**Degree Engineering**

**A Laboratory Manual for**

Operating Systems

**(3140702)**

**[ B.E. (Computer Engineering) : Semester - 4 ]**

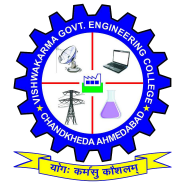
|  |  |
| --- | --- |
| Enrolment No | 220170107079 |
| Name | PATEL ABHI YOGEHSKUMAR |
| Branch | COMPUTER ENGINEERING |
| Academic Term | 2024-2025 |
| Institute Name | VISHWAKARMA GOVERNMENT ENGINEERING COLLGE |



**Directorate of Technical Education, Gandhinagar, Gujarat**

**Vishwakarma Government Engineering College,  
Chandkheda, Ahmedabad**

**Department of Computer Engineering**

****

**CERTIFICATE**

*This is to certify that Mr./Ms.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Enrollment No. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ of B.E. Semester -* ***IV*** *from* ***Computer Engineering Department*** *of this Institute (GTU Code: 017) has satisfactorily completed the Practical / Tutorial work for the subject* ***Operating System (3140702)*** *for the academic year 2024-25.*

**Place:** \_\_\_\_\_\_\_\_\_\_\_

**Date:**  \_\_\_\_\_\_\_\_\_\_\_

**Signature of Course Faculty Head of the Department**

**Preface**

Main motto of any laboratory/practical/field work is for enhancing required skills as well as creating ability amongst students to solve real time problem by developing relevant competencies in psychomotor domain. By keeping in view, GTU has designed competency focused outcome-based curriculum for engineering degree programs where sufficient weightage is given to practical work. It shows importance of enhancement of skills amongst the students and it pays attention to utilize every second of time allotted for practical amongst students, instructors and faculty members to achieve relevant outcomes by performing the experiments rather than having merely study type experiments. It is must for effective implementation of competency focused outcome-based curriculum that every practical is keenly designed to serve as a tool to develop and enhance relevant competency required by the various industry among every student. These psychomotor skills are very difficult to develop through traditional chalk and board content delivery method in the classroom. Accordingly, this lab manual is designed to focus on the industry defined relevant outcomes, rather than old practice of conducting practical to prove concept and theory.

By using this lab manual students can go through the relevant theory and procedure in advance before the actual performance which creates an interest and students can have basic idea prior to performance. This in turn enhances pre-determined outcomes amongst students. Each experiment in this manual begins with competency, industry relevant skills, course outcomes as well as practical outcomes (objectives). The students will also achieve safety and necessary precautions to be taken while performing practical.

This manual also provides guidelines to faculty members to facilitate student centric lab activities through each experiment by arranging and managing necessary resources in order that the students follow the procedures with required safety and necessary precautions to achieve the outcomes. It also gives an idea that how students will be assessed by providing rubrics.

Operating System is one of the core courses in Computer Engineering discipline. It includes basic working and application of Operating System. Process and Thread management with different Process scheduling Algorithms like FCFS, SJF, RR, Priority, etc. Concurrency control mechanisms of processes. To understand Inter Process Communication: Race Conditions, Critical Section, And Mutual Exclusion concepts are needed. Deadlock and its solutions for uninterrupted execution of processes. Memory management in Operating system. Input output and Disk scheduling algorithms. Virtualization concepts, Development of basic code of operating system using Shell Scripts.

Utmost care has been taken while preparing this lab manual however always there is chances of improvement. Therefore, we welcome constructive suggestions for improvement and removal of errors if any.

**DTE’s Vision**

* To provide globally competitive technical education
* Remove geographical imbalances and inconsistencies
* Develop student friendly resources with a special focus on girls’ education and support to

weaker sections

* Develop programs relevant to industry and create a vibrant pool of technical professionals

**Institute’s Vision**

* To create an ecosystem for proliferation of socially responsible and technically sound engineers, innovators and entrepreneurs.

**Institute’s Mission**

* To develop state-of-the-art laboratories and well-equipped academic infrastructure.
* To motivate faculty and staff for qualification up-gradation, and enhancement of subject knowledge.
* To promote research, innovation and real life problem solving skills.
* To strengthen linkages with industries, academic and research organizations.
* To reinforce concern for sustainability, natural resource conservation and social responsibility.

**Department’s Vision**

* To create an environment for providing value based education in Computer Engineering through innovation, team work and ethical practices.

**Department’s Mission**

* To produce computer engineering graduates according to the needs of industry, government, society and scientific community.
* To develop state of the art computing facilities and academic infrastructure.
* To develop partnership with industries, government agencies and R & D organizations for knowledge sharing and overall development of faculties and students.
* To solve industrial, governance and societal issues by applying computing techniques.
* To create environment for research and entrepreneurship.

**Programme Outcomes (POs)**

* 1. **Engineering knowledge:** Apply the knowledge of mathematics, science,engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
  2. **Problem analysis:** Identify, formulate, review research literature, and analyzecomplex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
  3. **Design/development of solutions:** Design solutions for complex engineeringproblems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
  4. **Conduct investigations of complex problems:** Use research-based knowledgeand research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
  5. **Modern tool usage:** Create, select, and apply appropriate techniques,resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
  6. **The engineer and society:** Apply reasoning informed by the contextualknowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
  7. **Environment and sustainability:** Understand the impact of the professionalengineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
  8. **Ethics:** Apply ethical principles and commit to professional ethics andresponsibilities and norms of the engineering practice.
  9. **Individual and team work:** Function effectively as an individual, and as amember or leader in diverse teams, and in multidisciplinary settings.
  10. **Communication:** Communicate effectively on complex engineering activitieswith the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
  11. **Project management and finance:** Demonstrate knowledge and understandingof the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
  12. **Life-long learning:** Recognize the need for, and have the preparation and abilityto engage in independent and life-long learning in the broadest context of technological change

**Program Specific Outcomes (PSOs)**

* Sound knowledge of fundamentals of computer science and engineering including software and hardware.
* Develop the software using sound software engineering principles having web based/mobile based interface.
* Use various tools and technology supporting modern software frameworks for solving problems having large volume of data in the domain of data science and machine learning.

**Program Educational Objectives (PEOs)**

* Possess technical competence in solving real life problems related to Computing.
* Acquire good analysis, design, development, implementation and testing skills to formulate simple computing solutions to the business and societal needs.
* Provide requisite skills to pursue entrepreneurship, higher studies, research and development and imbibe high degree of professionalism in the fields of computing.
* Embrace life-long learning and remain continuously employable.
* Work and excel in a highly competence supportive, multicultural and professional environment which abiding to the legal and ethical responsibilities.

**Practical – Course Outcome matrix**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Outcomes (COs)** | | | | | | | |
| **CO\_3140702.1** | | Analyze the structure of OS and basic architectural components involved in OS design | | | | | |
| **CO\_3140702.2** | | Compare and contrast various CPU scheduling algorithms. | | | | | |
| **CO\_3140702.3** | | Evaluate the requirements for the process synchronization and co-ordination in contemporary operating system. | | | | | |
| **CO\_3140702.4** | | Analyze various algorithms for memory management, I/O management and security aspects of operating system. | | | | | |
| **CO\_3140702.5** | | Write shell scripts in Unix/Linux O.S and write simple programs using kernel system calls. Also understand virtualization concept. | | | | | |
|  | | | | | | | |
| **Sr. No.** | **Practical Outcome/Title of experiment** | | **CO1** | **CO2** | **CO 3** | **CO4** | **CO5** |
| 1 | Study of Linux/Windows system Architecture , Installation and MS DOS Commands. | | **√** |  |  |  | **√** |
| 2 | Study and execute Basic and directory manipulation commands of LINUX/UNIX. | |  |  |  |  | **√** |
| 3 | Study and execute Basic File manipulation commands. | |  |  |  |  | **√** |
| 4 | Study and Execute Advance Filter Commands. | |  |  |  |  | **√** |
| 5 | Write a shell script program using Loop/ control structure. | |  |  |  |  | **√** |
| 6 | Loop/ control structure using shell script(Using while loop) | |  |  |  |  | **√** |
| 7 | Command execution via Shell script. | |  |  |  |  | **√** |
| 8 | Process Scheduling Algorithm and Comparison. | |  | **√** |  |  |  |
| 9 | Process creation and Thread Scheduling | |  |  | **√** |  |  |
| 10 | Page replacement and Disk Scheduling algorithm | |  |  |  | **√** |  |

**Industry Relevant Skills**

Operating system specialists are professionals who design, install, configure, maintain, and troubleshoot various operating systems, one need to have a solid foundation of technical skills and knowledge, as well as soft skills to work on Operating system.

**Guidelines for Faculty members**

1. Teacher should provide the guideline with demonstration of practical to the students with all features.
2. Teacher shall explain basic concepts/theory related to the experiment to the students before starting of each practical
3. Involve all the students in performance of each experiment.
4. Teacher is expected to share the skills and competencies to be developed in the students and ensure that the respective skills and competencies are developed in the students after the completion of the experimentation.
5. Teachers should give opportunity to students for hands-on experience after the demonstration.
6. Teacher may provide additional knowledge and skills to the students even though not covered in the manual but are expected from the students by concerned industry.
7. Give practical assignment and assess the performance of students based on task assigned to check whether it is as per the instructions or not.
8. Teacher is expected to refer complete curriculum of the course and follow the guidelines for implementation.

**Instructions for Students**

1. Students are expected to carefully listen to all the theory classes delivered by the faculty members and understand the COs, content of the course, teaching and examination scheme, skill set to be developed etc.
2. Students will have to perform experiments on computer system on which UNIX/Linux is installed to execute programs of Operating System.
3. Students should develop programs and execute all the programs using UNIX/Linux OS. Students have to show output of each program in their practical file.
4. Students are instructed to submit practical list as per given sample list shown on next page.
5. Student should develop a habit of submitting the experimentation work as per the schedule and she/he should be well prepared for the same.

**Common Safety Instructions**

Students are expected to

1. switch on the PC carefully (not to use wet hands)
2. shutdown the PC properly at the end of your Lab
3. carefully handle the peripherals (Mouse, Keyboard, Network cable etc)
4. use Laptop in lab after getting permission from Teacher
5. carefully handle all lab resources

**Index**

**(Progressive Assessment Sheet)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Objective(s) of Experiment** | **Pg. No.** | **Date**  **of performance** | **Date of submission** | **Assessment**  **Marks** | **Sign. of**  **Teacher with**  **date** | **Remarks** |
| 1. | Study of Linux/Windows system Architecture , Installation and MS DOS Commands. | | | | | | |
|  | * 1. Study of LINUX/UNIX Architecture AND Installation of Ubuntu using Virtual Box.   2. Give the advantages of UNIX over Windows.   3. Execute the DOS Commands |  |  |  |  |  |  |
| 2. | Study and execute Basic and directory manipulation commands of LINUX/UNIX. | | | | | | |
|  | 1. Study of Unix Shell and Environment Variables. 2. man, cal, date, echo, bc, who, uname 3. Using commands : pwd, mkdir, cd, rmdir ,ls generate given tree. |  |  |  |  |  |  |
| 3. | Study and execute Basic File manipulation commands. | | | | | | |
|  | Explore following commands: 1. cat 2.wc 3. cp 4. mv 5.rm 6. File 7. cmp 8. comm 9. diff 10. chmod 11. sort |  |  |  |  |  |  |
| 4. | Study and Execute Advance Filter Commands. | | | | | | |
|  | Explore following commands: 1.head 2. tail 3. paste 4. cut(-f) 5. cut(-c) 6. grep |  |  |  |  |  |  |
| 5. | Write a shell script program using Loop/ control structure. | | | | | | |
|  | 1. Write a shell script to find factorial of given number n. 2. Write a shell script which will generate first n Fibonacci numbers like: 1, 1, 2, 3, 5, 13, …" |  |  |  |  |  |  |
| 6. | Write a shell script program using Loop/ control structure.(Using while loop) | | | | | | |
|  | * 1. Write a shell script to read n numbers as command arguments and sort them in descending order.   2. Write a shell script to generate mark sheet of a student. Take 3 subjects, calculate and display total marks, percentage and Class obtained by the student." |  |  |  |  |  |  |
| 7. | Command execution via Shell script. | | | | | | |
|  | * 1. Write a shell script to display all executable files, directories and zero sized files from current directory   2. Write a menu driven shell script which will print the following menu and execute the given task.   MENU   * Display calendar of current month * Display today’s date and time * Display usernames those are currently logged in the system * Display your name at given x, y position * Display your terminal number   Exit |  |  |  |  |  |  |
| 8. | Process Scheduling Algorithm and Comparison. | | | | | | |
|  | Write a C Program to Implement Following CPU Scheduling algorithms.   * FCFS * Round Robin |  |  |  |  |  |  |
| 9. | Process creation and Thread Scheduling | | | | | | |
|  | * 1. Implement Producer consumer problem using thread using C/JAVA programming Language.   2. Create new thread using fork() system call using C programming Language. |  |  |  |  |  |  |
| 10. | Page replacement and Disk Scheduling algorithm | | | | | | |
|  | * 1. Implement FIFO Page replacement Algorithm using C/Java.   2. Implement C-SCAN Disk Scheduling Algorithm using C/Java. |  |  |  |  |  |  |
| **Total** | | | | |  |  |  |

**Experiment No – 1**

**AIM : S**tudy of Linux/Windows system Architecture , Installation and MS DOS Commands.

* 1. Study of LINUX/UNIX Architecture AND Installation of Ubuntu using Virtual Box.
  2. Give the advantages of UNIX over Windows.
  3. Execute the DOS Commands

**Date:** 21/02/2024

**Competency and Practical Skills:** Logic building and programming

**Relevant CO:** CO1, CO5

**Objectives:**

1. To analyze various Operating Systems structure
2. To use different Commands.
3. To differentiate working of types of OS.

**Equipment/Instruments:** Computer System with Winows/Linux

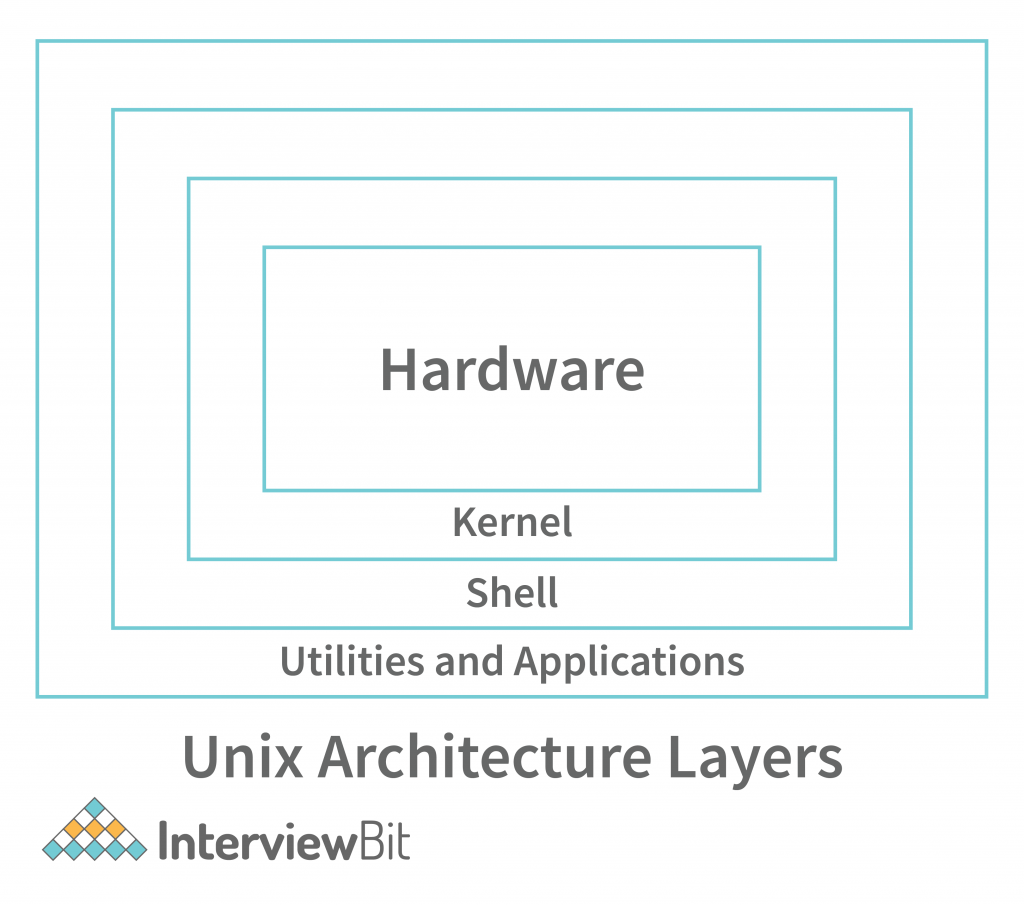
**Safety and necessary Precautions:**

* + Operate computer system carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

Computer system can be divided roughly into four components: the hardware, the operating system, the application programs, and the users).The hardware like the central processing unit (CPU), the memory, and the input/output (I/O) devices provides the basic computing resources for the system. The application programs such as word processors, spreadsheets, compilers, and Web browsers—define the ways in which these resources are used to solve users’ computing problems. The operating system controls the hardware and coordinates its use among the various application programs for the various users. In short, operating system provides the means for proper use of these resources in the operation of the computer system. It also provides an environment within which other programs can do useful work.

* 1. **Study of LINUX/UNIX Architecture AND Installation of Ubuntu using Virtual Box.**
* **Draw LINUX/UNIX Architecture Explain significance of each component.**



The Unix architecture has 4 layers. These layers are as shown below:

**Hardware:** Hardware is the most simple and least powerful layer in the Unix Architecture. Hardware is the components that are humanly visible. Whatever hardware is connected to a Unix operating system-based machine, comes in the hardware layer.

**Kernel:** This is the most powerful layer of the Unix architecture. The kernel is responsible for acting as an interface between the user and the hardware for the effective utilization of the hardware. The kernel handles the hardware effectively by using the device drivers. The kernel is also responsible for process management. So, the main 2 features of the kernel are process management and file management.

**Process Management:** The processes that execute within the operating system require a lot of management in terms of memory being allocated to them, resource allocation to the process, process synchronization, etc. All this is done by the Kernel in Unix OS. This is done using various Operating System Techniques like paging, framing, virtual memory, swapping, context-switching, etc.

**File Management:** File management involves managing the data stored in the files. This also includes the transmission of data stored in these files to the processes as and when they request it.

**Shell:** We understood the importance of the kernel and that it handles most of the important and complex tasks of Unix OS. Since the kernel is such an important program of the Unix Operating System, its direct access to the users can be dangerous. Hence, the Shell comes into the picture. Shell is an interpreter program that interprets the commands entered by the user and then sends the requests to the kernel to execute those commands. When the execution of the process is completed, the shell again sends a request to the kernel to display the program/information on the screen to the user. So, Kernel is an interface between the user and the hardware and the Shell is an interface between the user and the Kernel. The shell can be used for opening a file, writing into the files, executing programs, etc. There are 3 types of shells in the Unix Operating system.

**Bourne Shell (sh)**: It is the most widely available shell on Unix OS devices across the world. This was the first shell available in the Unix OS. It is simply called a shell.

**C Shell (csh)**: The University of California (Berkeley) developed C Shell is another Unix

shell that removes some of the obsolete features or problems from the Bourne Shell. So, it

enhances the performance of the Bourne Shell.

**Korn Shell (ksh)**: The name of the Korn Shell is based on its creator, David Korn. This

shell enhances the C shell further by removing the shortcomings of the C shell and also

enhancing the user interaction of the Bourne Shell.

**Applications/Application Programs**: The last layer of the Unix architecture is the Application Program layer. As the name suggests, this outermost layer of Unix Architecture is responsible for executing the application programs.

* **Write a step to install Ubuntu on Virtual Box / Computer System.**

**(Faculty Needs to Explain How VirtualBox/Virtual machine is helpful)**

**Steps are Given for the Installation of Ubuntu on Virtual Machine. Students need to Follow steps to Install Linux OS on the System.**

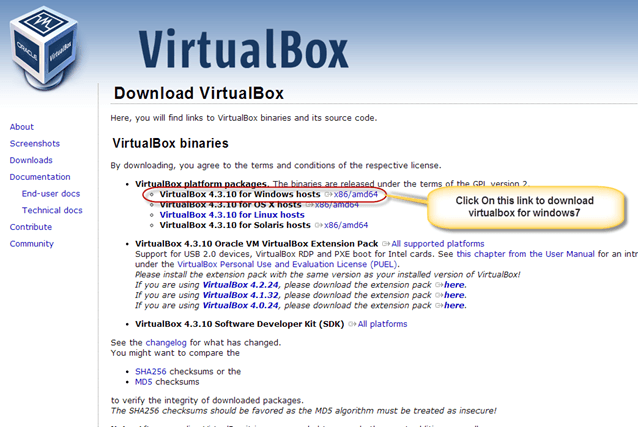
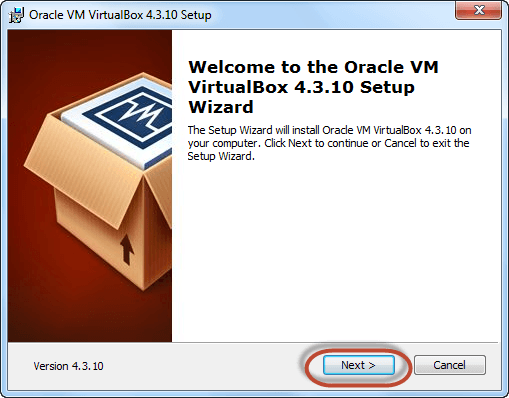
### Step-1 Installing Linux using Virtual Machine

This is a popular method to install a Linux operating system. The virtual installation offers you the freedom of running Linux on an existing OS already installed on your computer. This means if you have Windows running, then you can just run Linux with a click of a button.

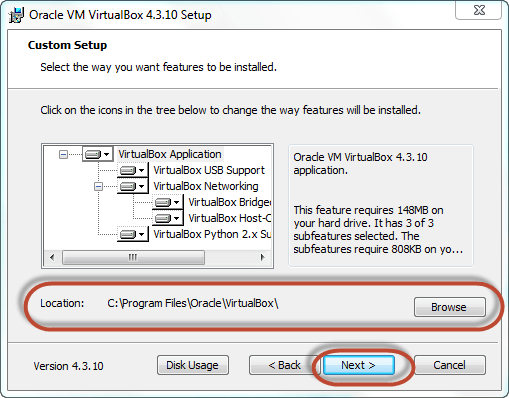
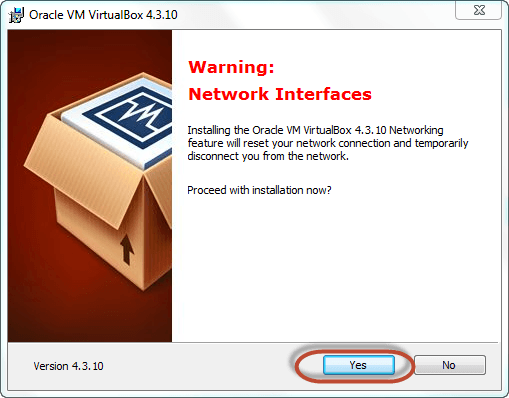
**Step-2 Download and Install Virtual Box**

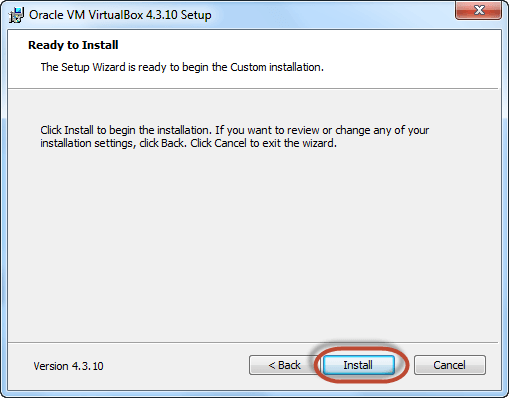
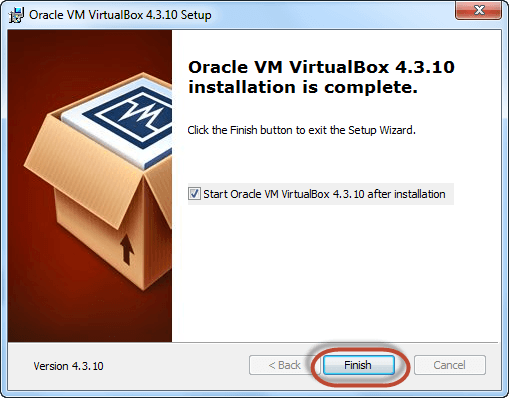
Download Virtual box depending on your processor and OS, select the appropriate package. In

our case, we have selected Windows with AMD. **Once Download Complete open the source run it and** install VirtualBox on your System. Once it is installed Open VirtualBox for further process. (snapshots are given below)

 [](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta3.png)

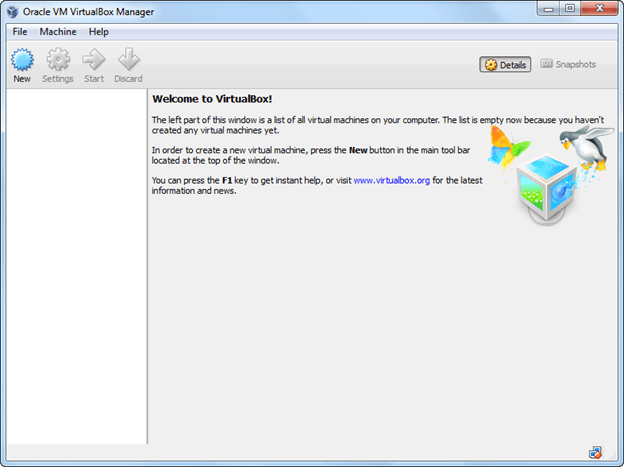
**Fig.1.1 Downloading Installing Virtual Box**

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta4.png) [](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta5.png)

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta6.png) [](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta7.png)

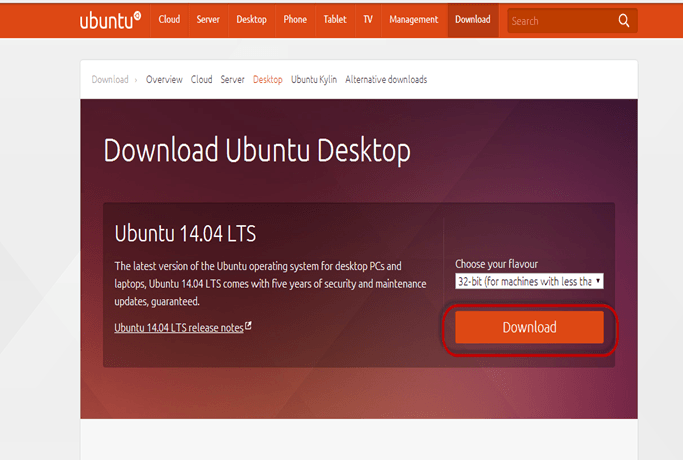
**Fig.1.2 Steps to follow while installing Virtual Box**

The virtual box dashboard looks like this-

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta8.png)

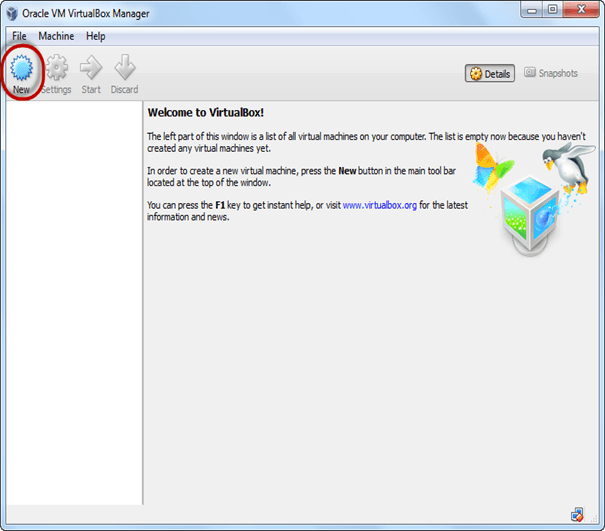
**Fig.1.3 Virtual Box dashboard after installation**

**Step-3   Download Ubuntu AND INSTALL.For download You can select 32/64-bit versions as per your choice.**

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta9.png)

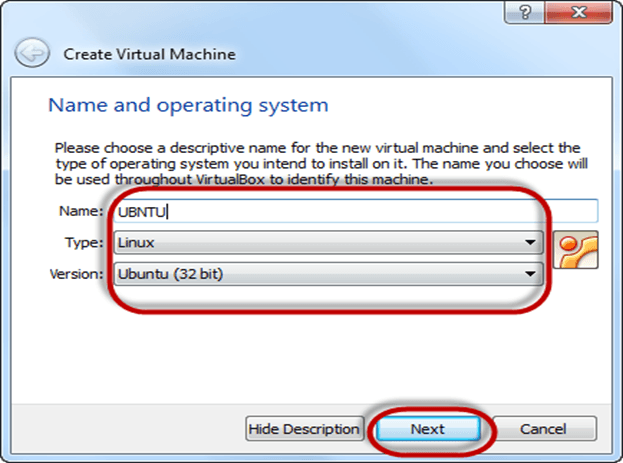
**Fig.1.4 Downloading Ubuntu**

**Step-4 Create a Machine in Virtual Box.**Open Virtual box and click on new button

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta10.png)

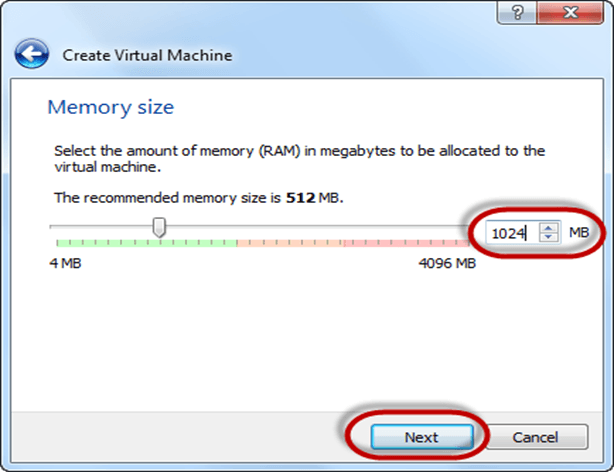
**Fig.1.5 instant starting of virtual box for ubuntu installation virtual Box**

**Step-5** In next window**,**give the name of your OS which you are installing in virtual box. And select OS like[Linux](https://www.guru99.com/unix-linux-tutorial.html)and version as Ubuntu 32 bit. And click on next

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta11.png)

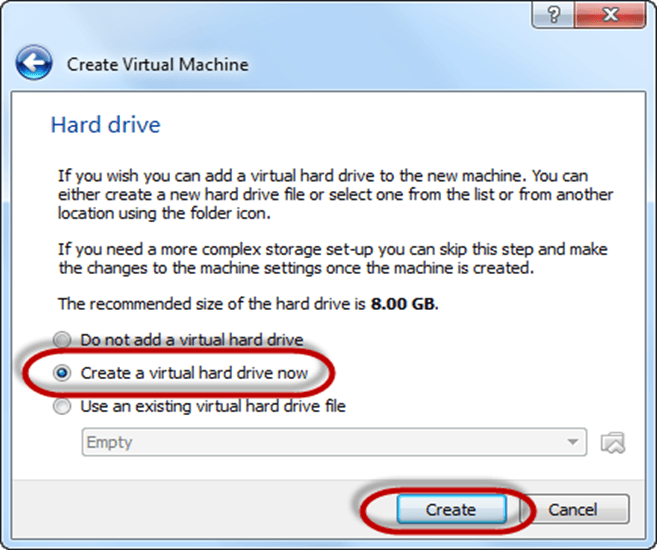
**Fig.1.6 providing basic system details for ubuntu installation**

**Step-6** Now Allocate RAM Size To your Virtual OS. It recommended keeping 1024MB (1 GB) RAM to run Ubuntu better. And click on next.

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta12.png)

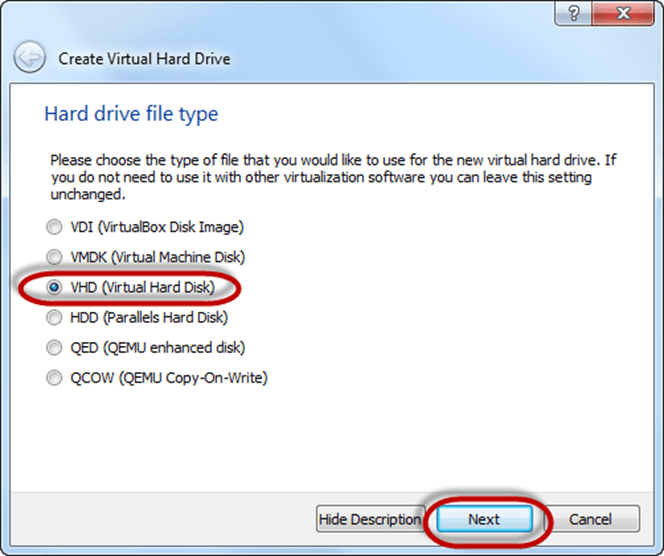
**Fig.1.7 providing memory requirement for ubuntu installation**

**Step-7** Now To run OS in virtual box we have to create virtual hard disk, click on create a virtual hard drive now and click on create button. The virtual hard disk is where the OS installation files and data/applications you create/install in this Ubuntu machine will reside

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta13.png)

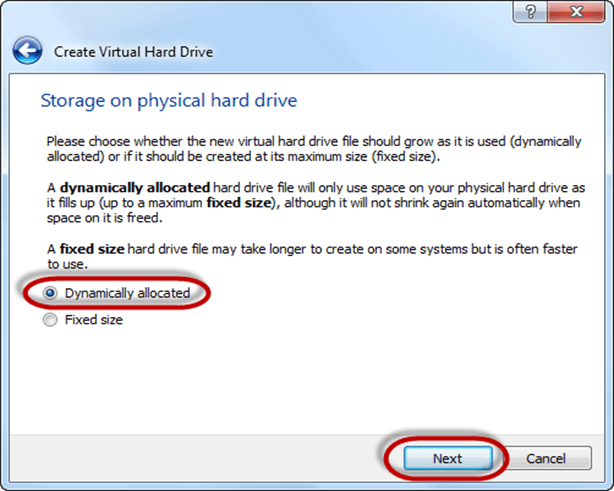
**Fig.1.8 creating virtual disk drive for ubuntu installation**

**Step-8**select VHD (virtual hard disk) option and click on next.

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta14.png)

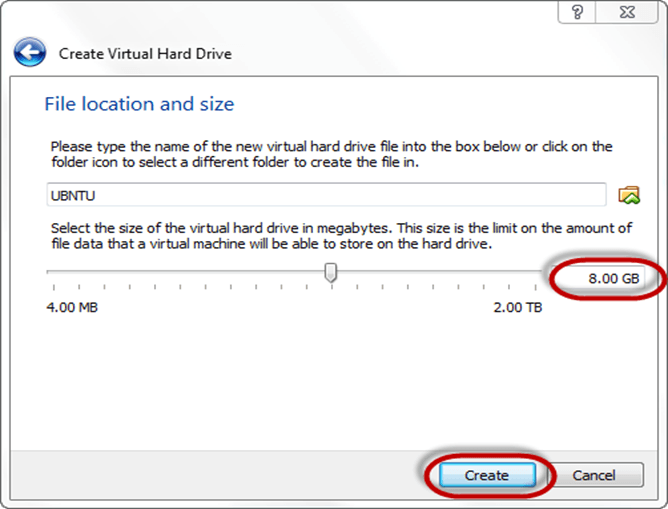
**Fig.1.9 creating virtual disk drive for ubuntu installation**

**Step-9** Click on dynamic allocated and click on next. This means that the size of the disk will increase dynamically as per requirement.

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta15.png)

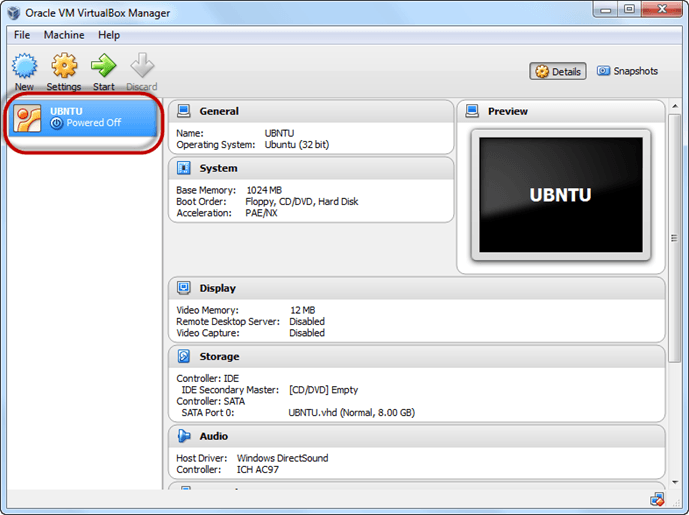
**Fig.1.10 creating virtual disk drive for ubuntu installation**

**Step-10**Allocate memory to your virtual hard drive .8GB recommended. Click on create button.

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta16.png)

**Fig.1.11 Memory allocation for ubuntu installation**

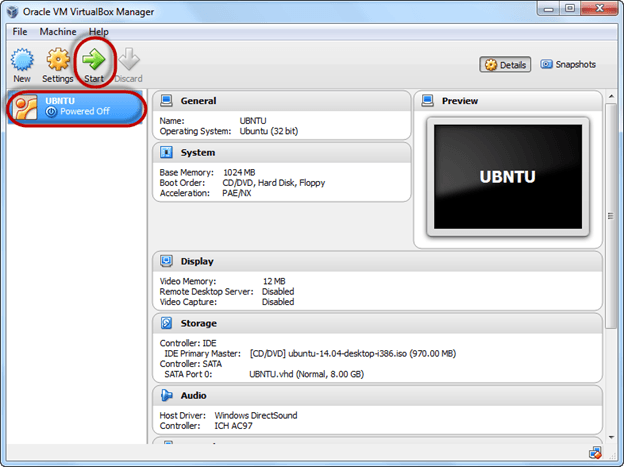
**Step-11**Now you can see the machine name in left panel

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta17.png)

**Fig.1.12 Machine configuration ready for Ubuntu installation**

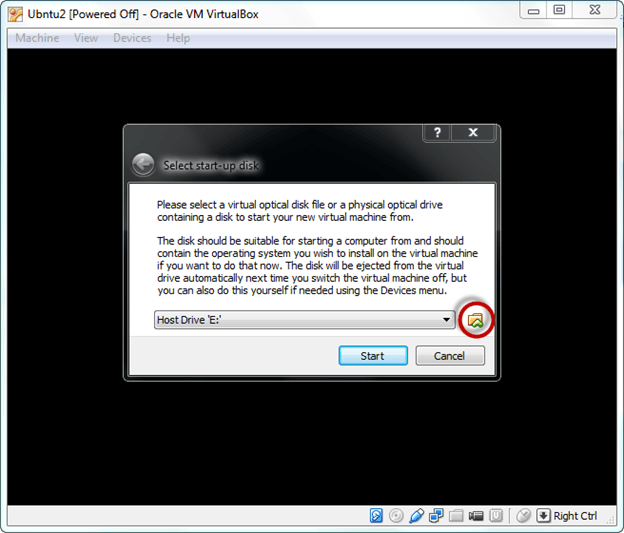
So, a Machine (PC) with 8GB hard disk, and 1GB RAM is ready.

**Step 12**  Select the Machine and Click on Start

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta18.png)

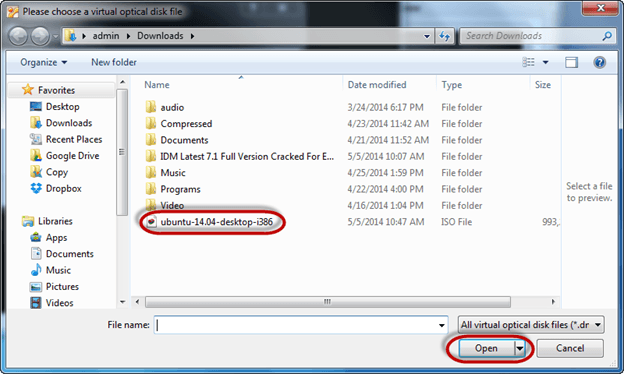
**Fig.1.13 start installing ubuntu**

**Step 13** Select the Folder Option

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta19.png)

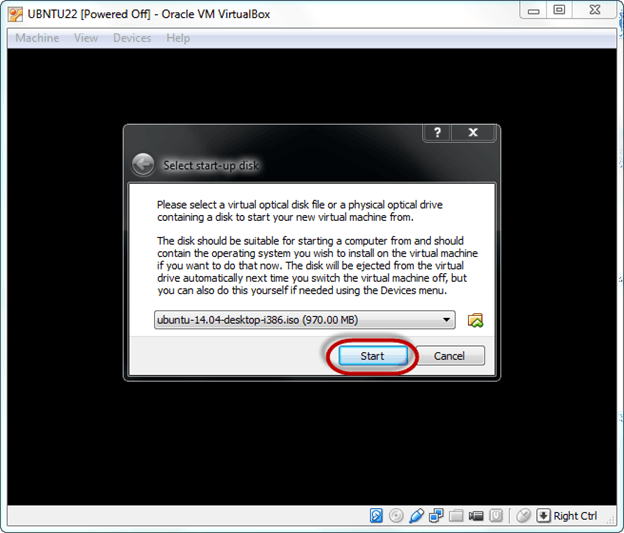
**Fig.1.14 selecting source for ubuntu installation**

**Step 14**  Select the Ubuntu iso file

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta20.png)

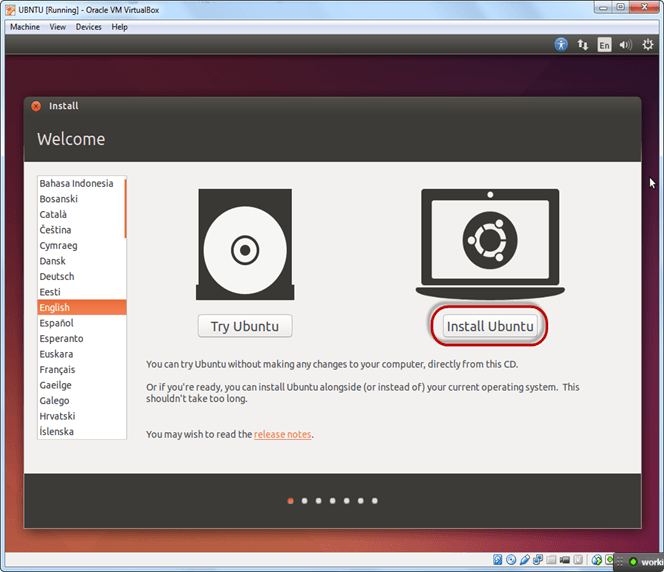
**Fig.1.15 selecting source for ubuntu installation**

**Step 15** Click Start

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta21.png)

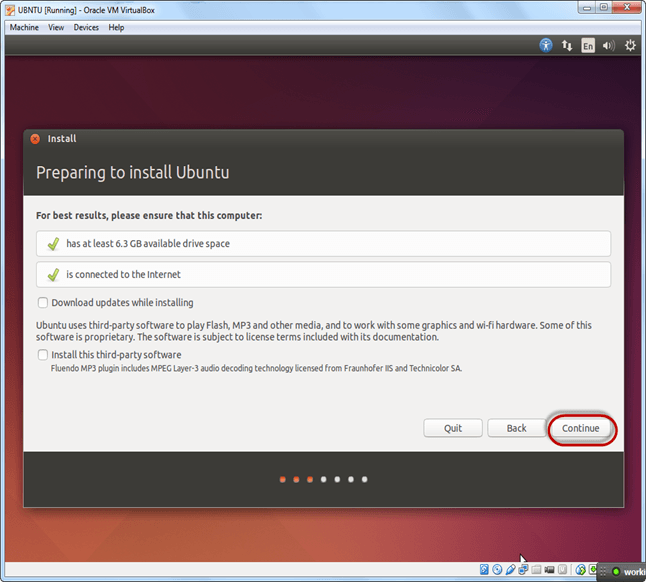
**Fig.1.16 selecting source for ubuntu installation**

**Step-16**You have an option to Run Ubuntu WITHOUT installing. In this tutorial will install Ubuntu

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta22.png)

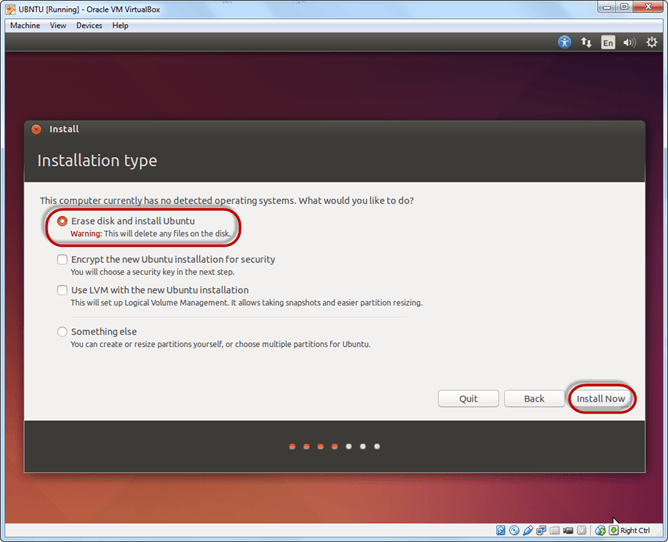
**Fig.1.17 start installation**

**Step-17** Click continue.

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta23.png)

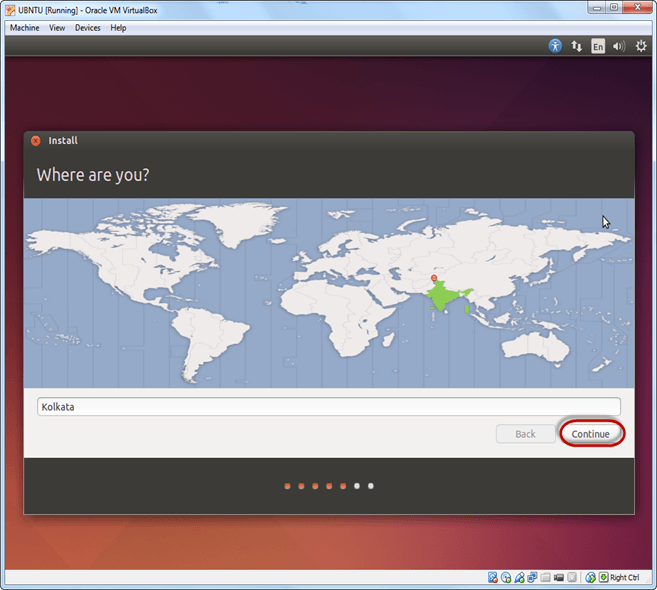
**Fig.1.18 Installation started**

**Step-18**Select option to erase the disk and install Ubuntu and click on install now. This option installs Ubuntu into our virtual hard drive which is we made earlier. It will not harm your PC or Windows installation

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta24.png)

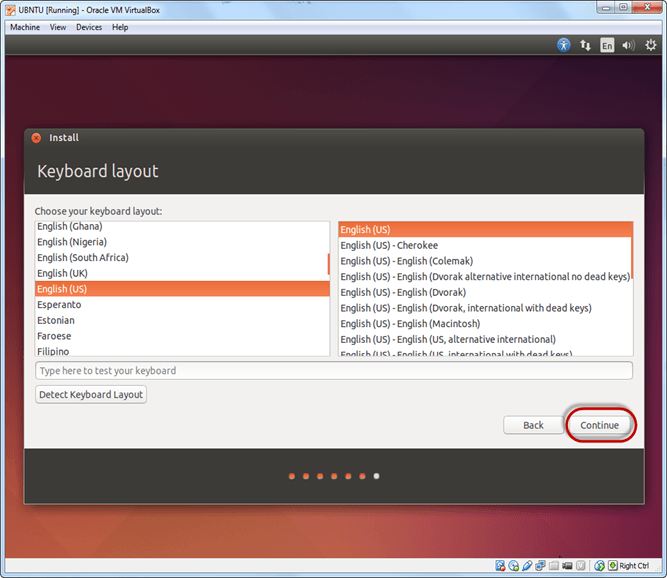
**Fig.1.19 Installation started**

**Step-19**Select your location for setting up time zone, and click on continue

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta25.png)

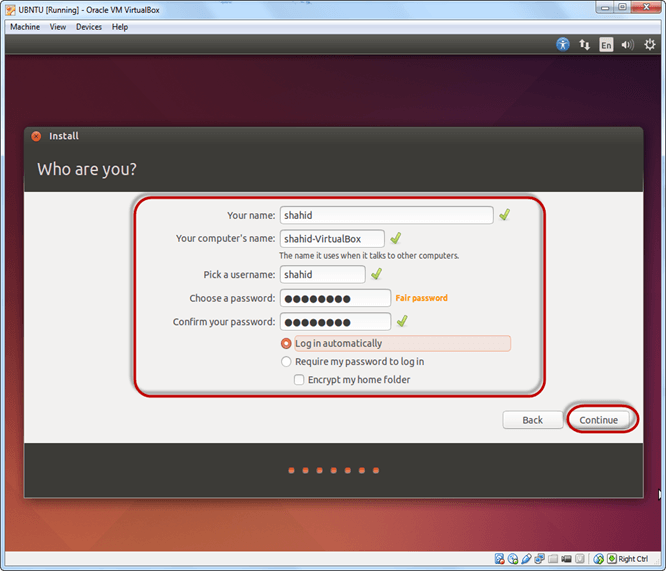
**Fig.1.20 setting up timezone**

**Step-20**Select your keyboard layout, by default English (US) is selected but if you want to change then, you can select in the list. And click on continue

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta26.png)

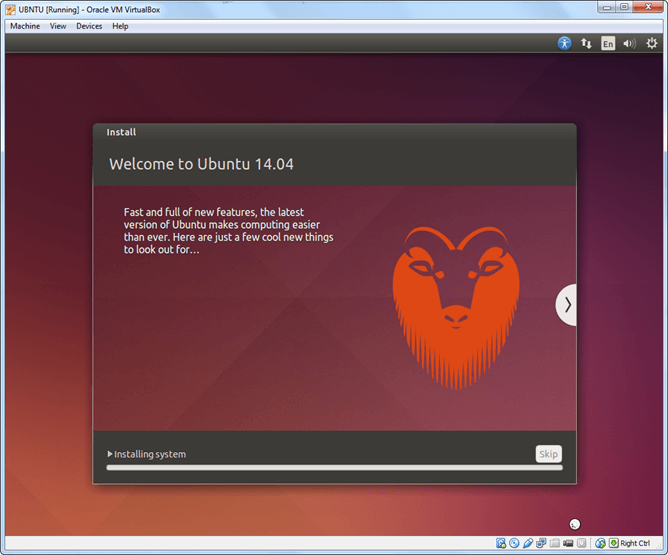
**Fig.1.21 setting up preferred Language**

**Step-21**Select your username and password for your Ubuntu admin account. This information has been needed for installing any software package into Ubuntu and also for login to your OS. Fill up your details and tick on login automatically to ignore login attempt and click on continue

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta27.png)

**Fig.1.22 setting up User credentials**

**Step-22**Installation process starts. May take up to 30 minutes. Please wait until installation process completes. After finishing the installation, you will see Ubuntu Desktop.

[](https://www.guru99.com/images/Big_Data/061114_1146_Ubuntuinsta28.png)

**Fig.1.23 Installation in Progess**

* **Faculty Member can Assign task to write Installation steps of Windows OS System.**
  1. **Give the advantage of Linux over Windows OS.**

**Advantages of Unix OS**

**Less Memory Usage:** The kernel is responsible for process management and it manages the processes in such an efficient manner that Unix programs use less memory while running.

**Safe and Secure:** The kernel is an interface between the user and the hardware and the Shell is an interface between the user and the kernel. So, there is no way a user can interact directly with the hardware or the kernel. Hence, Unix is highly secure. Also, Unix and Unix-based OS are considered safe from viruses. Hence, there is no need to install any external anti-virus software in Unix OS.

**Portable:** Unix is portable. This means that Unix can be installed on any computer or laptop. This is because it is developed using C language and C language is a high-performing and highly portable programming language.

**Multitasking and Multiuser:** We have talked about this feature of Unix OS and it is very important in today’s world where time is everything and the OS executing multiple programs saves time. However, we have discussed the concept of context-switching using which this illusion of multitasking and multiuser is created.

* 1. **Executes Following DOS Commands.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1.ATTRIB  2.CD  3.CHDIR  4.CLS  5.COPY | 1. DATE 2. DEL 3. DIR 4. ECHO 5. EXIT | 1. FC 2. FIND 3. FINDSTR 4. MKDIR 5. MOVE | 1. RENAME 2. REPLACE 3. RMDIR 4. TREE 5. SORT | 1. TITLE 2. PRINT 3. HELP 4. TIME 5. VER |

**Student has to perform above mentioned command on DOS Prompt. And also need to write each command as per given example.**

1. **ATTRIB:** - Using the **ATTRIB command**, you can change a file`s read/write attribute or set the archive attribute.

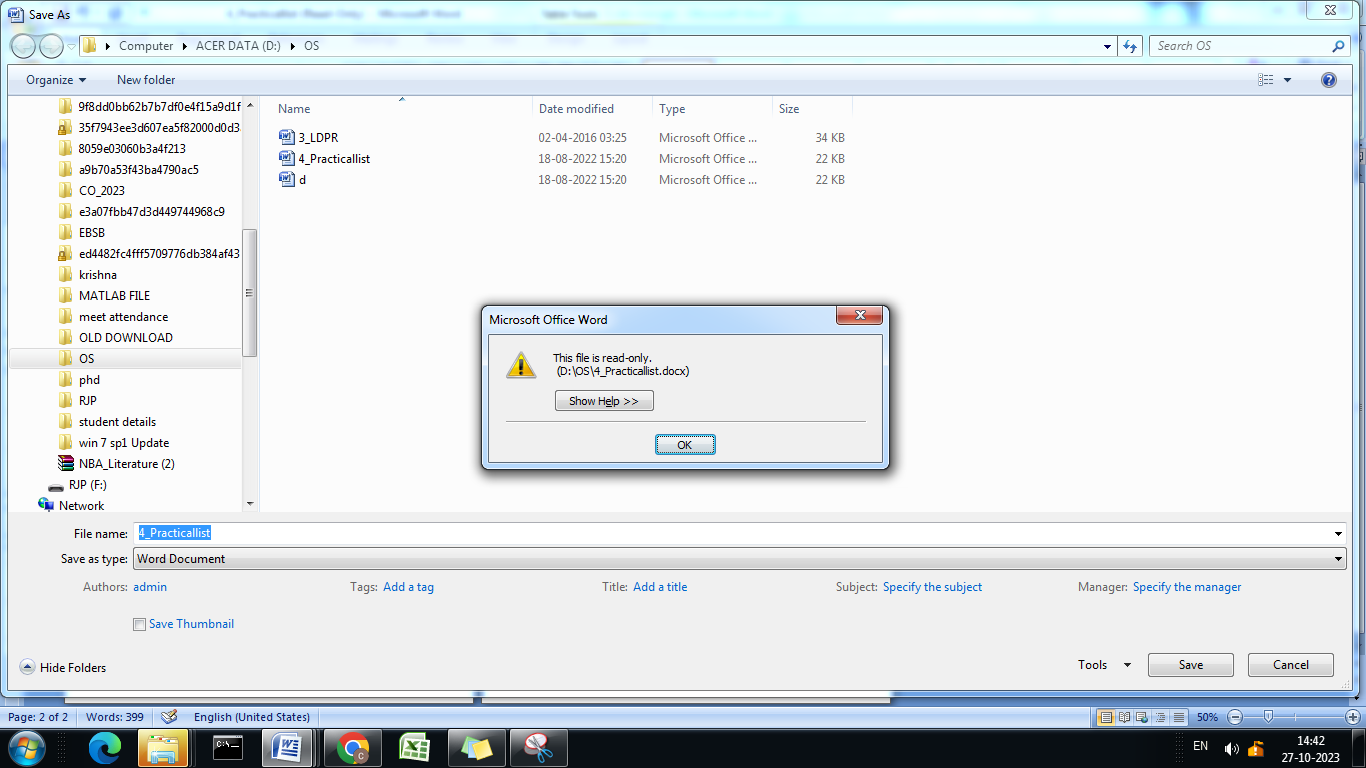
**Syntax:**  
ATTRIB [d:][path]filename [/S]  
ATTRIB [+ R|-R] [+A|-A] [+ H|-H] [+ S|-S] [d:][path]filename [/S]

+R - option to make a file read-only.  
 -R - option to change the file protection attribute back to normal (so it can be read, changed, or deleted).  
  
+A - option to set the ARCHIVE attribute of a file. When the +A option is used, this flags the file as available for archiving when using the BACKUP or XCOPY commands.  
-A - Use the -A option to turn off the ARCHIVE attribute.  
  
+H - to set the HIDDEN attribute of a file so that it will not appear in a directory listing.  
-H - Use the -H option to turn off the HIDDEN attribute.  
  
+S - With to set the SYSTEM attribute of a file. When the +S option is used, this flags the file as a command file used only by DOS. The file will not appear in a directory listing. This attribute is generally reserved for programmers.  
  
-S - Use the -S option to turn off the SYSTEM attribute.  
  
/S - Use the /S switch to set attributes on subdirectories found within the specified path.

**Execution of Command:**

**attrib.PNG**

While trying to edit File It Give message “File is Read Only”

****

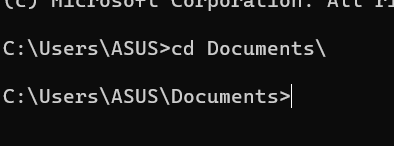
1. **CD:** - The CD command is used to change the current working directory in the command line interface.

**Syntax: CD [drive:][path]**

**Options:**

**[drive:]:** This specifies the drive letter to change to. If omitted, the command operates on the current drive.

**[path]:** This specifies the directory or path to change to. If omitted, the command switches to the current user's home directory.

****

**Examples:**

Change to a different directory within the current drive: **CD Documents**

Change to a directory in a different drive: **CD D:\Projects**

Change to the root directory of a specific drive: **CD C:\**

Change to the parent directory: **CD ..**

Change to the current user's home directory: **CD**

**Note:**

Use backslashes (\) to separate directory names within a path.

If a directory name contains spaces, enclose the entire path in double quotes (").

1. **CHDIR:** - The CHDIR command, also known as CD, is used to change the current working directory in the command-line interface.

**Syntax: CHDIR [drive:][path]**

**Options:**

**[drive:] :** Specifies the drive letter to change to. If omitted, the command operates on the current drive.

**[path] :** Specifies the directory or path to change to. If omitted, the command switches to the current user's home directory.

**Examples:**

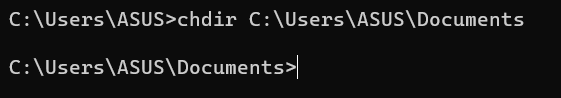
Change to a different directory within the current drive**: CHDIR Documents**

Change to a directory in a different drive**: CHDIR D:\Projects**

Change to the root directory of a specific drive**: CHDIR C:\**

Change to the parent directory**: CHDIR ..**

Change to the current user's home directory**:** CHDIR

****

1. **CLS:** - The CLS command is used to clear the contents of the command-line interface (CLI) screen in Windows operating systems.

**Syntax: CLS**

**Description:** This command clears the screen, removing all previously displayed commands and their outputs from the CLI window.

**Example: CLS**



1. **COPY:** The COPY command is used to copy one or more files from one location to another in the command-line interface (CLI) in Windows operating systems.

**Syntax: COPY source destination**

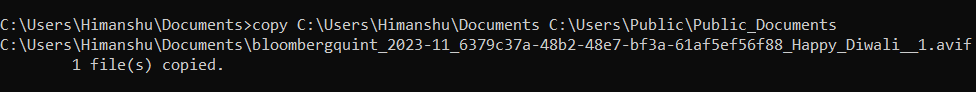
**Description:** 1. **source:** Specifies the file or files to be copied.

2. **destination:** Specifies the location where the file or files will be copied.

**Examples:**

1. Copy a single file to another location**: COPY file1.txt C:\Destination**

2. Copy multiple files to another location**: COPY file1.txt file2.txt C:\Destination**



1. **DATE:** The DATE command is used to display or set the current date in the command-line interface (CLI) in Windows operating systems.

**Syntax: DATE [new\_date]**

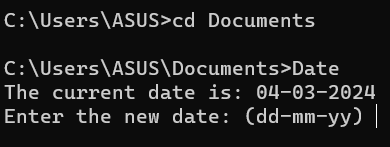
**Options: new\_date:** Specifies the new date to be set.

**Examples:**

Display the current date**: DATE**

Set the system date to a specific date: **DATE 02-20-2024**

Set the system date based on user input (prompted): **DATE**



1. **DEL: -** The DEL command is used to delete one or more files from a specified location in the command-line interface (CLI) in Windows operating systems.

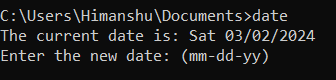
**Syntax: DEL file1 [file2 ...]**

**Description:** file1, file2, etc.: Specifies the file or files to be deleted. Wildcards (\*) can be used to specify multiple files.

**Examples:**

Delete a single file: **DEL filename.txt**

Delete multiple files using wildcards: **DEL \*.txt**



1. **DIR: -**The DIR command is used to display a list of files and subdirectories contained within a specified directory in the command-line interface (CLI) in Windows operating systems.

**Syntax: DIR [drive:][path] [options]**

**Options:**

**[drive:][path]:** Specifies the directory for which to display the contents. If omitted, the current directory is assumed.

**/S:** Displays files in the specified directory and all subdirectories.

**/B:** Uses bare format (only filenames) without any additional information.

**/W:** Uses wide list format, displaying files in multiple columns.

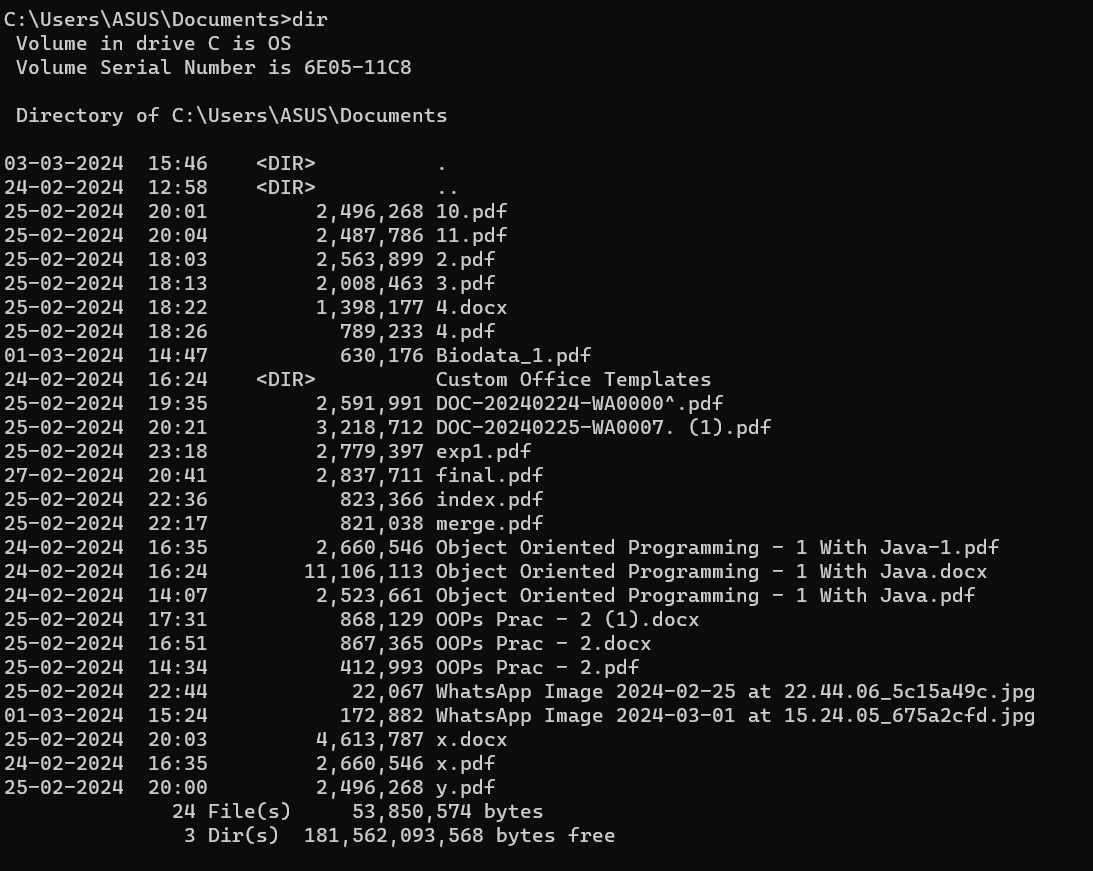
**/A:** Displays files with specified attributes (e.g., /A: H for hidden files).

**Examples:**

Display the contents of the current directory**: DIR**

Display the contents of a specific directory: **DIR C:\Users**

Display all files and subdirectories in a directory and its subdirectories**: DIR /S**



1. **ECHO: -**The ECHO command is used to display messages or enable or disable the display of command echo in the command-line interface (CLI) in Windows operating systems.

**Syntax: ECHO [message]**

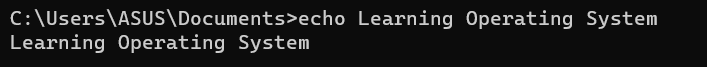
**Description:**

**message:** Specifies the message to be displayed. If omitted, the current echo state is displayed.

**Examples:**

Display a message**: ECHO Hello, World!**

Enable or disable command echo**: ECHO ON ECHO OFF**

****

1. **EXIT:** The EXIT command is used to terminate the current command-line session or batch script in Windows operating systems.

**Syntax: EXIT [/B] [exit\_code]**

**Options:**

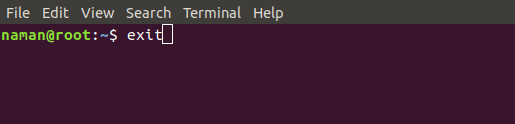
**/B:** Specifies to exit the current batch script without exiting the command prompt window.

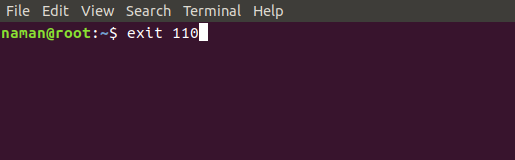
**exit\_code:** Specifies an optional numeric exit code. If provided, this exit code will be returned to the calling process.

**Examples:**

**Terminate the current command-line session: EXIT**

**Terminate a batch script with an exit code: EXIT 0**





1. **FC: -**The FC command is used to compare two files or sets of files and display the differences between them in the command-line interface (CLI) in Windows operating systems.

**Syntax:** FC [/A] [/C] [/L] [/LBn] [/N] [/OFF[LINE]] [/T] [/U] [/W] [/nnnn] [first\_file] [second\_file]

**Options:**

**- /A:** Specifies to display only the first and last lines of each set of differing lines**.**

**- /C:** Specifies to ignore case differences when comparing files.

**- /L`:** Specifies to display only the number of differing lines.

**- /LBn:** Specifies the number of lines for the buffer size (default is 100).

**- /N:** Specifies to display line numbers when comparing files.

**- /OFF[LINE]:** Specifies to ignore the line ending characters (LF, CR) when comparing files.

**- /T:** Specifies to expand tabs to spaces when comparing files**.**

**- /U:** Specifies to compare files as Unicode text files.

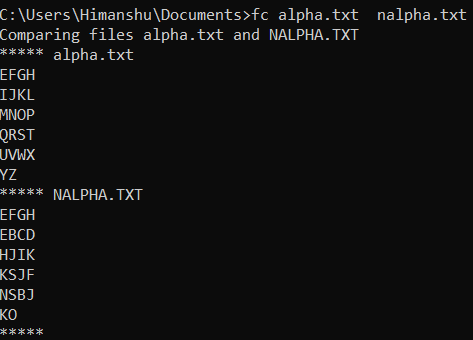
**- /W:** Specifies to compress white space (tabs and spaces) when comparing files.

**- /nnnn:** Specifies the number of lines to be compared (default is 100).

**Examples:**

Compare two files and display the differences**: FC file1.txt file2.txt**

Compare two files ignoring case differences: **FC /C file1.txt file2.txt**



1. **FIND:** The FIND command is used to search for a specified string in one or more files or input from another command in the command-line interface (CLI) in Windows operating systems.

**Syntax: FIND [/V] [/C] [/N] [/I] "string" [filename(s)]**

**Options:**

**/V:** Specifies to display all lines not containing the specified string.

**/C:** Specifies to display only a count of matching lines.

**/N:** Specifies to display line numbers with the matching lines.

**/I:** Specifies to ignore case when searching for the specified string.

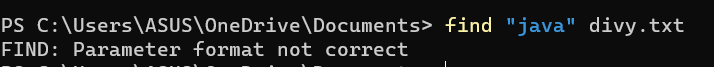
**"string":** Specifies the string to search for.

**[filename(s)]:** Specifies the file or files to search. If omitted, input is taken from the keyboard.

**Examples:**

Search for the string "keyword" in a file**: FIND "keyword" filename.txt**

Search for the string "error" in all `.log` files in a directory**: FIND "error" \*.log**

****

1. **FINDSTR:** The FINDSTR command is used to search for strings in files and outputs lines containing a match to the specified criteria in the command-line interface (CLI) in Windows operating systems.

**Syntax: FINDSTR [/B] [/E] [/L] [/R] [/S] [/C:string] [/I] [/X] [/V] [/N] [/M] [/O] [/P] [/F:file] [/C:stringlist] [/G:file] [/D:dirlist] [/A:color\_attributes] [/OFF[LINE]] [string] [[drive:][path]filename[ ...]]**

**Options:**

**/B:** Matches the pattern if it occurs at the beginning of a line.

**/E:** Matches the pattern if it occurs at the end of a line.

**/L:** Uses literal search (does not support regular expressions).

**/R:** Uses regular expressions for the search pattern.

**/S:** Searches files in the current directory and all subdirectories.

**/C:**string: Uses the specified string as a literal search string.

**/I:** Ignores case when searching for the string.

**/X:** Prints lines that match exactly.

**/V:** Prints only lines that do not contain a match.

**/N:** Prints line numbers along with the matched lines.

**/M:** Prints only the file names that contain a match.

**/O:** Prints character offset before each matching line**.**

**/P:** Skip files with non-printable characters.

**/F:file:** Reads search strings from the specified file**.**

**/C:stringlist:** Uses multiple literal search strings.

**/G:file:** Gets search strings from the specified file.

**/D:dirlist:** Searches files in the directories specified in the dirlist.

**/A:color\_attributes:** Specifies color attributes with hexadecimal color codes.

**/OFF[LINE]:** Ignores the line ending characters (LF, CR).

**Examples:**

Search for the string "keyword" in a file**: FINDSTR "keyword" filename.txt**

Search for the string "error" in all .log files in a directory: **FINDSTR "error" \*.log**



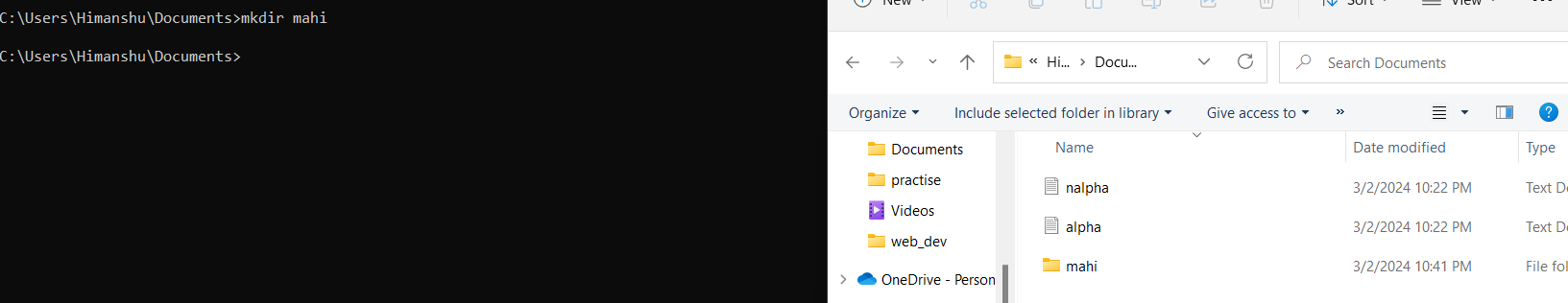
1. **MKDIR:-** The `MKDIR` command is used to create a new directory (folder) in the command-line interface (CLI) in Windows operating systems**.**

**Syntax: MKDIR [drive:][path]dirname**

**Examples:**

Create a new directory named "NewFolder" in the current directory**: MKDIR NewFolder**

Create a new directory named "Projects" in the D: drive: **MKDIR D:\Projects**



1. **MOVE: -** The MOVE command is used to move one or more files from one location to another in the command-line interface (CLI) in Windows operating systems.

**Syntax: MOVE [/Y | /-Y][drive:][path]filename1[,...] destination**

**Options:**

**/Y: Suppresses prompting to confirm you want to overwrite an existing destination file.**

**/-Y: Prompts you to confirm you want to overwrite an existing destination file.**

**Examples:**

Move a file named "file1.txt" to the "Documents" directory: **MOVE file1.txt Documents**

Move multiple files to another directory: **MOVE file1.txt file2.txt C:\Destination**



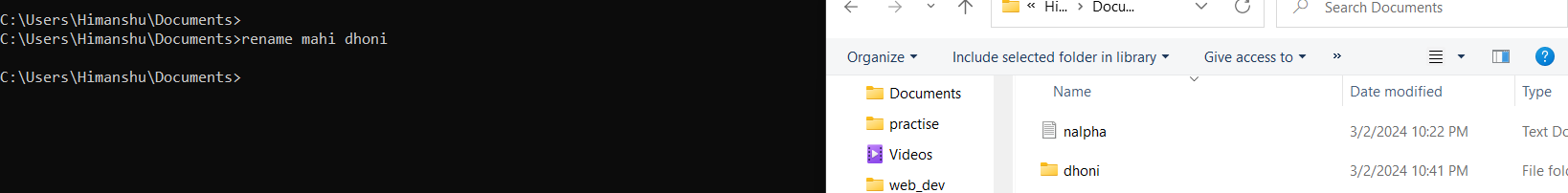
1. **RENAME:** The RENAME command is used to rename files or directories in the command-line interface (CLI) in Windows operating systems.

**Syntax: RENAME [drive:][path]filename1 filename2**

**Examples:**

Rename a file named "oldname.txt" to "newname.txt"**: RENAME oldname.txt newname.txt**

Rename a directory named "olddir" to "newdir": **RENAME olddir newdir**



1. **REPLACE:** The REPLACE command is used to replace files in one directory with files from

another directory in the command-line interface (CLI) in Windows operating systems.

**Syntax: REPLACE [drive1:][path1]filename [drive2:][path2]filename [/A]**

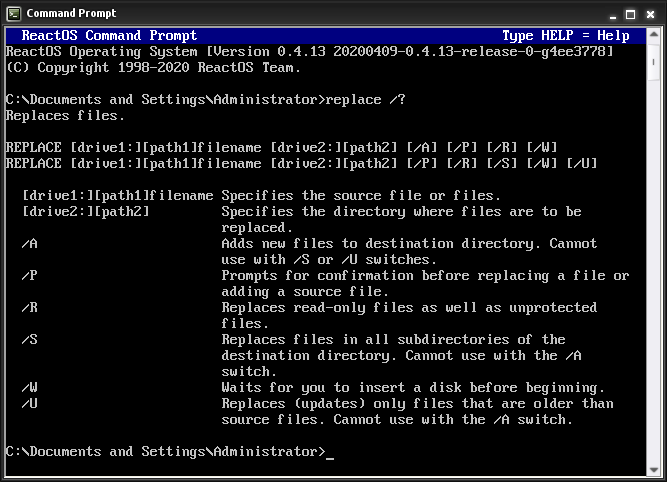
**Options:**

**/A:** Copy only files with the archive attribute set, and reset it.

**Examples:**

Replace a file in "C:\Folder1" with a file from "D:\Backup":

**REPLACE C:\Folder1\file.txt D:\Backup\file.txt**



1. **RMDIR:** The RMDIR command is used to remove (delete) directories in the command-line interface (CLI) in Windows operating systems.

**Syntax: RMDIR [/S] [/Q] [drive:][path]dirname**

**Options:**

**/S:** Removes all directories and files in the specified directory in addition to the directory itself. Used to remove a directory tree.

**/Q:** Quiet mode, do not ask if it is ok to remove a directory tree with /S.

**Examples:**

Remove an empty directory named "EmptyDir": **RMDIR EmptyDir**

Remove a directory named "Directory" and all its contents: **RMDIR /S Directory**



1. **TREE:** The TREE command is used to display the folder structure of a specified directory and its subdirectories in the command-line interface (CLI) in Windows operating systems.

**Syntax: TREE [drive:][path] [/F] [/A] [/I] [/P] [/S] [/Q] [/D]**

**Options:**

**[drive:][path]:** Specifies the directory for which to display the folder structure. If omitted, the current directory is assumed.

**/F:** Displays the names of the files in each directory in addition to the folder names.

**/A:** Uses ASCII characters instead of extended characters.

**/I:** Specifies not to include the directory paths in the output.

**/P:** Pauses after each screenful of information.

**/S:** Displays the folder structure of the specified directory and all its subdirectories.

**/Q:** Omits printing the directory structure header.

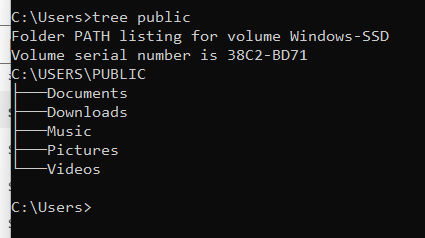
**/D:** Lists the folder structure as a directory tree instead of using ASCII characters.

**Examples:**

Display the folder structure of the current directory**: TREE**

Display the folder structure of a specific directory and its subdirectories:

**TREE C:\Projects /F /S**



1. **SORT:** The SORT command is used to sort the contents of a file or the output of a command alphabetically or numerically in the command-line interface (CLI) in Windows operating systems.

**Syntax: SORT [/R] [/+n] [/M] [/L] [/REC:delimiter]**

**Options:**

**/R:** Sorts the output in reverse order.

**/+n:** Specifies the character position from which to start each line comparison (where `n` is a positive integer).

**/M:** Specifies that the sort should be performed in memory. This is required if the input file size is larger than the available memory.

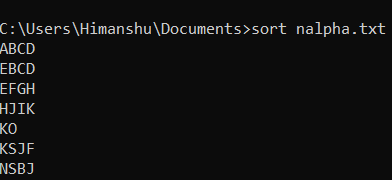
**/L:** Compares the sort strings with case sensitivity.

**/REC:delimiter:** Specifies the character that separates fields in each record for sorting.

**Examples:**

Sort the lines in a file alphabetically**: SORT filename.txt**

Sort the lines in a file numerically**: SORT /+3 filename.txt**



1. **TITLE:** The TITLE command is used to set the window title for the command prompt window in Windows operating systems.

**Syntax: TITLE [string]**

**Examples:**

**Set the window title to "Command Prompt": TITLE Command Prompt**



1. **PRINT**: The `PRINT` command is used to print a text file to a specified printer in the command-line interface (CLI) in Windows operating systems**.**

**Syntax: PRINT [/D:device] [drive:][path]filename**

**Options:**

**/D:device:** Specifies the printer device to which the file should be printed.

**Examples:**

Print a text file named "document.txt" to the default printer: **PRINT document.txt**



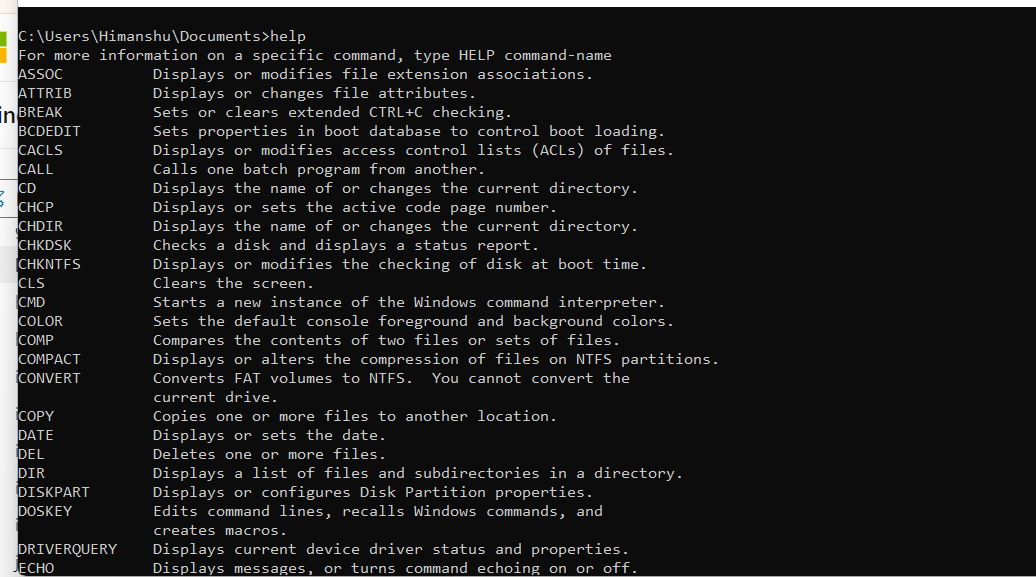
1. **HELP:** The HELP command is used to display information about the commands available in the command-line interface (CLI) in Windows operating systems.

**Syntax: HELP [command]**

**Examples:**

Display general help information: **HELP**

Display help information for a specific command, such as DIR: **HELP DIR**

****

1. **TIME:** The TIME command is used to display or set the system time in the command-line interface (CLI) in Windows operating systems.

**Syntax: TIME [new\_time]**

**Options:**

**new\_time:** Specifies the new system time to be set.

**Examples:**

Display the current system time: **TIME**

Set the system time to 9:30 AM**: TIME 09:30**

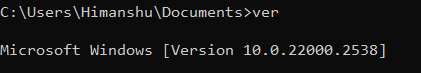


1. **VER:** The VER command is used to display the version number of the Windows operating system in the command-line interface (CLI).

**Syntax: VER**

**Examples:**

Display the version number of the Windows operating system: **VER**



**Observations:**

We studied Linux/Windows system architecture and MS-DOS commands. We observed differences in file system structure: Linux has a hierarchical system, while Windows is drive-based. Installation methods varied, with Linux needing manual partitioning and Windows having a graphical setup. We also learned MS-DOS commands like CD, DIR, and RENAME for file/directory management in Windows CLI.

**Conclusion:**

Linux and Windows OS architectures, installation methods, and command-line operations. Linux provides more flexibility during installation, catering to advanced users, while Windows prioritizes ease of use for everyday users. Exploring MS-DOS commands underscored the significance of command-line skills in Windows file management. Overall, the experiment deepened our understanding of system architecture and essential operations in both environments.

**Quiz:**

1. **Identify types of OS based on its working.**

**Ans.** Operating systems can be categorized into several types based on their working, including:

- **Real-time Operating Systems (RTOS):** These OS prioritize real-time processing and respond to input within a specified time frame, commonly used in embedded systems and critical applications like aerospace and industrial automation.

- **Single-user, Single-tasking OS:** Designed to support one user and one task at a time, such as MS-DOS.

- **Single-user, Multi-tasking OS:** Allows a single user to perform multiple tasks concurrently, like Microsoft Windows and macOS.

- **Multi-user OS**: Supports multiple users accessing the system simultaneously, often found in server environments running Linux or Unix.

- Distributed OS: Manages a group of independent computers and makes them appear as a single computer, enabling resource sharing and load balancing.

- **Embedded OS**: Designed to run on embedded systems with limited resources, commonly used in devices like smartphones, IoT devices, and digital cameras.

1. **Write use of Shell in Linux.**

**Ans.** The shell in Linux serves as the command-line interpreter that provides a user interface for interacting with the operating system. Its primary uses include:

- **Executing commands**: Users can enter commands directly into the shell to perform various tasks, such as navigating the file system, managing files, and executing programs.

- **Scripting**: Shell scripting allows users to automate tasks by writing scripts that contain a series of shell commands. These scripts can be executed to perform repetitive tasks efficiently.

- **Redirection and piping**: The shell supports input/output redirection and piping, enabling users to manipulate and process data streams from commands or files.

1. **Give the benefits of using Virtualbox?**

**Ans. Platform independence:** VirtualBox is available for various host operating systems, including Windows, macOS, Linux, and Solaris, allowing users to run virtual machines (VMs) on different platforms.

**- Cost-effective:** VirtualBox is open-source software and free to use, making it a cost-effective solution for individuals and organizations looking to implement virtualization.

**- Resource management:** VirtualBox efficiently manages hardware resources and allows users to allocate specific amounts of CPU, memory, and disk space to each virtual machine, optimizing performance and resource utilization.

**- Snapshot and cloning:** VirtualBox supports snapshotting, enabling users to capture the current state of a VM and revert to it later if needed. It also allows for cloning VMs, making it easy to replicate configurations and environments.

**- Testing and development:** VirtualBox is widely used for testing and development purposes, providing a sandboxed environment for experimenting with different operating systems, applications, and configurations without impacting the host system**.**

**Suggested Reference:**

1. Operating Systems: Internals & Design Principles, 9th Edition, William Stallings, Pearson Education India
2. Operating System Concepts, 9th edition Peter B. Galvin, Greg Gagne, Abraham Silberschatz, John Wiley & Sons, Inc.
3. Modern Operating Systems-By Andrew S. Tanenbaum (PHI)
4. UNIX : Concepts and Applications | 4th Edition by Sumitabha Das ,McGraw Hill
5. https://ubuntu.com/tutorials/how-to-run-ubuntu-desktop-on-a-virtual-machine-using-virtualbox#1-overview

# References used by the students:

# https://www.interviewbit.com/blog/unix-architecture/

# https://ubuntu.com/tutorials/how-to-run-ubuntu-desktop-on-a-virtual-machine-using-virtualbox

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding Related to OS/Computer System (4) | | | Command Execution(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

**Experiment No: 2**

**AIM : Study and execute Basic and directory manipulation commands of LINUX/UNIX.**

* 1. Study of Unix Shell and Environment Variables.
  2. man, cal, date, echo, bc, who, uname
  3. Using commands: pwd, mkdir, cd, rmdir ,ls generate given tree.

**Date:**

**Competency and Practical Skills:** Basic Skills to work with Computer Systems/ Linux Terminal

**Relevant CO:** , CO5

**Objectives:**

1. To understand the importance of Shell/Envirnment Variable
2. To Work with basic Commands.
3. To work/Access the Directory commands.

**Equipment/Instruments:** Computer System with Linux OS.

**Safety and necessary Precautions:**

* + Operate computer system carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

A Unix shell is a command-line interpreter or shell that provides a command line user interface for Unix-like operating systems. The shell is both an interactive command language and a scripting language, and is used by the operating system to control the execution of the system using shell scripts.

Environment variables basically define the behavior of the environment. They can affect the running processes or programs executed in the environment. Every Linux process has an associated set of environment environment variables are typically accessed through the shell. The shell is a command-line interface that interprets and executes commands entered by the user. It provides a way to set, modify, and retrieve environment variables.just like programming

language variables The scope of any variable is the region from which it can be accessed or over which it is defined. An environment variable in Linux can have global or local scope.

There are Large set of Commands supported by Linux/Unix. In this Practical we will study Basic System Command and Directory Commands.

**Basic Linux and Directory commands**

|  |  |
| --- | --- |
| **Command Name** | **Description** |
| man | It is used to open help manual for any linux command |
| cal | It displays the current month calendar (System). |
| date | It displays current date (System). |
| echo | It prints the string as provided by user. |
| bc | It is a command line calculator used to do basic mathematical calculations. |
| who | It prints information about users who are currently logged in. |
| uname | It displays the information about the system. |
| pwd | It displays name of present working directory. |
| mkdir | It creates new directory. |
| cd | It is used to move from one directory to another. |
| rmdir | It is used remove the directory. |
| ls | Listing the files and directory. |

**For any command, help manual is available in Linux System. We can use man Command to open help manual for any command using following command:**

**$man command name**

* **Based on Questions asked, student has to execute command and need to write the answer.**

**Before using the command to solve the questions, student has to write name of the**

**command, Syntax, options available to use the command and description(as per requirement).**

**Example:**

**Command Name: date (**print or set the system date and time**)**

**Syntax:** date [OPTION]... [+FORMAT]

date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]

**OPTIONS**

%a locale's abbreviated weekday name (e.g., Sun)

%A locale's full weekday name (e.g., Sunday)

%b locale's abbreviated month name (e.g., Jan)

%B locale's full month name (e.g., January)

%c locale's date and time (e.g., Thu Mar 3 23:05:25 2005)

%C century; like %Y, except omit last two digits (e.g., 20)

%d day of month (e.g., 01)

%D date; same as %m/%d/%y

%e day of month, space padded; same as %\_d

%F full date; same as %Y-%m-%d

%g last two digits of year of ISO week number (see %G)

%G year of ISO week number (see %V); normally useful only with %V

%h same as %b

%H hour (00..23)

**Note: There are number of option available to work with specific command student can write description of any option as per requirement of practical.**

* **To Write the Answer of Question Example is given below.**

**Example.**

**Command Name:** Shell

**Syntax:** $Shell

**Output:**

**2.png**

* **Cal:** The cal command in Linux is used to display a calendar for a specific month or year. It provides a textual representation of the calendar, including days of the week and dates.

**Syntax:** **cal [month] [year]**

cal command in linux

**Options:**

**[month]:** Specifies the month for which the calendar should be displayed (1-12). If omitted,

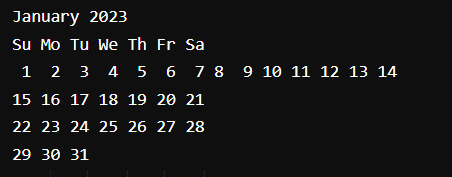
the current month is assumed.

**[year]:** Specifies the year for which the calendar should be displayed. If omitted, the current year is assumed

**Example:**

cal

cal 2 2024



* **Date:** The date command in Linux is used to print or set the system date and time. It allows users to display the current date and time in various formats or set the system date and time to a specified value.

**Syntax:** **date [OPTION]... [+FORMAT]**

**Options:**

**[OPTION]:** Specifies various options for formatting the output or setting the system date and time.

**[+FORMAT]:** Specifies the format for displaying the date and time.

Display Current Date and Time in Linux

**Example:**

date

date "+%A, %B %d, %Y"

* **echo command**

[**echo command**](https://www.geeksforgeeks.org/echo-command-in-linux-with-examples/) in Linux is specially used to print something in the terminal

**Syntax:-**

echo <message>  
  
**Command:**

echo command in linux

**Output:**

output of echo command in linux

* **bc command**

**bc** command is used for command line calculator. It is similar to basic calculator by using which we can do basic mathematical calculations.   
Arithmetic operations are the most basic in any kind of programming language. Linux or Unix operating system provides the **bc command** and **expr command** for doing arithmetic calculations. You can use these commands in bash or shell script also for evaluating arithmetic expressions.   
  
**Syntax:**

bc [ -hlwsqv ] [long-options] [ file ... ]

**Options:**   
**-h**, {- -help } : Print the usage and exit   
**-i**, {- -interactive } : Force interactive mode   
**-l**, {- -mathlib } : Define the standard math library   
**-w**, {- -warn } : Give warnings for extensions to POSIX bc   
**-s**, {- -standard } : Process exactly the POSIX bc language   
**-q**, {- -quiet } : Do not print the normal GNU bc welcome   
**-v**, {- -version } : Print the vers



* **who command**

The **who** command in Linux is used to display information about users who are currently logged in to the system. It provides details such as the username, terminal, login time, and originating IP address or hostname.

Here's the basic syntax of the **who** command:

bash

who [options]

Commonly used options include:

* **-q or --count:** Display only the count of logged-in users.
* **-b or --boot:** Show the time of the last system boot.
* **-d or --dead:** Display information about dead processes (those that have exited).

Without any options, the **who** command typically displays the following information:

* Username
* Terminal
* Date and time of login
* Remote hostname or IP address

Example:

who

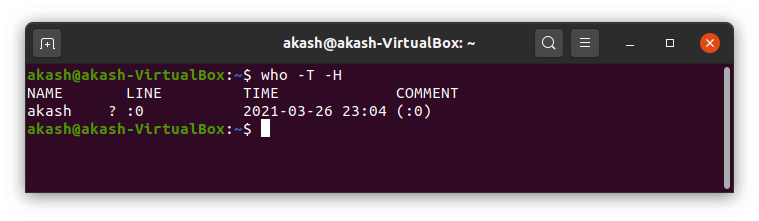
Output might look like:

yaml

username1 tty1 2022-01-01 08:00 username2 pts/0 2022-01-01 09:30 (192.168.1.2)

This indicates that "username1" is logged in on the local terminal (tty1) since 8:00 AM, and "username2" is logged in on a pseudo-terminal (pts/0) since 9:30 AM from the IP address 192.168.1.2.

The **who** command is helpful for system administrators to monitor user activity and identify who is currently using the system.



* **Uname command**

The **[uname command](https://www.geeksforgeeks.org/uname-command-in-linux-with-examples/)** is used to check the complete OS information of the system. Check out the command and the output below

**Syntax:-**

uname

**Command:**

Lightbox

**Output:**

output of uname command in linux

* **pwd command**

The **[pwd command](https://www.geeksforgeeks.org/pwd-command-in-linux-with-examples/)** is mostly used to print the current working directory on your terminal. It is also one of the most commonly used commands.

Now, your terminal prompt should usually include the entire directory. If it doesn’t, this is a quick command to see which directory you’re in. Another purpose for this command is when creating scripts because it can help us find the directory in which the script was saved. The below pictures are the output with the command.

**Syntax:-**

pwd  
  
**Command:**

pwd command in linux

**Output:**

output of pwd command in linux

* **mkdir command**

This [mkdir command](https://www.geeksforgeeks.org/mkdir-command-in-linux-with-examples/) allows you to create fresh directories in the terminal itself. The default syntax is **mkdir <directory name>** and the new directory will be created.

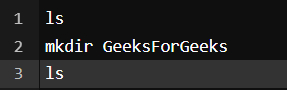
For example, if you want to create a directory as **“GeeksforGeeks”** then the basic syntax would be:

mkdir GeeksforGeeks

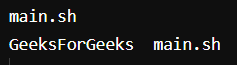
In case you want to create another directory inside the main directory GeeksforGeeks to store projects, you can use the following command to do so. **mkdir GeeksforGeeks/projects**

**Syntax:-**

mkdir <directory name>  
  
**Command:**



**Output:**



You can see we used ls first to see the directories present there and then **mkdir**to create another directory followed by **ls**to view the created directories.

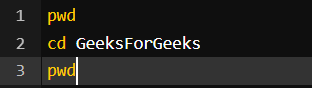
* **cd command**

The [**cd command**](https://www.geeksforgeeks.org/cd-command-in-linux-with-examples/) is used to navigate between directories. It requires either the full path or the directory name, depending on your current working directory. If you run this command without any options, it will take you to your home folder. Keep in mind that it can only be executed by users with **[sudo](https://www.geeksforgeeks.org/sudo-command-in-linux-with-examples/)** privileges.

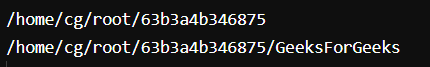
**Syntax:-**

cd <directory name>

**Command:**



**Output:**



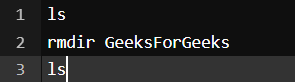
Here we used **pwd**to view the current directory for reference and then we used **cd *GeeksforGeeks***to switch the directory and with again **pwd**command we can see the output is the switched directory, i.e – ***GeeksforGeeks***

* **rmdir command**

The **[rmdir command](https://www.geeksforgeeks.org/rmdir-command-in-linux-with-examples/)** is used to delete permanently an empty directory. To perform this command the user running this command must be having **sudo**privileges in the parent directory.

**Syntax:-**rmdir <directory name>

**Command:**



**Output:**

output of rmdir command in linux

Here we used the **ls**command to check the directories present there and used **rmdir <directory name>** to delete the directory and again the **ls**command to view the directories after deleting the same.

* **ls command**

The [ls command](https://www.geeksforgeeks.org/practical-applications-ls-command-linux/) is commonly used to identify the files and directories in the working directory. This command is one of the many often-used Linux commands that you should know.

This command can be used by itself without any arguments and it will provide us the output with all the details about the files and the directories in the current working directory. There is a lot of flexibility offered by this command in terms of displaying data in the output. Check the below image for the output.

**Syntax:-**

ls

**Output:-**

ls command in linux

* 1. **Study of Unix Shell and Environment Variables.**
     1. **Display The name of the current shell of your System.**

Command: echo $0

* + 1. **Write the different shell names available in Linux.**

**Command: cat /etc/shells**

* + 1. **Display the name of Enviornment Variable.**

**Command: echo $USER**

* 1. **cal, date, echo, bc, who, uname**
     1. Display current month Calendar

**cal**

* + 1. Display the Calendar of given month and year value provided by user.

**cal [month] [year]**

* + 1. Display the calendar of current, previous and next month.

cal -3

* + 1. Display a calendar in which start day of week is Monday.

**cal -m**

* + 1. Display a calendar in which start day of week is Sunday.

**cal -m -S**

* + 1. Display a calendar in which current date is not highlighted.

**cal -h**

* + 1. Display current month number using DATE command.

date +%m

* + 1. Display current year number in two digit form using DATE command.

**date +%y**

* + 1. Display the day of current date in short as well as full name using DATE command.

**date +%a**

**date +%A**

* + 1. Display the month of current date in short as well as full name using DATE command.

date +%b

date +%B

* + 1. Display date in mm/dd/yyyy format.

date +%m/%d/%Y

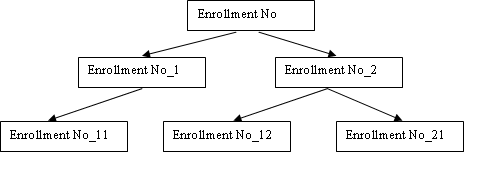
* + 1. Display date in yyyy-mm-dd format.

date +%Y-%m-%d

* + 1. Display the century of current year.

**date +%C**

* 1. **Using pwd, mkdir, cd, rmdir,ls commands generate given tree.**



Student has to create Each directory with his/her enrollment number.

* + 1. Display the path from root directory to the last level directory of the tree.(Consider Root directory of given tree as root)

echo /EnrollmentNo\_/EnrollmentNo\_2/EnrollmentNo\_12

* + 1. Remove Directory Enrollment\_12.

rm -r /EnrollmentNo\_/EnrollmentNo\_2/EnrollmentNo\_12

* + 1. Remove Directore Enrollment\_1.

**rm -r /EnrollmentNo\_/EnrollmentNo\_1**

**Observations:**

***During the lab experiment focused on studying and executing basic and directory manipulation commands of LINUX/UNIX, we observed the functionality and versatility of fundamental commands essential for navigating and managing the file system. Basic commands like ls, cd, pwd, and mkdir allowed us to list directory contents, change directories, display the current working directory, and create new directories, respectively. Additionally, directory manipulation commands such as rmdir and rm -r facilitated the removal of directories, with the latter being particularly useful for deleting directories and their contents recursively.***

**Conclusion:**

***In conclusion, this lab experiment provided valuable hands-on experience with essential LINUX/UNIX commands for basic and directory manipulation tasks. Through the execution of these commands, we gained a deeper understanding of how to navigate the file system, manage directories, and perform common file operations efficiently in a LINUX/UNIX environment. The proficiency gained in executing these commands equips us with essential skills for interacting with the operating system and lays a strong foundation for more advanced LINUX/UNIX administration tasks.***

**Quiz:**

1. **Give Significance of Shell in Linus OS.**

**The shell in Linux is a crucial component that acts as an interface between the user and the operating system. It interprets user commands and executes them, facilitating interaction with the underlying system. The shell provides a command-line interface (CLI) through which users can perform various tasks, such as navigating the file system, running programs, and managing processes. Additionally, it supports scripting, allowing users to automate tasks and create customized workflows. The shell's significance lies in**

**its role as a powerful tool for controlling and manipulating the Linux operating system, offering flexibility, efficiency, and versatility to users and system administrators.**

1. **How to move in/out from directory using cd in single step.**

**The shell in Linux is a crucial component that acts as an interface between the user and the operating system. It interprets user commands and executes them, facilitating interaction with the underlying system. The shell provides a command-line interface (CLI) through which users can perform various tasks, such as navigating the file system, running programs, and managing processes. Additionally, it supports scripting, allowing users to automate tasks and create customized workflows. The shell's significance lies in its role as a powerful tool for controlling and manipulating the Linux operating system, offering flexibility, efficiency, and versatility to users and system administrators.**

1. **Write Use of bc Command.**

**The bc command in Linux is a versatile and powerful arbitrary-precision calculator language. It provides users with a command-line interface for performing various mathematical calculations, including arithmetic operations, trigonometric functions, exponentiation, and more. Users can input mathematical expressions directly into the bc command prompt and receive precise results with adjustable precision settings. Additionally, bc supports script execution, allowing users to automate complex calculations and mathematical tasks. With its rich set of features and flexibility, the bc command serves as a valuable tool for mathematicians, engineers, programmers, and anyone requiring accurate and customizable mathematical computation capabilities in the Linux environment.**

**Suggested Reference:**

1. Operating Systems: Internals & Design Principles, 9th Edition, William Stallings, Pearson Education India
2. Operating System Concepts, 9th edition Peter B. Galvin, Greg Gagne, Abraham Silberschatz, John Wiley & Sons, Inc.
3. Modern Operating Systems-By Andrew S. Tanenbaum (PHI)
4. UNIX : Concepts and Applications | 4th Edition by Sumitabha Das ,McGrawHill

# References used by the students:

# <https://www.geeksforgeeks.org/basic-linux-commands/>

# <https://chat.openai.com/c/193837ef-579f-44ce-a8f0-733dac107e86>

# https://gemini.google.com/app/e91e80883ea1450e

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of commands (4) | | | Ability to use Command for question solving(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

**Experiment No: 3**

**AIM:** Study and execute Basic File manipulation commands.

1. cat 2. wc 3. cp 4. mv 5.rm 6. File 7. cmp 8. comm 9. diff 10. chmod 11. sort

**Date:**

**Competency and Practical Skills:** Basic Skills to work with Computer System/ Linux Terminal

**Relevant CO:** CO5

**Objectives:**

1. To understand the importance of Shell/Environment Variable
2. To work with basic File Operations.
3. To work/Access the Directory commands.

**Equipment/Instruments:** Computer System with Linux OS.

**Safety and necessary Precautions:**

* + Operate computer system carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

File manipulation commands are mainly used for Operations like create File, Copy File, Delete File, rename file, Searching No. of counts from file, finding Common values from two file, Comparing to files, finding difference between two file to make them identical. There is also File Permission command chmod to change the file permission.

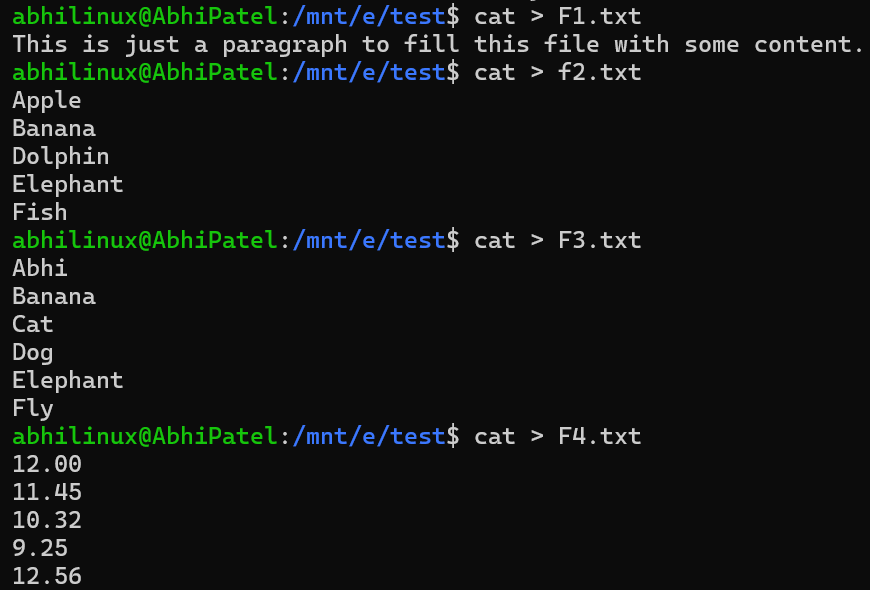
**Basic File commands**

|  |  |
| --- | --- |
| **Command Name** | **Description** |
| cat | It redirects standard output to/from the file. |
| wc | It counts words, characters and lines from the file. |
| cp | Copy one file to another. |
| mv | Rename or move files from one directory to another. |
| rm | Remove the file from directory. |
| file | Displays file types and other details. |
| cmp | Compare the content of two files. |
| comm. | Find common between two files. |
| diff | Find the difference between two give way to make them identical. |
| chmod | It changes file permission. |
| sort | It sorts the file content. |

* 1. **Create 4 Files using cat Command.**
* **Create 4 Files F1.txt, F2.txt, F3.txt, F4.txt with some content.**
* **Preferable to put a paragraph in one file.**
* **In Files F2 and F3 write content in the form of words in alphabetically sorted order**

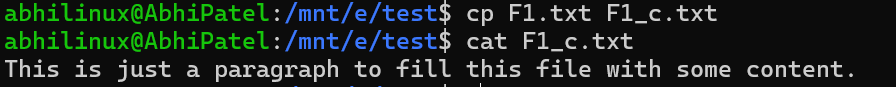
**Also put some common values in both files (like student name or engineering branch names, fruits, vegetables, etc.)**

* **In the fourth File write decimal numbers separated by a newline.**



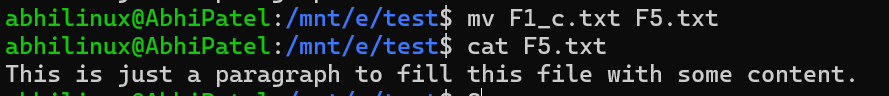
* 1. Copy Content of File F1 to F1\_c.

cp F1.txt F5.txt



* 1. Rename File F1\_c to F5.

mv F1\_c.txt F5.txt



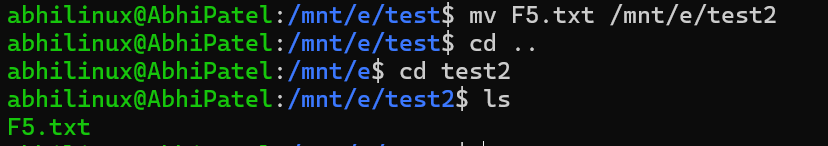
* 1. Compare File F1 and F5

cmp F1.txt F5.txt



* 1. Move file F5 to another Directory of your Choice.

mv F5.txt /home/test2



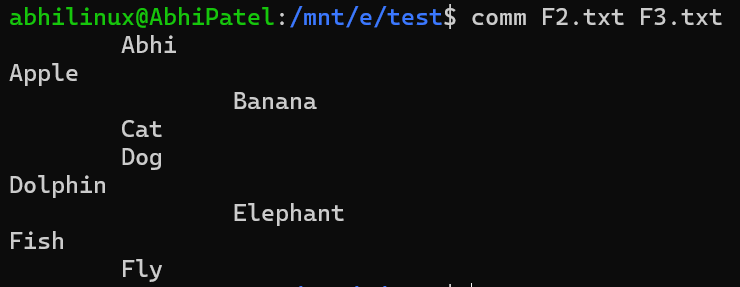
* 1. Count no. of characters, words and Lines of F1 file.

wc F1.txt



* 1. Find Common Value between File F2 and F3.

comm F2.txt F3.txt



* 1. Find the Difference Between File F2 and F3.

diff F2.txt F3.txt



* 1. Change the F1 file permission to read Only.

chmod +r F1.txt



* 1. Change F2 permission to Read and Write Only.

chmod +rw F2.txt



* 1. Change File F4 permission to Read, Write and Execute.

chmod +rwx F4.txt



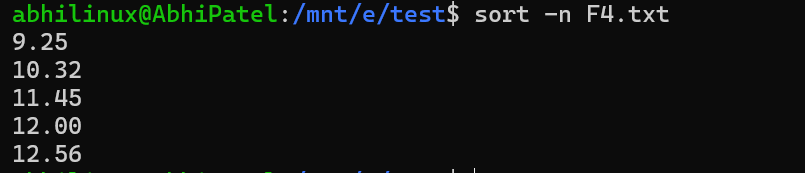
* 1. Perform sort command on F4.

sort F4.txt



* 1. Perform numeric Sort on File F4.

sort -n F4.txt



**Observations:**

The practical utilization of Linux commands have significantly enhanced my understanding of file manipulation and system administration. These commands offer efficient methods for comparing files, detecting discrepancies, organizing data, and controlling file permissions. Through hands-on experience, I've observed the effectiveness of each command in facilitating tasks such as identifying unique and common lines, byte-by-byte comparison, highlighting differences, sorting data, and managing file access.

**Conclusion:**

In conclusion, the exploration of Linux commands such as comm, cmp, diff, sort, and chmod has provided insights into file management. These commands enable efficient comparison of files, identification of differences, sorting of data, and modification of file permissions. Through practical application, I have gained proficiency in analyzing, organizing, and securing files and directories, enhancing my skills in Linux administration.

**Quiz:**

* + - 1. Why there is need to change file permission.

The need to change file permissions arises to control access to files and directories, ensuring the security and privacy of data. Changing permissions allows restricting or granting read, write, and execute permissions to users, groups, and others based on specific requirements.

* + - 1. How to Change File permission in a different way using chmod command.

File permissions can be changed using the chmod command in various ways:

Numerical mode: Using octal numbers to specify permission modes (e.g., chmod 755 filename).

Symbolic mode: Using symbols (+, -, = along with letters (u, g, o) and permissions (r, w, x) to modify permissions (e.g., chmod u+x filename).

* + - 1. Write the use of chown command.

The chown command is used to change the ownership of files and directories in Linux. It allows changing both the owner and group of a file or directory, providing administrative control over file ownership and access.

* + - 1. Write use of “**|”** character in Linux.

The " **|** " character in Linux is known as a pipe symbol. It is used in command-line interfaces to redirect the output of one command as input to another command. This enables chaining multiple commands together, allowing for complex data manipulation and processing.

**Suggested Reference:**

1. Operating Systems: Internals & Design Principles, 9th Edition, William Stallings, Pearson Education India
2. Operating System Concepts, 9th edition Peter B. Galvin, Greg Gagne, Abraham Silberschatz, John Wiley & Sons, Inc.
3. Modern Operating Systems-By Andrew S. Tanenbaum (PHI)
4. UNIX: Concepts and Applications | 4th Edition by Sumitabha Das ,McGrawHill

# References used by the students:

# <https://www.geeksforgeeks.org/basic-linux-commands/>

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of commands (4) | | | Ability to use Command for question solving(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

**Experiment No: 4**

**AIM:** Study and Execute Advance Filter Commands.

1. head 2. tail 3. paste 4. cut(-f) 5. cut(-c) 6. grep

**Date:**

**Competency and Practical Skills:** Basic Skills to work with Computer System/ Linux Terminal

**Relevant CO:,** CO5

**Objectives:**

1. To understand the importance of File filter commands.
2. To merge, split and Search in different way from the file.

**Equipment/Instruments:** Computer System with Linux OS.

**Safety and necessary Precautions:**

* + Operate computer system carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

Filter commands accept input data from standard input and produce output standard output. It transforms plain-text data in a meaningful way and can be used to merge with other output or files. These filters are very small programs that are designed for a specific function which can be used as building blocks. There are number of commands we already have covered I n previous practicals like sort, comm., cat and others are cut , paste, head, tail, grep, tee ,uniq**, grep. Using these commands one can search and display specif content from files also filter some specific data.**

**For more description about the command we can read a Linux Help manual using command :**

**$man Commandname.**

|  |  |
| --- | --- |
| **Command Name** | **Description** |
| head | It displays first 10 lines from the input file. |
| tail | It displays last 10 lines from the file. |
| **cut** | It cut the file content Vertically(-f) as well as Horizontally(-c). |
| paste | It pastes content of different files and displays the output. |
| **grep** | Search the line with specified pattern present in the file(s). |

**grep command searches the simple pattern as well pattern specified using Regular expression.To create a pattern which matches different types of string we can use different wildcard characters to create a new pattern.** The basic set of wildcards in are:

**\* –**This wildcard represents all the characters. Also represent one or more occurrence of preceded character.

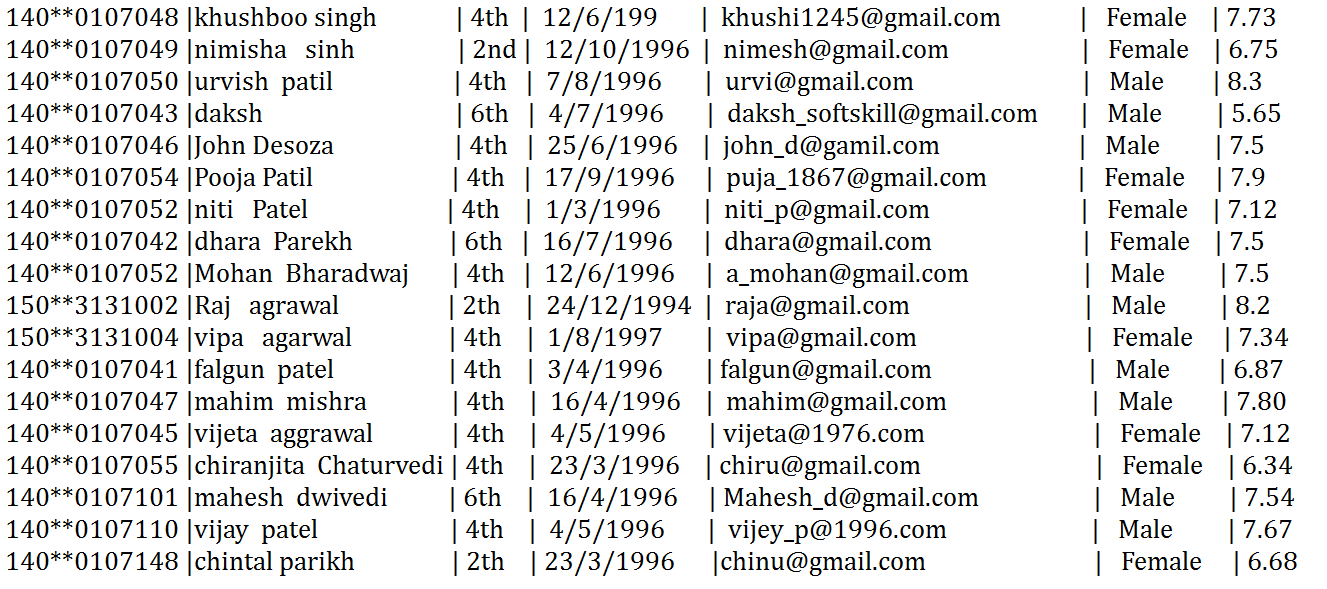
**+ -** represent one or more occurrence of preceded character.

**? –**This wildcard represents a single character

**[ ] –**This wildcard represents a range of characters.

To solve the question given student has to create a text file with Student details like:

**Enrollment number, Student name, Birth-date, Semester, Gender, Email Address, SPI, where each field is separated by delimiter character “|” or any character as per wish.**



* 1. Display first 7 lines of the file.
  2. Display Last 4 Lines of the File.
  3. Cut the file column wise and display Student Enrollment number, Gender, Email Address.
  4. Cut the file column wise and display Student name, Branch, semester.
  5. Merge the result of Question 4.3 and 4.4 using delimiter $ and store it File named result1 using tee command.
  6. Cut the File Fieldwise (vertically) to display second, third and fourth field of the file.
  7. Display the First three student details having highest SPI in decreasing order.
  8. Change file delimiter “|” with another “\* “.
  9. Display student details studying in same branch.
  10. Display student details studying in same branch and same semester.
  11. Display students have SPI greater than 6.
  12. Display student details studying Whose surname starts will “p” and ends with “l”.
  13. Display the name of student whose surname is agarwal (surname may be in any form).
  14. Display details of all girls students.

**Observations:**

***// Write your observation here***

**Conclusion:**

***// Write conclusion here***

**Quiz:**

1. What is the use of tee and cat command.
2. By default, how many lines are displayed using the head command? which option used to display specific lines from the file?
3. Which option is used to display file content in reverse order?

**Suggested Reference:**

1. Operating Systems: Internals & Design Principles, 9th Edition, William Stallings, Pearson Education India
2. Operating System Concepts, 9th edition Peter B. Galvin, Greg Gagne, Abraham Silberschatz, John Wiley & Sons, Inc.
3. Modern Operating Systems-By Andrew S. Tanenbaum (PHI)
4. UNIX : Concepts and Applications | 4th Edition by Sumitabha Das ,McGrawHill

# References used by the students:

# *// Write references used by you here*

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of commands (4) | | | Ability to use Command for question solving(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

# 

**Experiment No: 5**

**AIM : Write a shell script program using Loop/ control structure.**

* 1. **Write a shell script to find factorial of given number n**
  2. **Write a shell script which will generate first n fibonnacci numbers like: 1, 1, 2, 3, 5, 13,…**

**Date: *// Write date of experiment here***

**Competency and Practical Skills:** Logic building and programming

**Relevant CO:** CO5

**Objectives:**

1. To understand and use the loop and control structure to sove problem using shell script.

**Equipment/Instruments:** Computer System with Linux OS.

**Safety and necessary Precautions:**

* + Operate computer system carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

Linux/Unix shells are interactive, by means they accept commands as input from users and execute them and display the output accordingly. Normally we are executing command independently one by one. Sometimes it is require to execute same no.of commands repeatedly which is time consuming task but linux system also has solution for that called shell programming. For that we can put all commands together in single file and execute them in shell to avoid repetitive work. These files are called **Shell Scripts**or**Shell Programs**. Shell scripts are similar to the batch file in MS-DOS. The shell script file is saved with**“.sh”**  extension e.g., **First\_program.sh.**

A shell script has syntax like other programming language. If you have any prior experience of programming language like Python, C/C++ etc. It would be very easy to understand shell programming. It has it’s Shell Keywords, Control flow statements, Loop statements, Shell (linux command we had used in previous experiments)and Functions.

To run the shell script program, file must be executable. To set execute permission of file we can use chmod command.e.g**. chmod 777 file.sh** command allows file.sh to execute.to run the file one can write the command **./file.sh.**

* **There are 3 types loop statements supported by shell programming:**

1. **while statement**

Syntax:

while <condition>  
do  
 <command statement 1>  
 <command statement 2>

.

.

<command statement n>  
done

1. **for statement**

The for loop operates on lists . It repeats a set of commands for every item as per list value.  var is the variablename and var takes value from the list value1, value2, … value n on each iteration, respectively,

**Syntax:**

for <var> in <value 1 value 2 ... value n>  
do  
 <command statement 1>  
 <command statement 2>

.

.

<command statement n>  
done

1. **do…until statement**

The do… until loop is executed as many times as th condition/command evaluates too false. The loop terminates when the condition/command becomes true.

**Syntax:**

until <condition>  
do  
 <command statement 1>  
 <command statement 2>

.

.  
done

To change the flow of loop statements, two commands are used they are,

1. break
2. continue

* **There are basically 2 types of contro statement supported in shell programming:**

1. **if –else statement (different versions)**
2. **Simple if statement**

**Syntax:**

if [ expression ]

then

<command statement 1>

fi

1. **if-else statement**

**Syntax:**

if [ expression ]

then

<command statement 1>

else

<command statement 2>

fi

1. if..elif..else..fi statement (Else If ladder)

**Syntax:**

if [ expression1 ]

then

<command statement 1>

<command statement 2>

elif [ expression2 ]

then

<command statement 3>

<command statement 4>

.

else

<command statement 5>

fi

1. if..then..else..if..then..fi..fi..(Nested if)

**Syntax:**

1. **switch statement**

**Syntax:**

case “expression” in

Pattern 1) < command Statement 1> ;;

Pattern 2) < command Statement 2> ;;

.

.

.

Pattern n) < command Statement n> ;;

esac

**Example1 : “Checking whether two numbers are equal or not?”**

a=20

b=21

if [ $a -gt $b ]

then

#If they are equal then print this

echo "a is greater than b"

else

#else print this

echo "a is less than b"

fi

**OUTPUT:**

5.PNG

**Example 2: “shell script to Display branchcode of respective branch using case control statement”**

echo "Enter department name"

read DEPARTMENT

case $DEPARTMENT in

"CE") echo " branch code:07" ;;

"EC") echo " branch code:11" ;;

"CIVIL") echo " branch code:06" ;;

"IT") echo " branch code:16" ;;

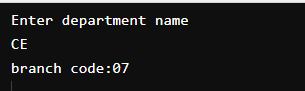
"MECH") echo " branch code:19" ;;

\*) echo -n "Invalid"

;;

esac

**OUTPUT:**

****

* 1. **Write a shell script to find factorial of given number n.**

**Program:**

***// Write code of your program here***

**Output:**

***// Write output of your program here***

* 1. **Write a shell script which will generate first n fibonnacci numbers like: 1, 1, 2, 3, 5, 13,…**

**Program:**

***// Write code of your program here***

**Output:**

***// Write output of your program here***

**Observations:**

***// Write your observation here***

**Conclusion:**

***// Write conclusion here***

**Quiz:**

1. How to read and print the value of variable in shell script?
2. Write syntax to retrieve variable value using shell script.
3. Write a step to run shell script.

**Suggested Reference:**

1. Operating Systems: Internals & Design Principles, 9th Edition, William Stallings, Pearson Education India
2. Operating System Concepts, 9th edition Peter B. Galvin, Greg Gagne, Abraham Silberschatz, John Wiley & Sons, Inc.
3. Modern Operating Systems-By Andrew S. Tanenbaum (PHI)
4. UNIX : Concepts and Applications | 4th Edition by Sumitabha Das ,McGrawHill

# References used by the students:

# *// Write references used by you here*

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of Shell programming syntax (4) | | | Ability to implement program for given problem using Shell script(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

# 

**Experiment No – 6**

**AIM :** Loop/ control structure using shell script(Using while loop)

1. Write a shell script to read n numbers as command arguments and sort them in descending order.
2. Write a shell script to generate mark sheet of a student. Take 3 subjects, calculate and display total marks, percentage and Class obtained by the student.

**Date: *// Write date of experiment here***

**Competency and Practical Skills:** Logic building and programming

**Relevant CO:** CO5

**Objectives:** explore usage of while loop in shell script

**Equipment/Instruments:** Computer System with Winows/Linux

**Safety and necessary Precautions:**

* + Operate computer system carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

Example:

Shows loop terminates as soon as a becomes 5

a=0

while [ $a -lt 10 ]

do

echo $a

if [ $a -eq 5 ]

then

break

fi

a=`expr $a + 1`

done

**6.1 Write a shell script to read n numbers as command arguments and sort them in descending order.**

**Program:**

**Output:**

* 1. **Write a shell script to generate mark sheet of a student. Take 3 subjects, calculate and display total marks, percentage and Class obtained by the student.**

**Output:**

**Observations:**

***// Write your observation here***

**Conclusion:**

***// Write conclusion here***

**Suggested Reference:**

[**https://www.tutorialspoint.com/**](https://www.tutorialspoint.com/)

[**https://www.geeksforgeeks.org/**](https://www.geeksforgeeks.org/)

**https://www.javatpoint.com/**

# <https://www.tutorialspoint.com/unix/unix-loop-control.htm>

# References used by the students:

# *// Write references used by you here*

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of Shell programming syntax (4) | | | Ability to implement program for given problem using Shell script(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

**Experiment No – 7**

**AIM:** Command execution via Shell script.

* 1. Write a shell script to display all executable files, directories and zero sized files from current directory.
  2. Write a menu driven shell script which will print the following menu and execute the given task.

MENU

* Display calendar of current month
* Display today’s date and time
* Display usernames those are currently logged in the system
* Display your name at given x, y position
* Display your terminal number

Exit

**Date: *// Write date of experiment here***

**Competency and Practical Skills:** Logic building and programming

**Relevant CO:** CO5

**Objectives:** explore usage of various searching and date - time related commands.

**Equipment/Instruments:** Computer System with Winows/Linux

**Safety and necessary Precautions:**

* + Operate computer system carefully and responsibly.
  + Use required lab resources cautiously

**7.1 Write a shell script to display all executable files, directories and zero sized files from current directory.**

**Theory:**

find command

The **find** command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date, owner and permissions. By using the ‘-exec’ other UNIX commands can be executed on files or folders found.

Options:

* -exec CMD: The file being searched which meets the above criteria and returns 0 for as its exit status for successful command execution.
* -ok CMD : It works same as -exec except the user is prompted first.
* -inum N : Search for files with inode number ‘N’.
* -links N : Search for files with ‘N’ links.
* -name demo : Search for files that are specified by ‘demo’.
* -newer file : Search for files that were modified/created after ‘file’.
* -perm octal : Search for the file if permission is ‘octal’.
* -print : Display the path name of the files found by using the rest of the criteria.
* -empty : Search for empty files and directories.
* -size +N/-N : Search for files of ‘N’ blocks; ‘N’ followed by ‘c’can be used to measure the size in characters; ‘+N’ means size > ‘N’ blocks and ‘-N’ means size < ‘N’ blocks.
* -user name : Search for files owned by username or ID ‘name’.
* \(expr \) : True if ‘expr’ is true; used for grouping criteria combined with OR or AND.
* ! expr : True if ‘expr’ is false.

Example: Search for a file with a specific name.

**1.$ find ./GFG -name sample.txt**

It will search for sample.txt in GFG directory.

**2. $ find ./GFG -name \*.txt**

It will give all files which have ‘.txt’ at the end.

**Program:**

**Output:**

**7.2 Write a menu driven shell script which will print the following menu and execute the given task.**

MENU

* Display calendar of current month
* Display today’s date and time
* Display usernames those are currently logged in the system
* Display your name at given x, y position
* Display your terminal number

Exit

**Theory:**

# Case Statement

A case statement in bash scripts is used when a decision has to be made against multiple choices. In other words, it is useful when an expression has the possibility to have multiple values. This methodology can be seen as a replacement for multiple if-statements in a script. Case statements have an edge over if-statements because it improves the readability of our code and they are easier to maintain. Case statements in a Bash script are quite similar to Case statements in C language. But unlike C, the Bash Case statement stops continuing the search as soon as the match occurs. In simple words, they don’t require any break statement that is mandatory to be used in C to stop searching for a pattern further.

The basic syntax of a case statement is given below,

Syntax:

case EXPRESSION in

Pattern\_Case\_1)

STATEMENTS

;;

Pattern\_Case\_1)

STATEMENTS

;;

Pattern\_Case\_N)

STATEMENTS

;;

\*)

STATEMENTS

;;

esac

**cal command**

By default, the cal command shows the current month calendar as output.

cal command is a calendar command in Linux which is used to see the calendar of a specific month or a whole year.

Syntax:

cal [ [ month ] year]

cal 08 2000 : Shows calendar of selected month and year.

cal 2018 : Shows the whole calendar of the year.

cal -3 : Shows calendar of previous, current and next month

**who command**

The who command is used to get information about currently logged in user on to system.

Syntax : $who [options] [filename]

Examples :

1. The who command displays the following information for each user currently logged in to the system if no option is provided :

Login name of the users

Terminal line numbers

Login time of the users in to system

Remote host name of the user

**Program:**

**Output:**

**Observations:**

***// Write your observation here***

**Conclusion:**

***// Write conclusion here***

**Suggested Reference:**

[**https://www.tutorialspoint.com/**](https://www.tutorialspoint.com/)

[**https://www.geeksforgeeks.org/**](https://www.geeksforgeeks.org/)

**https://www.javatpoint.com/**

# References used by the students:

# *// Write references used by you here*

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of Shell programming syntax (4) | | | Ability to implement program for given problem using Shell script(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

**Experiment No – 8**

**AIM :** Process Scheduling Algorithm and Comparison.

**Date: *// Write date of experiment here***

**Competency and Practical Skills:** Logic building and programming

**Relevant CO:** CO2

**Objectives:** Study and implement process scheduling algorithms.

**Equipment/Instruments:** Computer System with Winows/Linux

**Safety and necessary Precautions:**

* + Operate computer systems carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

**8.1 Write a C Program to Implement Following CPU Scheduling Algorithms.**

* **FCFS**
* **Round Robin**

**First Come First Serve (FCFS)** is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which request the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.

**Characteristics of FCFS CPU Scheduling Algorithm**

**•** It supports non-preemptive and pre-emptive scheduling algorithm.

• Jobs are always executed on a first-come, first-serve basis.

• It is easy to implement and use.

• This method is poor in performance, and the general wait time is quite high.

**Round** Robin is a CPU scheduling algorithm where each process is assigned a fixed time slot in a cyclic way. It is basically the preemptive version of First come First Serve CPU Scheduling algorithm. Round Robin CPU Algorithm generally focuses on Time Sharing technique.

The period of time for which a process or job is allowed to run in a pre-emptive method is called time quantum.

Each process or job present in the ready queue is assigned the CPU for that time quantum, if the execution of the process is completed during that time then the process will end else the process will go back to the waiting table and wait for its next turn to complete the execution.

**Characteristics of Round Robin CPU Scheduling Algorithm**

It is simple, easy to implement, and starvation-free as all processes get fair share of CPU.

One of the most commonly used technique in CPU scheduling as a core.

It is preemptive as processes are assigned CPU only for a fixed slice of time at most.

The disadvantage of it is more overhead of context switching.

**Program:**

**Output:**

**Observations:**

***// Write your observation here***

**Conclusion:**

***// Write conclusion here***

**Suggested Reference:**

https://www.tutorialspoint.com/

https://www.geeksforgeeks.org/

https://www.javatpoint.com/

# References used by the students:

# *// Write references used by you here*

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of Shell programming syntax (4) | | | Ability to implement program for given problem using Shell script(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

**Experiment No – 9**

**AIM : Process creation and Thread Scheduling**

9.1 Implement Producer consumer problem using thread using C/JAVA programming Language.

9.2 Create new thread using fork() system call using C programming Language.

**Date: *// Write date of experiment here***

**Competency and Practical Skills:** Logic building and programming

**Relevant CO:** CO3

**Objectives:** Study and implement thread management.

**Equipment/Instruments:** Computer System with Winows/Linux

**Safety and necessary Precautions:**

* + Operate computer systems carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

The producer-consumer problem (also known as the bounded-buffer problem) is a classic example of a multi-process synchronization problem. The problem describes two processes, the producer and the consumer, which share a common, fixed-size buffer used as a queue.

The producer’s job is to generate data, put it into the buffer, and start again. At the same time, the consumer is consuming the data (i.e. removing it from the buffer), one piece at a time.

Problem:

To make sure that the producer won’t try to add data into the buffer if it’s full and that the consumer won’t try to remove data from an empty buffer.

Solution:

The producer is to either go to sleep or discard data if the buffer is full. The next time the consumer removes an item from the buffer, it notifies the producer, who starts to fill the buffer again. In the same way, the consumer can go to sleep if it finds the buffer to be empty. The next time the producer puts data into the buffer, it wakes up the sleeping consumer.

An inadequate solution could result in a deadlock where both processes are waiting to be awakened.

Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU. Each part of such program is called a thread. So, threads are light-weight processes within a process.

Threads can be created by using two mechanisms :

* Extending the Thread class
* Implementing the Runnable Interface

The **Fork** system call is used for creating a new process in Linux, and Unix systems, which is called the child process, which runs concurrently with the process that makes the fork() call (parent process). After a new child process is created, both processes will execute the next instruction following the fork() system call.

The child process uses the same pc(program counter), same CPU registers, and same open files which use in the parent process. It takes no parameters and returns an integer value.

Below are different values returned by fork().

**Negative Value:** The creation of a child process was unsuccessful.

**Zero:** Returned to the newly created child process.

**Positive value:** Returned to parent or caller. The value contains the process ID of the newly created child process.

**Example:**

#include <stdio.h>

#include <sys/types.h>

#include <unistd.h>

int main()

{

// make two process which run same

// program after this instruction

pid\_t p = fork();

if(p<0){

perror("fork fail");

exit(1);

}

printf("Hello world!, process\_id(pid) = %d \n",getpid());

return 0;

}

**9.1 Implement Producer consumer problem using thread using C/JAVA programming Language.**

**Program:**

**Output:**

**9.2 Create new thread using fork() system call using C programming Language.**

**Program:**

**Output:**

**Observations:**

***// Write your observation here***

**Conclusion:**

***// Write conclusion here***

**Suggested Reference:**

https://www.tutorialspoint.com/

https://www.geeksforgeeks.org/

https://www.javatpoint.com/

# <https://www.geeksforgeeks.org/producer-consumer-solution-using-threads-java/>

# https://www.geeksforgeeks.org/thread-functions-in-c-c/

# References used by the students:

# *// Write references used by you here*

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of Shell programming syntax (4) | | | Ability to implement program for given problem using Shell script(4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |

**Experiment No – 10**

**AIM :** Page replacement and Disk Scheduling algorithm

10.1 Implement FIFO Page replacement Algorithm using C/Java.

10.2 Implement C-SCAN Disk Scheduling Algorithm using C/Java.

**Date: *// Write date of experiment here***

**Competency and Practical Skills:** Logic building and programming

**Relevant CO:** CO4

**Objectives:** Study and implement memory management by operating system.

**Equipment/Instruments:** Computer System with Winows/Linux

**Safety and necessary Precautions:**

* + Operate computer systems carefully and responsibly.
  + Use required lab resources cautiously

**Theory:**

In an operating system that uses paging for memory management, a page replacement algorithm is needed to decide which page needs to be replaced when a new page comes in.

**Page Fault:** A page fault happens when a running program accesses a memory page that is mapped into the virtual address space but not loaded in physical memory. Since actual physical memory is much smaller than virtual memory, page faults happen. In case of a page fault, Operating System might have to replace one of the existing pages with the newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce the number of page faults.

**First In First Out (FIFO):** This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.

A Process makes the I/O requests to the operating system to access the disk. Disk Scheduling Algorithm manages those requests and decides the order of the disk access given to the requests.

Important Terms related to Disk Scheduling Algorithms

**Seek Time -** It is the time taken by the disk arm to locate the desired track.

Rotational Latency - The time taken by a desired sector of the disk to rotate itself to the position where it can access the Read/Write heads is called Rotational Latency.

Transfer Time - It is the time taken to transfer the data requested by the processes.

**Disk Access Time -** Disk Access time is the sum of the Seek Time, Rotational Latency, and Transfer Time.

**C-SCAN**

This algorithm is the same as the SCAN algorithm. The only difference between SCAN and C-SCAN is, it moves in a particular direction till the last and serves the requests in its path. Then, it returns in the opposite direction till the end and doesn't serve the request while returning. Then, again reverses the direction and serves the requests found in the path. It moves circularly.

**10.1: Implement FIFO Page replacement Algorithm using C/Java.**

**Program:**

**Output:**

**10.2: Implement C-SCAN Disk Scheduling Algorithm using C/Java.**

**Program:**

**Output:**

**Observations:**

***// Write your observation here***

**Conclusion:**

***// Write conclusion here***

**Suggested Reference:**

https://www.tutorialspoint.com/

https://www.geeksforgeeks.org/

https://www.javatpoint.com/

# <https://www.baeldung.com/cs/fifo-page-replacement>

# References used by the students:

# *// Write references used by you here*

# Rubric-wise marks obtained:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Rubrics | Understanding of Shell programming syntax (4) | | | Ability to implement program for given problem using Shell script (4) | | | Documentation &Timely Submission(2) | | | Total(10) |
| Good(4) | Avg.(3-2) | Poor(1-0) | Good(4) | Avg.(3-2) | Poor(1-0) | Good(2) | Avg.(1) | Poor(0) |  |
| Marks |  | | |  | | |  | | |  |