secret key 16](#_Toc68986863)

[Table 4entering secret key 17](#_Toc68986864)

[Table 5entering country code 17](#_Toc68986865)

[Table 6 entering invalid code 18](#_Toc68986866)

[Table 7 entering valid code 19](#_Toc68986867)

[Table 8 selecting wrong player 20](#_Toc68986868)

[Table 9 selecting valid list 21](#_Toc68986869)

[Table 10 restarting program 22](#_Toc68986870)

[Table 11 terminating program 22](#_Toc68986871)

[Table 12hechiery of memory management 27](#_Toc68986872)

[Table 13temproral locality 33](#_Toc68986873)

[Table 14spatial locality 34](#_Toc68986874)

[Table 15memory acolation **Error! Bookmark not defined.**](#_Toc68986875)

[Table 16pagging 35](#_Toc68986876)

[Table 17segmentation 36](#_Toc68986877)

**TASK A**

# Introduction

The Linux operating system shell accepts, processes, and executes commands. This is the user interface where the user interacts with programs, commands, and scripts. The shell can be accessed through a terminal that is running it.

When you run the terminal, the Shell prompts you for a command line (usually $) where you can type whatever you want. On the terminal, the output or result is displayed. (technadvice.com › types-of-linux-operating-systems, 2021)

To run the shell, type a series of SHELL SCRIPTING commands. Long and repetitive scripts can be combined into a single simple script that can be archived and run at any time. An user's effort is reduced as a result of this.

The script's instance is used to write a shell script that the program runs after the name and ID of the program's two parameters are specified. When a user is asked to enter a secret key, the program greets them by entering their name and ID, as well as the date and time of execution, if they enter the correct secret key, otherwise it asks for the key. Enter this code five times, then exit the registry. The user is now asked to determine which country has the best cricket team by looking at the names and codes of the five countries that have played the game. The program displays a description of the team after the user enters the code; otherwise, an appropriate error message is displayed, and the user can guess until the correct option is selected. The program then asks the user to enter three codes from the displayed list to match the names of the five players who have played cricket. Following the program, you will be presented with three code lists from which to choose. A description of the reader is displayed if the selected code contains a readable file; otherwise, the corresponding message is displayed. The user is asked at the end if he wants to continue the program, and if he does, the process described above will continue if the user connects, and if the program will not end.

As a result, the applied activity is carried out by creating an interactive request that employs UNIX commands, user input for error checking, diagnosis of these errors with the appropriate message, and script testing.

## Script

This includes the code of the program

#!/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 Parameters"

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

**Testing**

**Test1**

***Screenshots***

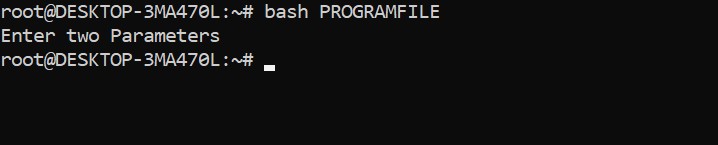


Figure 1passing one parameter while running program

|  |  |
| --- | --- |
| Test No | 1 |
| Input | bash PROGRAMFILE |
| Expected Output | Program should not run displaying message. |
| Actual Output | Program didn’t run displaying message. |
| Result | Test was successful. |

Table 1 passing one parameter while running program

**Test2**

***Screenshots***

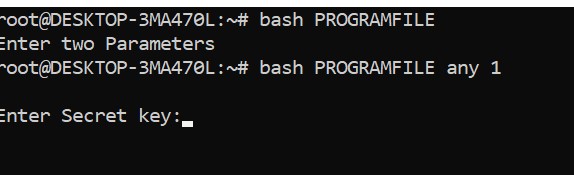


Figure 2passing valid parameter

|  |  |
| --- | --- |
| Test No | 2 |
| Input | bash PROGRAMFILE any 1 |
| Expected Output | Program should ask for the password. |
| Actual Output | Program asked for the password. |
| Result | Test was successful. |

Table 2 passing valid parameters

**Test3**

***Screenshots***

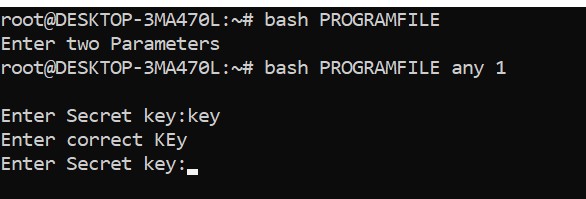
****

Figure 3enterring incorrect secret key

|  |  |
| --- | --- |
| Test No | 3 |
| Input | key |
| Expected Output | Program should display the wrong password message for five times and terminate the program. |
| Actual Output | Program displayed the wrong password message for five times and terminate the program. |
| Result | Test was successful. |

Table 3entering incorrect secret key

**Test4**

***Screenshots***

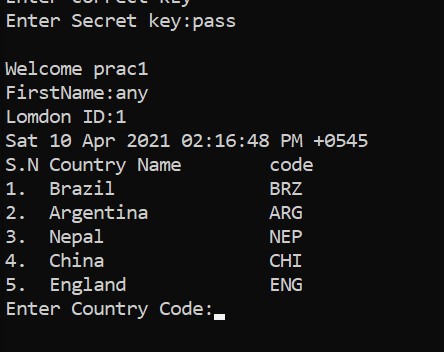
****

Figure 4 entering secret key

|  |  |
| --- | --- |
| Test No | 4 |
| Input | pass |
| Expected Output | Program should start with a message and date and time. |
| Actual Output | Program started with a message and date and time |
| Result | Test was successful. |

Table 4entering secret key

**Test5**

***Screenshots***

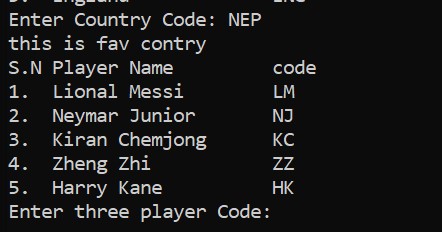
****

Figure 5 entering country code

|  |  |
| --- | --- |
| Test No | 5 |
| Input | NEP |
| Expected Output | The program should display the country and code in a list. |
| Actual Output | The program displayed the country and code in a list. |
| Result | Test was successful. |

Table 5entering country code

**Test6**

***Screenshots***

****

Figure 6 entering invalid code

Table 6 entering invalid code

|  |  |
| --- | --- |
| Test No | 6 |
| Input | LM NJ KC ZZ |
| Expected Output | The program should display the message of invalid player. |
| Actual Output | The program displayed the message of invalid player. |
| Result | Test was successful. |

**Test7**

***Screenshots***

****

Figure 7 entering valid code

|  |  |
| --- | --- |
| Test No | 7 |
| Input | NJ KC LM |
| Expected Output | The program should display the chosen player in the list. |
| Actual Output | The program displayed the chosen player in the list. |

Table 7 entering valid code

**Test8**

***Screenshots***

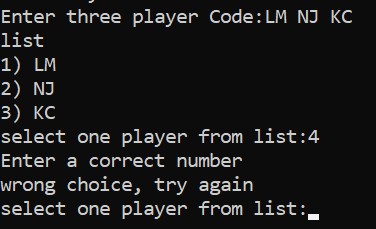
****

Figure 8 selecting wrong player

Table 8 selecting wrong player

|  |  |
| --- | --- |
| Test No | 8 |
| Input | 4 |
| Expected Output | The program should display the appropriate message for entering the wrong input. |
| Actual Output | The program displayed the appropriate message for entering the wrong input. |
| Result | Test was successful. |

**Test9**

***Screenshots***

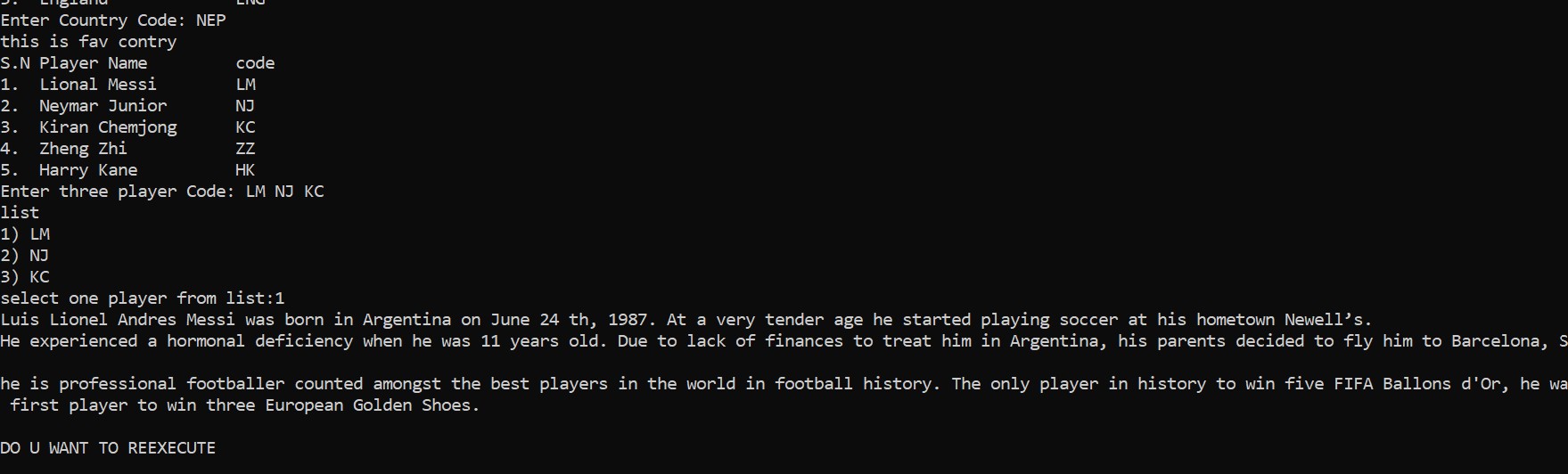
****

Figure 9selecting valid player

|  |  |
| --- | --- |
| Test No | 9 |
| Input | 1 |
| Expected Output | The program should display the description of the chosen player list. |
| Actual Output | The program displayed the description of the chosen player list. |
| Result | Test was successful. |

Table 9 selecting valid list

**Test10**

***Screenshots***

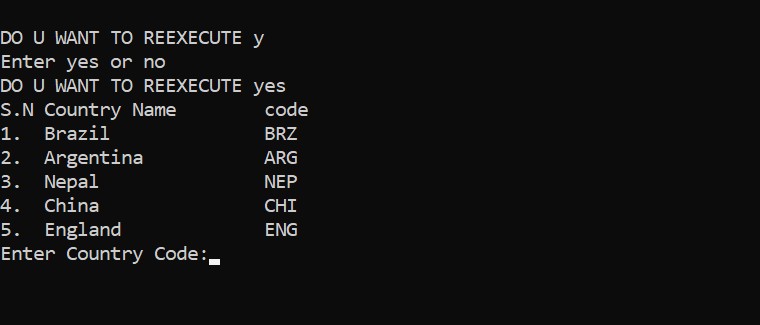
****

Figure 10 restarting program

|  |  |
| --- | --- |
| Test No | 10 |
| Input | Yes |
| Expected Output | The program should run again. |
| Actual Output | The program was started. |
| Result | Test was successful. |

Table 10 restarting program

**Test11**

***Screenshots***

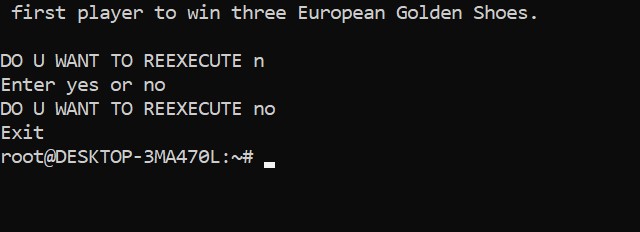
****

Figure 11 terminating the program

|  |  |
| --- | --- |
| Test No | 11 |
| Input | No |
| Expected Output | The program should terminate. |
| Actual Output | The program was terminated. |
| Result | Test was successful. |

Table 11 terminating program

## Contents of three files: (TEXTS)

### LM

### Luis Lionel Andres Messi was born in Argentina on June 24th , 1987. At a very tender age he started playing soccer at his hometown Newell’s. He experienced a hormonal deficiency when he was 11 years old. Due to lack of finances to treat him in Argentina, his parents decided to fly him to Barcelona, Spain. He is professional footballer counted amongst the best players in the world in football history. The only player in history to win five FIFA Ballons d'Or, he was also the first player to win three European Golden Shoes.

### NJ

### Neymar da Silva Santos Júnior (Brazilian Portuguese born in 5 February 1992) He is a Brazilian professional footballer who plays as a forward for Ligue 1 club Paris Saint-Germain and the Brazil national team. He is widely regarded as one of the best players in the world.

### KC

### Kiran Chemjong, also known as Kiran Kumar Limbu, is a Nepali footballer plays as a goalkeeper for Nepal national football team. he was born in march 24 2020. he is one of the famous player in nepal and counted as good player among other asian players.

**Conclusion**

The program required by course work 2 has been successfully developed and tested. The first part of the course covered the relevant UNIX commands, which introduce a small interactive program written in a well-known UNIX shell environment using the UNIX command and a scripting language, such as bash shell script. Thanks to the entire Linux operating system course, which was completed in a virtual machine. Linux is a free and open source operating system that runs on a variety of platforms. The Linux shell terminal was used to run all of the commands. The script that was added to the script had to write a shell script based on that scenario. As a result, the task was to use UNIX commands to create an interactive request, check user input for errors, diagnose those errors with an appropriate message, and test the script. I've gained a better understanding of UNIX commands and the Ubuntu shell terminal as a result of this course. Furthermore, this course aided in the development of the fundamentals and concepts of bash shell scripts. Using scripting language to present a small interactive program, on the other hand, was not an easy task. This course, on the other hand, has motivated me to learn more about UNIX-based operating systems and bash shell script.

**Task B**

## Introduction

The operating system for a network. (NOS) A network operating system that includes software that allows computers to communicate with one another over a network. This enables the sharing of resources such as files, application programs, and printers between computers. Berkeley Software Distribution Unix, Novell, LAntastic, and MS LAN Manager are some examples.

Task B of coursework 2 part 2 is to conduct research on network operating systems with an emphasis on memory issues. Memory management is an operating system feature that manages main memory and transfers processes back and forth between main memory and the disk during execution. Memory management keeps track of every memory location in the system, whether it is allocated to the process or not. Management checks the amount of memory allocated to each process at regular intervals. It goes through a series of steps to determine where it will get memory at any given time. Memory management checks the free or unallocated memory on a regular basis and updates the status as needed. (network-operating-system-rtos, 2019)

**Aims**

Task B focuses at memory management in the operating system, including descriptions of memory types, dynamic memory allocation, memory hierarchy, and reference position, and determines the course of investigation. This report's main goal is to present a technical report that can be used as a reference in the field of memory management. It is also known as computer architecture in the operating system, and it includes memory management and hierarchy, as well as a cache memory edition algorithm.

**Objectives**

The following are the objectives of the technical report:

• Research and self-knowledge from various sources, such as books, magazines, articles, online sites, and studies of operating system memory management.

• Keep track of your time to ensure a successful project completion.

**Background of memory management**

Memory is made up of a variety of words and actions, each with its own address. The CPU retrieves instructions from memory based on the program counter value. These counters lead to additional loading from a memory address that has been saved.

A program usually finds a disk as a binary executable file. For the program to be executed, it must be remembered and entered into the process. It is a process to move between disk and memory during its execution, depending on the memory management in use. The input queue is a collection of processes on disk that are waiting to run in memory. (www.cs.iit.edu › , 2018)

**Hierarchy of memory**

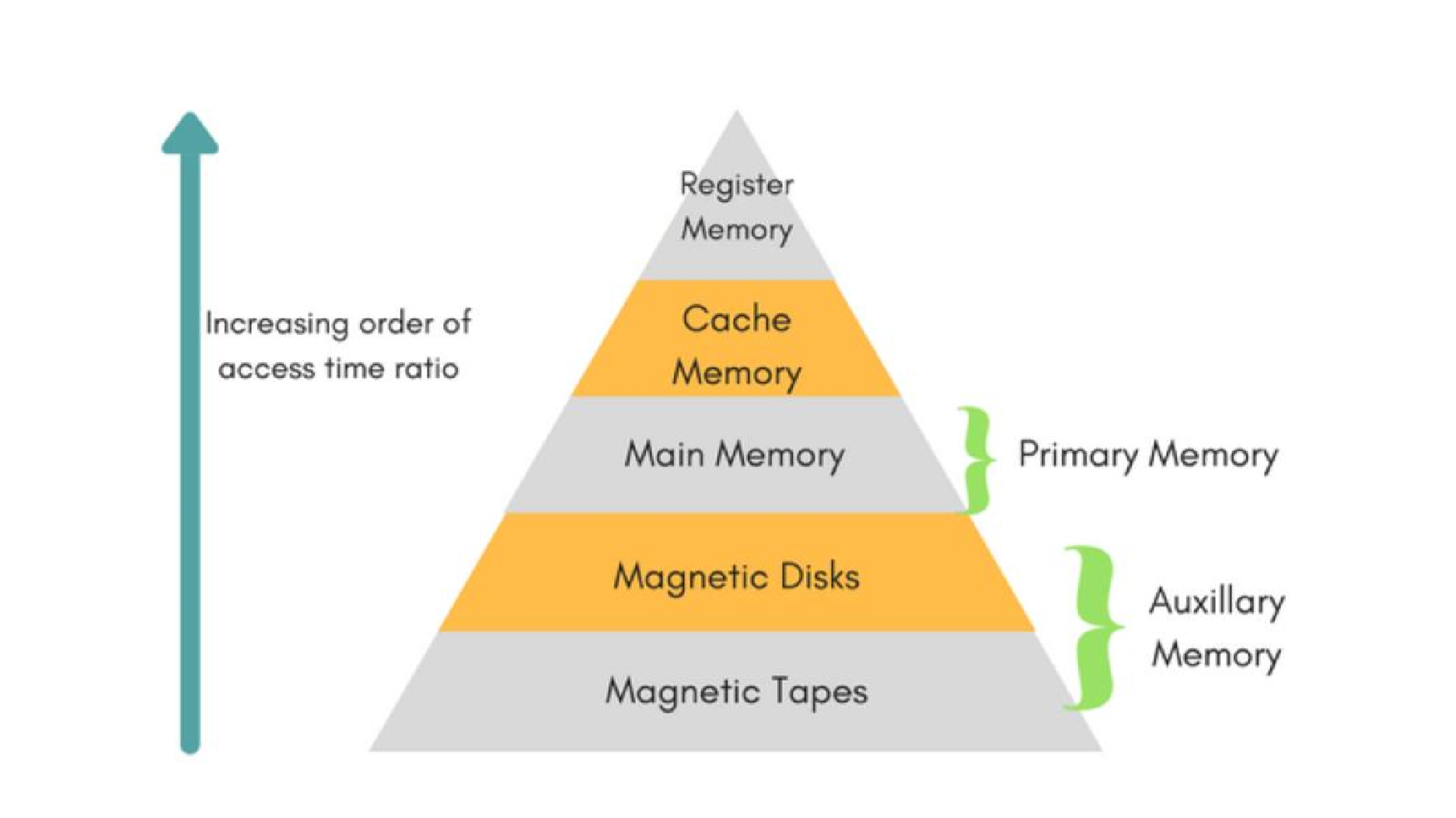
Although the main / auxiliary distinction is widely useful, computer memory organization creates a hierarchy of levels, ranging from the very small, fast, and expensive CPU program hierarchy to small, fast cache memory; larger DRAM; very large hard disks; and slow and cheap non-volatile backup storage.

Registers, cache, main memory, magnetic disks, and magnetic tapes are the five memory hierarchies. The first three hierarchies are volatile memory, which means that if you run out of energy, your saved data will be lost. The data is stored permanently despite the fact that the last two hierarchies are not volatile.

A memory item is a group of storage devices used to store binary data in bit format. In general, memory can be classified into two types: volatile and non-volatile. (Hemmendinger, 2016)

**Hierarchy of memory management**

Memory hierarchy is an increase or improvement in how memory is organized in order to reduce access times when designing a computer system. The memory hierarchy was created using a program behavior known as the district of references. The following diagram depicts the various levels of the memory hierarchy. (sap, 2018)

Table 12hechiery of memory management

(www.javatpoint.com ›, 2019)

Memory management is crucial in the operating system. It is concerned with memory and the movement of processes from disk to primary memory for execution, as well as the reverse. The following are some basic memory management concepts:

**Virtual address space and physical address space:**

The virtual address is the address generated by the CPU, while the physical address is the address seen by the memory. Both the virtual and physical addresses are the same in compile time and load time address binding schemes. Only the execution time address binding scheme differs.

The virtual address space is made up of all the logical addresses generated by a program, while the physical address space is made up of all the physical addresses that correspond to these logical addresses. (2018)

**Swapping**

A process is occasionally swapped out of main memory, that is, moved from main memory to secondary storage. This could be done to make room for other processes to run more quickly. Later, the swapped-out process can be swapped back in for execution.

**Allocation of Memory**

The operating system and software applications manage memory allocation, which is primarily a computer hardware operation. In the management of physical and virtual memory, the memory allocation process is very similar. Programs and services are given special memory based on the requirements for their implementation. When a program is completed or inactive, memory is freed and allocated to another program or merged with main memory.

There are two types of memory allocation;

**Static memory allocation:** The program allocates memory during compilation.

**Dynamic memory allocation:** programs run memory at runtime.

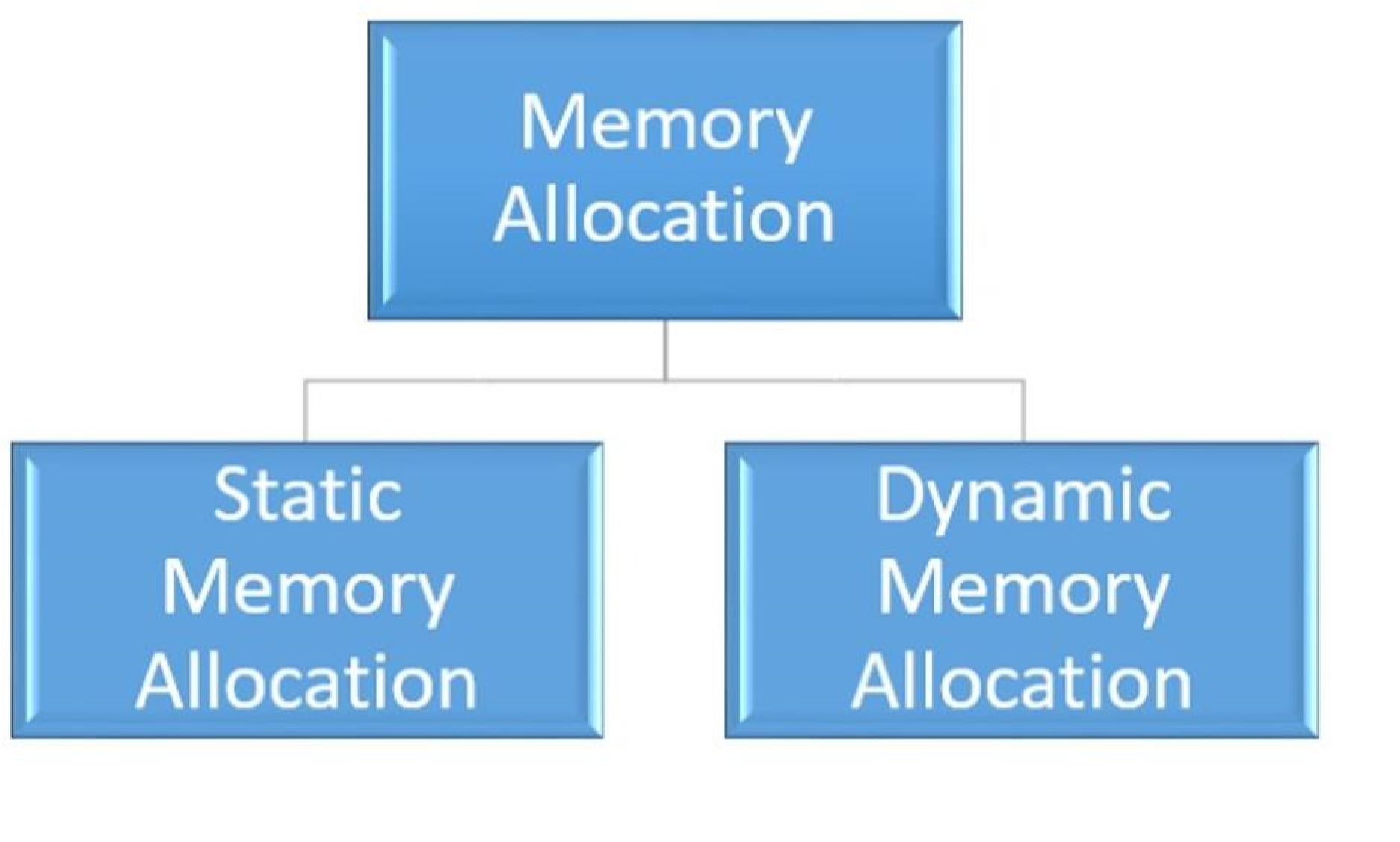


Table 13memory acolation

**Advantages of static and dynamic memory allocation**

**Static memory allocation**

* Allocating memory to a process is an effective way of allocating static memory.

* All memory allocation operations are performed before execution starts.

**Dynamic allocation of memory**

* Dynamic memory allocation allows flexible allocation of process memory.

* Dynamic memory allocation reduces memory waste by allocating memory to a process while the program is running.

# Dynamic Storage Allocation

Dynamic memory allocation is a memory distributed in the current period of the program and is a method of dividing the ownership of limited memory resources between many data and codes. A dynamically allocated object remains occupied until it is explicitly distributed, by the programmer or the garbage collector; this is significantly different from automatic and static memory allocation. The life cycle of such an object is said to be dynamic. The life cycle of such an object is said to be dynamic.

Satisfying an allocation request that involves finding an unused block of memory of a certain size in a heap is a difficult task. (www.geeksforgeeks.org, 2020)

**The Initial Fit:**

The first available memory space that is large enough for the process's needs is assigned to it. The search can begin at the beginning or where the previous first fit search ended.

**Best Match**:

The process is given the smallest memory block that will suffice for its needs. Unless the block list is kept in or, the best fit algorithm must search the entire block list.

**The Worst Fit:**

The largest memory block available is assigned to the process. Unless the block list is kept in order, the entire block list must be searched for this algorithm.

**Paging:**

Virtual memory, which is more than the physical memory available in a computer system, can be created. The concept of paging is used to achieve this.

The physical address space is divided into frames and the logical address space is divided into pages in paging. Both the pages and the frames are the same size. The number of pages used to measure the process.

By storing the pages into the frames, these pages are now brought from the logical address space to the physical address space. To determine which page is stored in which folder, a page table is created.

**Cache Memory**

Data or content from main memory that is frequently accessed by the CPU is cached so that the processor can access it more quickly. If the CPU requires memory, it first checks the cache for the required data. Data is read from fast memory when it is cached. Otherwise, the CPU is shifted to main memory to process the data. (www2)

The three types of cache are:

* + **L1 Cache**

A level 1 cache (L1 cache) is a memory cache that is inserted directly into the microprocessor and used to store information recently available to the microprocessor, hence the primary cache.

It is used to store data recently accessed by the processor, critical files that must be executed immediately and is the first cache that can be accessed and processed when the processor executes computer instructions.

* + **L2 Cache**

A level 2 cache (L2 cache) is a CPU cache memory that is located outside the core of a microprocessor chip and is separate although it is on a chip package on the same processor.

Incorporating L2 caches into the microprocessor design is very common in modern CPUs, although they may not be as fast as the L1 cache, but because they are off core, the capacity can be increased and still be faster than the main memory. The level 2 housing is also known as a secondary or external cache.

* + **L3 Cache**

A level 3 cache (L3) is a special cache used by the CPU and generally built on the motherboard and in some special processors in the CPU module itself. The L3 cache passes information to the L2 cache, which forwards the information to the L1 cache. Generally, memory performance is slower than L2 cache, but also faster than main memory (RAM). (eco)

**Main Memory**

Memory that points directly to the CPU, auxiliary memory, and cache memory is called main memory. Most of the memory is made up of RAM and ROM and most of the RAM chips. (2017)

**Locality of reference**

In operating systems, the concept of the reference site states that the operating system can only load the number of pages into the main memory accessed by the CPU and operating system, rather than loading the whole process into the main memory. you can only load table page entries suitable for multiple pages.

There are two kinds of locality of reference. They are:

* **Temporal Locality**
* **Spatial Locality**

**Temporal Locality –**

Current downloadable data or instructions may be needed soon. Therefore, these data or instructions must be stored in cache memory to avoid the same data in main memory. If we refer to some data, there is a high probability that we will refer to them soon.

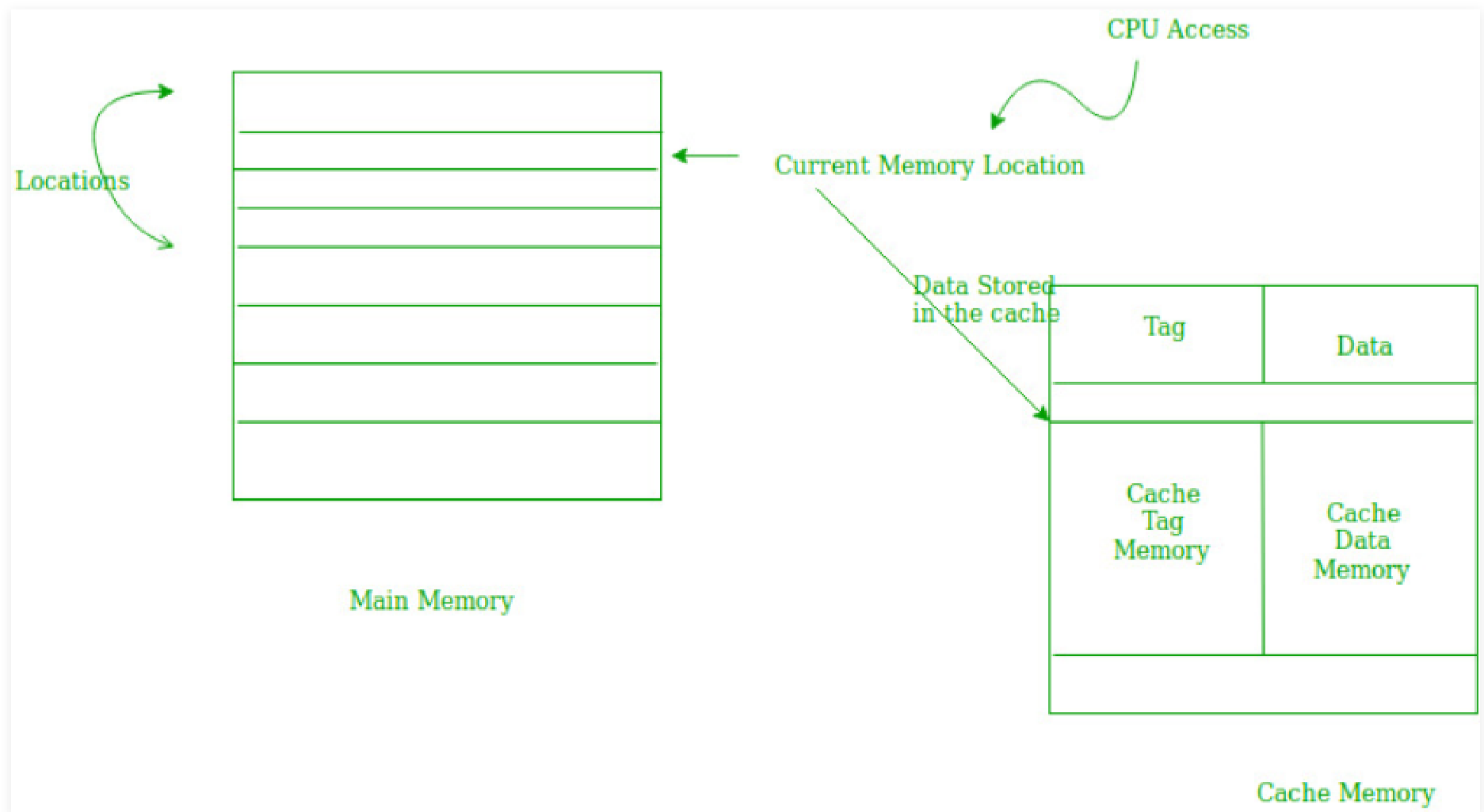


Table 14temproral locality

**Spatial Locality** **–**

The spatial area indicates that there are instructions or data close to the current scannable memory location, which may be needed in the near future. This is a little different from the weather. Here we are talking about memory locations that are almost present, while in space of time we are talking about the actual memory location that we are recovering from. (sta)

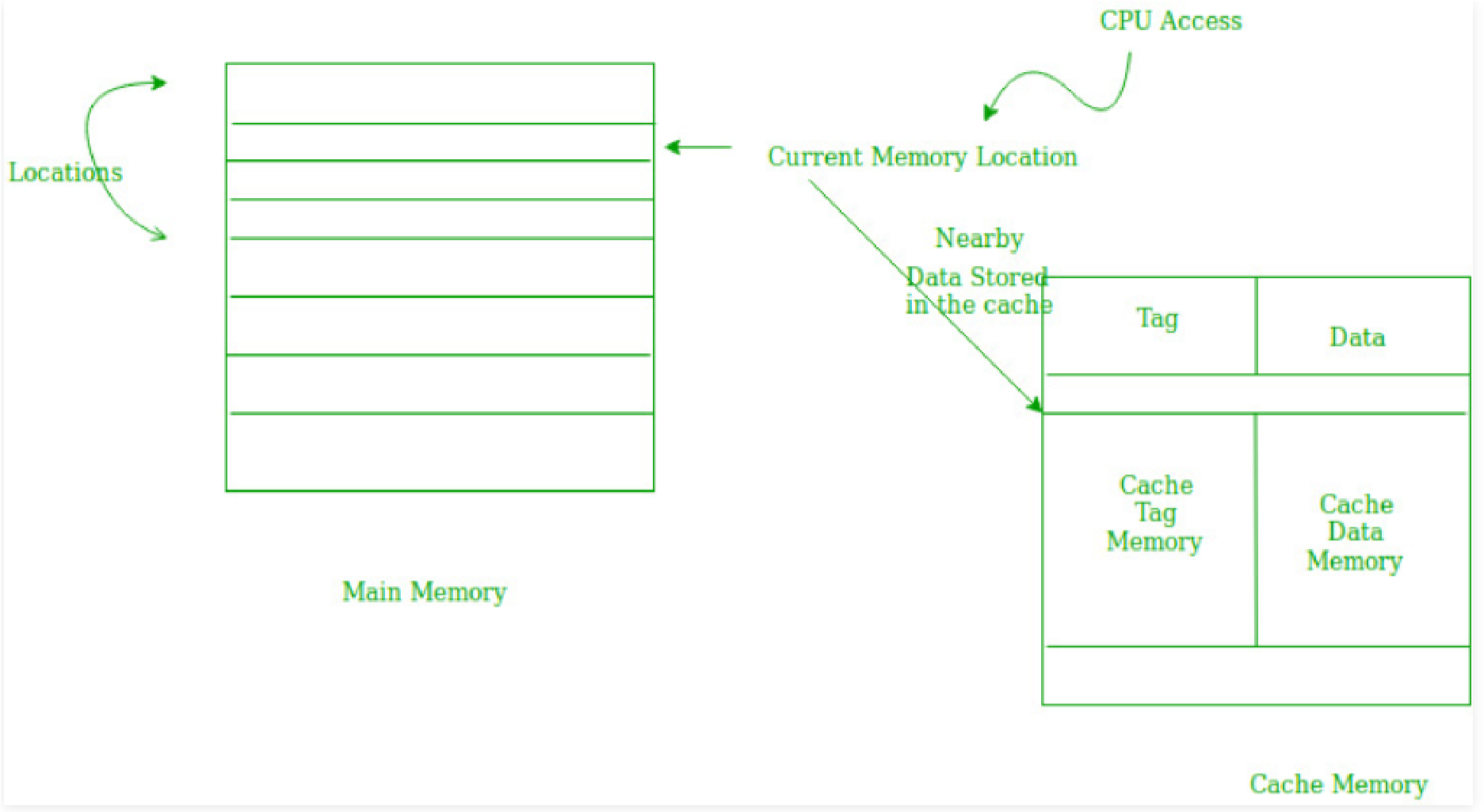


Table 15spatial locality

**Description of paging and segmentation**

### 

**Paging:**

Paging is an archiving mechanism that allows the operating system to retrieve processes from secondary storage and save them as pages in primary storage. The main memory is divided into small and medium-sized blocks of physical addresses called frames when in paging mode.

Frame sizes should match the page size to maximize primary memory usage and avoid external fragmentation. Paging is used to access data faster and this is a logical concept. (www.geeksforgeeks.org , 2019) Virtual memory, which is more than the physical memory available in a computer system, can be created. The concept of paging is used to achieve this.

The physical address space is divided into frames and the logical address space is divided into pages in paging. Both the pages and the frames are the same size. The number of pages used to measure the process. By storing the pages into the frames, these pages are now brought from the logical address space to the physical address space. To determine which page is stored in which folder, a page table is created.

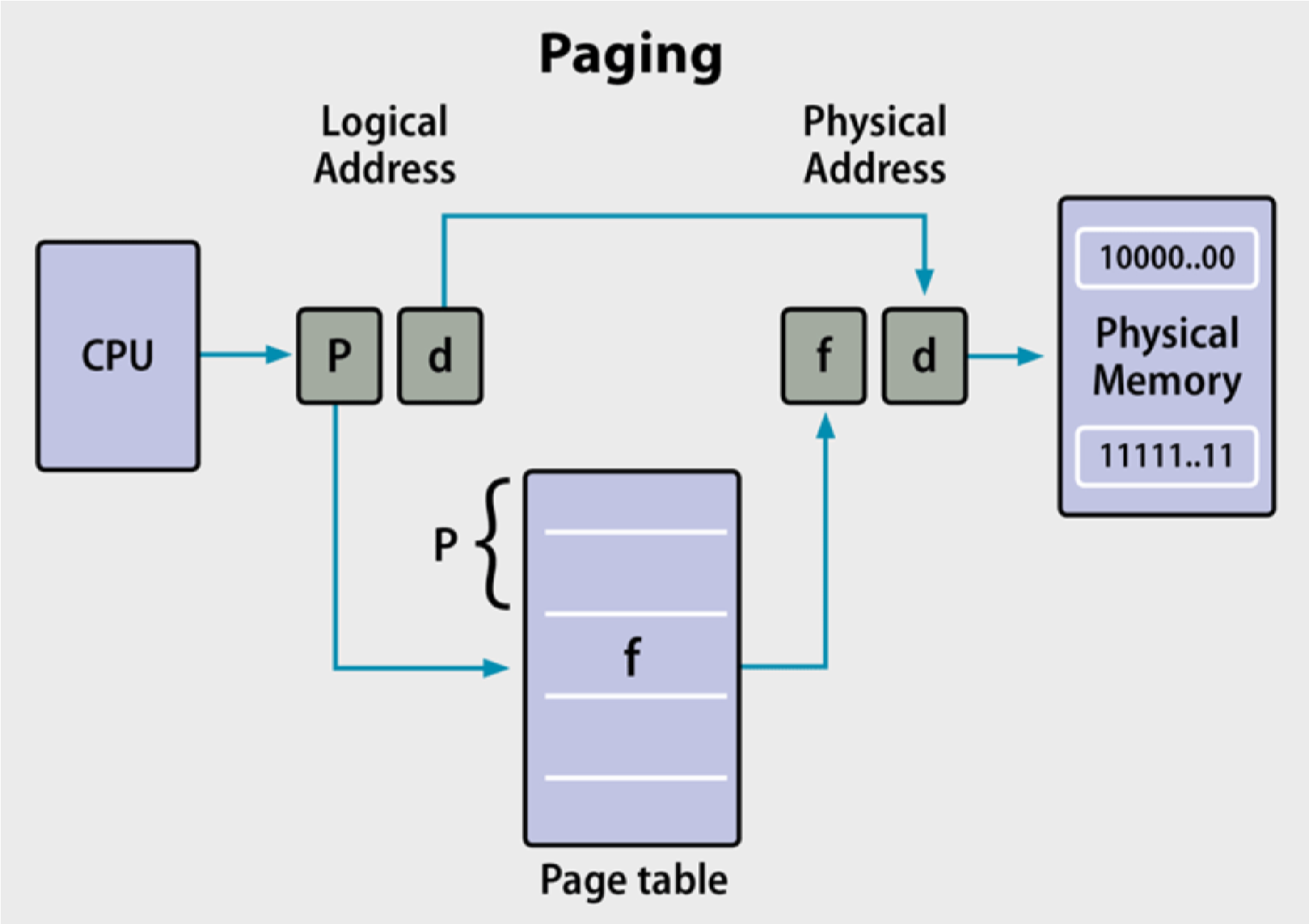


Table 16pagging

### 

### **Segmentation**

**Segmentation of data:**

This is a memory management technique that aids in the user's memory perception. A group of segments divides the logical address space. These segments have a name and a memory length associated with them. As a result, the name, base address, and length of the segment are specified. (Gagne, 2018 )

The only difference between the segmentation method and paging is that the segments have different lengths, whereas the paging method always has fixed dimensions.The main program function, data structures, auxiliary functions, and so on are all found in the program segment. The operating system keeps track of all processes in a segment table. It contains a list of free memory blocks in main or virtual memory, their sizes, the number of segments, and memory locations. (www.geeksforgeeks.org , 2019)

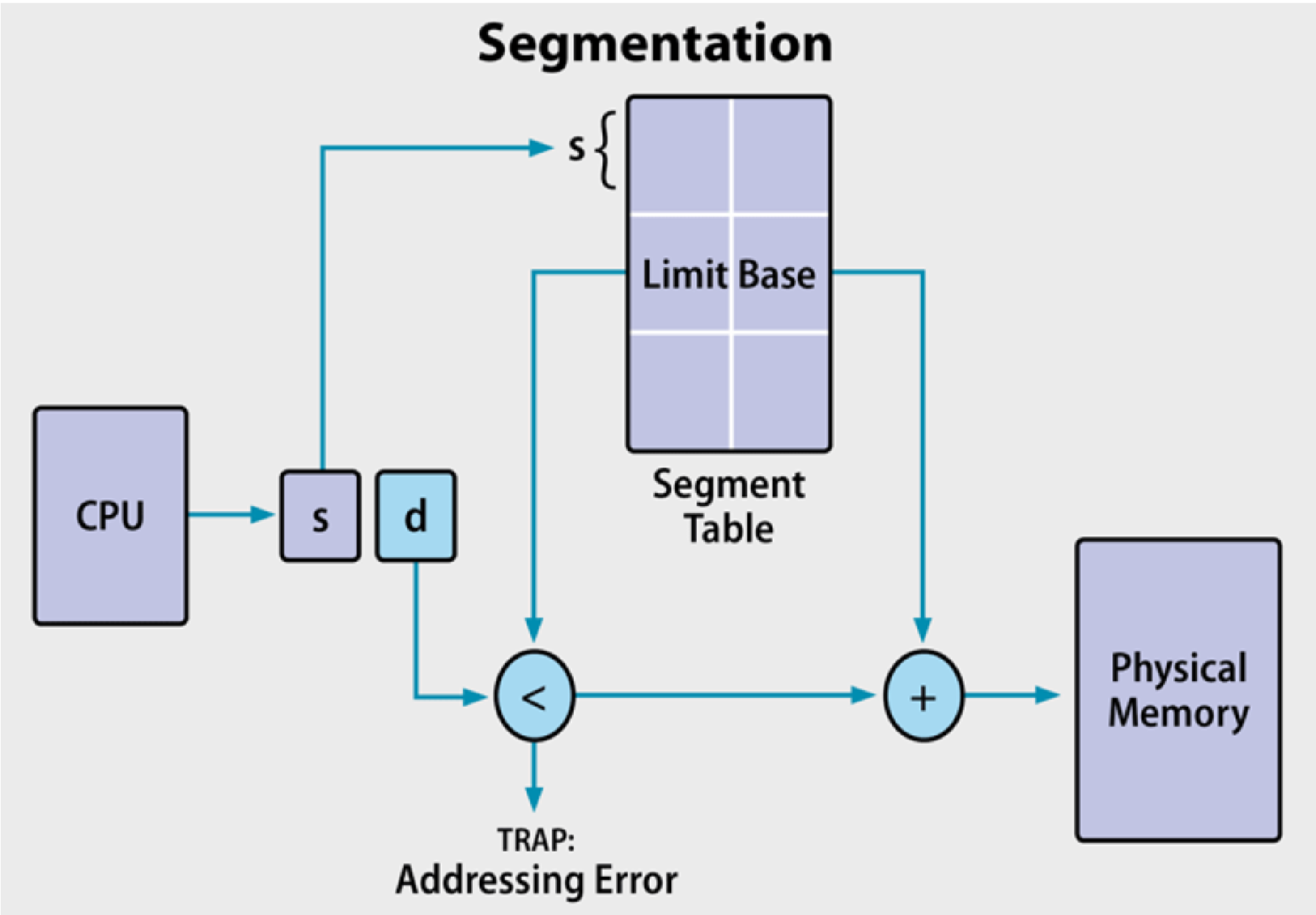


Table 17segmentation

**Conclusion**

Course B assignment was to investigate the memory management of UNIX-type operating systems, which resulted in a technical report that primarily focused on the memory hierarchy in the reference position. As a result, I've come to the conclusion that the technical aspect of this project is dependent on research. As a result, the information gathered from the previous report on the reference site and the memory hierarchy has aided in the generation of some memory management knowledge.

The difficulties I encountered while analyzing the technical meaning of the memory and reference space hierarchy were not easy to overcome. To complete the project, I conducted extensive research in various magazines and books. I've gathered a great deal of data on paging and segregation. It was extremely beneficial to me. The concept of memory management is now clear to me. In conclusion, this project motivated me to learn more about memory management in operating systems.

# References

(n.d.). Retrieved from www.techopedia.com

(n.d.). Retrieved from ecomputernotes.com/fundamental/disk-operating-system/what-is-cache-memory

(n.d.). Retrieved from stackoverflow.com ›

(2017, oct 25). Retrieved from www.desktopclass.com

(2018, 10 11). Retrieved from sap: sapbasissolutions.wordpress.com

(2018, feb 26). Retrieved from 1stopmove.wordpress.com ›

(2019, nov 14). Retrieved from vitual memory: www.windowscentral.com

(2019, aug 19). Retrieved from www.geeksforgeeks.org : www.geeksforgeeks.org

(2019, oct 17). Retrieved from network-operating-system-rtos: www.tutorialspoint.com

(2021, feb 5). Retrieved from www.easytechjunkie.com: www.easytechjunkie.com

(2021, march 9). Retrieved from technadvice.com › types-of-linux-operating-systems: https://search.yahoo.com/search;\_ylt=AwrEzesvgnJgJDEAArhXNyoA;\_ylc=X1MDMjc2NjY3OQRfcgMyBGZyA21jYWZlZQRmcjIDc2ItdG9wBGdwcmlkA0Rva0RuTkNKU0JlRVdFVWwucnRQTUEEbl9yc2x0AzAEbl9zdWdnAzEwBG9yaWdpbgNzZWFyY2gueWFob28uY29tBHBvcwMwBHBxc3RyAwRwcXN0cmwDMARxc3RybAMyMgRx

Gagne, G. a. (2018 ). Retrieved from Operating System Concepts: www.coursehero.com

Hemmendinger. (2016, feb 15). Retrieved from www.elprocus.com: www.elprocus.com

*www.cs.iit.edu ›* . (2018). Retrieved from www.cs.iit.edu ›

*www.geeksforgeeks.org*. (2020, feb 8). Retrieved from www.geeksforgeeks.org

*www.javatpoint.com ›*. (2019). Retrieved from www.javatpoint.com ›

# Appendix

### **Memory Controller**

A memory controller is an integrated circuit consisting of an integrated circuit capable of controlling DRAM. An integrated circuit is a memory controller, regardless of whether that integrated circuit includes such a memory control circuit without additional functionality or whether other integrated circuits, including integrated circuits, include other functions and / or unlimited capacity: microprocessors or processors. graphics, ASIC or other on-chip solutions.(www.easytechjunkie.com, 2021)

### **Virtual Memory**

Virtual memory is a memory allocation scheme in which secondary memory can be treated as if it were part of primary memory. The addresses that the program can use for the reference memory are distinct from the addresses used by the memory system to identify the physical memory locations and the addresses generated by the program are automatically transferred to the appropriate machine addresses.

The address scheme of the virtual system is limited by the size of the virtual archive and the amount of secondary storage available, not by the actual number of primary storage locations. (vitual memory, 2019)