

|  |  |  |
| --- | --- | --- |
| **Operating Systems** | | |
| Lab Manual | | |
| **Department of Computer Science and Engineering**  **The NorthCap University, Gurugram** | | |
|  | |
|  | |
|  | |
|  | |
|  |

**Operating Systems Lab Manual**

**CSL 303**

**Dr. Priyanka Vasisth**



Department of Computer Science and Engineering

NorthCap University, Gurugram- 122001, India

Session 2020-21

*Published by:*

**School of Engineering and Technology**

**Department of Computer Science & Engineering**

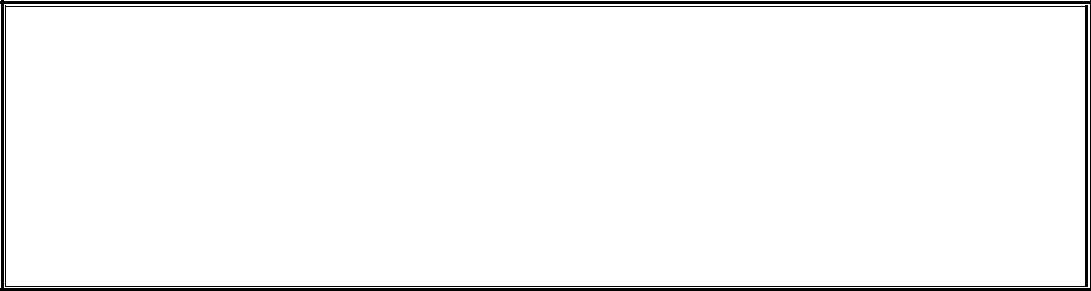
**The NorthCap University Gurugram**

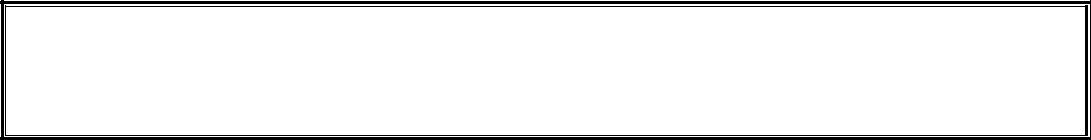
**• Laboratory Manual is for Internal Circulation only**

**© Copyright Reserved**

*No part of this Practical Record Book may be*

*reproduced, used, stored without prior permission of The NorthCap University*



Copying or facilitating copying of lab work comes under cheating and is considered as use of unfair means. Students indulging in copying or facilitating copying shall be awarded zero marks for that particular experiment. Frequent cases of copying may lead to disciplinary action. Attendance in lab classes is mandatory.

Labs are open up to 7 PM upon request. Students are encouraged to make full use of labs beyond normal lab hours.

**PREFACE**

Operating System Lab Manual is designed to meet the course and program requirements of NCU curriculum for B.Tech III year students of CSE branch. The concept of the lab work is to give brief practical experience for basic lab skills to students. It provides the space and scope for self-study so that students can come up with new and creative ideas.

The Lab manual is written on the basis of “teach yourself pattern” and expected that students who come with proper preparation should be able to perform the experiments without any difficulty. Brief introduction to each experiment with information about self-study material is provided. The laboratory exercises will include familiarization with LINUX system calls for process management and inter-process communication; Experiments on process scheduling and other operating system tasks through simulation/implementation. Students would require design process synchronization, CPU scheduling algorithms, memory management and disc management algorithms in high level languages like c, c++, python. Finally, the students would require applying the operating system concepts by experimenting on either xv6/minix operating systems. At the start of each experiment a question bank for preparation and practice is suggested which may be used to test the basic understanding of the students about the experiment. Students are expected to come thoroughly prepared for the lab.General disciplines, safety guidelines and report writing are also discussed.

The lab manual is a part of curriculum for the TheNorthCap University, Gurugram. Teacher’s copy of the experimental results and answer for the questions are available as sample guidelines.

We hope that lab manual would be useful to students of CSE, IT, ECE and BSc branches and author requests the readers to kindly forward their suggestions / constructive criticism for further improvement of the work book.

Author expresses deep gratitude to Members, Governing Body-NCU for encouragement and motivation.

**Authors**

**The NorthCap University**

**Gurugram, India**

|  |  |  |  |
| --- | --- | --- | --- |
| **CONTENTS** | | | |
| S.N. | **Details** | Page No. | |
|  | **Syllabus** |  | |
| 1 | **Introduction** |  | |
| 2 | **Lab Requirement** |  |
| 3 | **General Instructions** |  |
| 4 | **List of Experiments** |  |
| 5 | **List of Flip Assignment** |  |
| 6 | **List of Projects** |  |
| 7 | **Rubrics** |  |
| 8 | **Annexure 1 (Format of Lab Report)** |  |
| 9 | **Annexure 2 (Format of Lab Certificate)** |  |

**SYLLABUS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. **Department:** | | **Department of CSE** | | |
| 1. **Course Name: Operating Systems** | | 1. **Course Code :** | 1. **L- P** | 1. **Credits** |
| **Code: CSL 303** | 3-2 | 4 |
| 1. **Type of Course (Check one):** | | Programme Core Programme Elective Open Elective  **✓** | | |
| **✓**   1. **Frequency of offering (check one):**Odd Even Either Sem. Every Sem. | | | | |
| 1. **Brief Syllabus:** This is an introductory course which briefs LINUX Operating System Concepts that forms an integral part of computer science engineering in development of software applications in many diverse areas, including Web Development, Windows Applications, Research, Analytics and Processing. It lays the foundation of Process Management & Scheduling, Memory Management, Deadlocks and other Operating system Concepts. | | | | |
| 1. **Total lecture and Practical Hours for this course: 30 Hours**   The class size is maximum 30 learners. | | | | |
| 1. **Course Outcomes (COs)**   Possible usefulness of this course after its completion i.e. how this course will be practically useful to him once it is completed | | | | |
| **CO 1** | The students will be able to understand the basic architecture of Linux. | | | |
| **CO 2** | The students will be able to understand the process management & scheduling of Linux. | | | |
| **CO 3** | The students will be able to understand the memory management of Linux. | | | |
| **CO 4** | The students will be able to understand the inter process communication of Linux. | | | |
| **CO 5** | They will understand the main principles and techniques to handle the deadlocks. | | | |
| **CO6** | They will understand the I/O device management & the VFS of Linux. | | | |
| 1. **UNIT WISE DETAILS No. of Units: -05** | | | | |
| **Unit 1: Introduction to Linux OS Hours: 6**  Introduction & overview: functions of operating systems, Overview of various Operating Systems, Linux architecture, Boot strap loader of Linux, Tasks of the kernel, implementation strategies of kernel, System Calls. | | | | |
| **Unit II: Process Management & Scheduling Hours: 6**  Process priorities, process life cycle of Linux, process representation: process types, process identification numbers, process management system calls, kernel thread, overview of different scheduling algorithms, Linux scheduler: priority and completely fair share scheduling algorithm. | | | | |
| **Unit III: Process Synchronization and Memory Management Hours: 8**  Implementation of Producer- Consumer problem, implementation of semaphores, Page-Replacement Algorithms. | | | | |
| **Unit IV: Deadlocks Hours: 6**  Implementation of Banker’s Algorithm, | | | | |
| **Unit V: Virtual File System Hours: 4**  Disk scheduling algorithms, Introduction to VFS File System types, Common File model, Structure of the VFS | | | | |
| 1. **Guided Project (No. of Hours):**Case Study on Windows OS 2. **Unguided Project (No. of Hours):**Case Study ofLinux, Window, MAC OS | | | | |
| 1. **Brief Description of Self-learning component by students (through books/resource material etc.): Topics:** Linux syntax for shell scripting, revise c/c++/Python and data structure concepts from previous semesters | | | | |
| 1. **Suggested Readings**   GNU/Linux Command−Line Tools Summary [eBook]  <http://www.tldp.org/LDP/Bash-Beginners-Guide/Bash-Beginners-Guide.pdf>  **websites:**   * <https://www.linuxjournal.com/> * <https://www.omgubuntu.co.uk/> | | | | |

1. **INTRODUCTION**

That ‘learning is a continuous process’ cannot be over emphasized. The theoretical knowledge gained during lecture sessions need to be strengthened through practical experimentation. Thus practical makes an integral part of a learning process.

The purpose of conducting experiments can be stated as follows:

* To familiarize the students with the basic concepts, programming skill development and the take home laboratory assignments mainly implementation-oriented which have to be coded in high level language. The lab sessions will be based on exploring the concepts discussed in class.
* Observing basic structure and characteristics of Operating Systems
* Reporting and analyzing the complexities.
* Hands on experience on the experimental setup and software tools

1. **LAB REQUIREMENTS**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Requirements** | **Details** |
| **1** | **Software Requirements** | Linux’s Shell, Python/c/c++ |
| **2** | **Operating System** | Linux Operating System |
| **3** | **Hardware Requirements** | Windows and Linux: Intel 64/32 or AMD Athlon 64/32, or AMD Opteron processor  2 GB RAM  80 GB hard disk space |
| **4** | **Required Bandwidth** | NA |

1. **GENERAL INSTRUCTIONS** 
   1. **General discipline in the lab**
   * Students must turn up in time and contact concerned faculty for the experiment they are supposed to perform.
   * Students will not be allowed to enter late in the lab.
   * Students will not leave the class till the period is over.
   * Students should come prepared for their experiment.
   * Experimental results should be entered in the lab report format and certified/signed by concerned faculty/ lab Instructor.
   * Students must get the connection of the hardware setup verified before switching on the power supply.
   * Students should maintain silence while performing the experiments. If any necessity arises for discussion amongst them, they should discuss with a very low pitch without disturbing the adjacent groups.
   * Violating the above code of conduct may attract disciplinary action.
   * Damaging lab equipment or removing any component from the lab may invite penalties and strict disciplinary action.
   1. **Attendance**

* Attendance in the lab class is compulsory.
* Students should not attend a different lab group/section other than the one assigned at the beginning of the session.
* On account of illness or some family problems, if a student misses his/her lab classes, he/she may be assigned a different group to make up the losses in consultation with the concerned faculty / lab instructor. Or he/she may work in the lab during spare/extra hours to complete the experiment. No attendance will be granted for such case**.**
  1. **Preparation and Performance**
* Students should come to the lab thoroughly prepared on the experiments they are assigned to perform on that day. Brief introduction to each experiment with information about selfstudy reference is provided on LMS.
* Students must bring the lab report during each practical class with written records of thelast experiments performed complete in all respect.
* Each student is required to write a complete report of the experiment he has performed and bring to lab class for evaluation in the next working lab. Sufficient space in work book is provided for independent writing of theory, observation, calculation and conclusion.
* Students should follow the Zero tolerance policy for copying / plagiarism. Zero marks will be awarded if found copied. If caught further, it will lead to disciplinary action.
* Refer **Annexure 1** for Lab Report Format

1. **LIST OF EXPERIMENTS**

|  |  |  |
| --- | --- | --- |
| Exp. No. | Division of Experiments | List of Experiments |
| 1 | Basics of Linux | Explain the structure of Linux Operating System |
| 2 | Installation of Ubuntu Operating system |
| 1 | Shell Programs | Write a shell program to find factorial of a number. |
| 2 | Write a shell program to find gross salary of an employee. |
| 3 | Write a shell program to display the menu and execute instructions accordingly  (i)List of file (ii)Process Status (iii) Date (iv) users in program (v) Quit |
| 4 | Write a shell program to find Fibonacci series. |
| 5 | Write a shell program to find largest of three numbers. |
| 6 | Write a shell program to find average of N numbers |
| 7 | CPUSchedulingAlgorithms | Write a C program to simulate the following non-preemptive CPU scheduling algorithms to find turnaround time and waiting time.  a) FCFS b) SJF c) Round Robin (pre-emptive) d) Priority |
| 8 | \*Write a C program to simulate multi-level queue scheduling algorithm considering the following scenario. All the processes in the system are divided into two categories – system processes and user processes. System processes are to be given higher priority than user processes. Use FCFS scheduling for the processes in each queue. |
| Implement the following CPU scheduling Algorithms.  i) Round Robin  ii) Priority Based |
| 9 | DeadlockManagement  Technique | Write a program to simulate Bankers algorithm for the purpose ofdeadlock avoidance |
| 10 | PageReplacement  Algorithms | Write a C program to simulate page replacement algorithms  a) FIFO b) LRU c) LFU |
| 11 | Write a C program to simulate page replacement algorithms  a) Optimal |

1. **LIST OF FLIP EXPERIMENTS**
2. Execute the**who** command written in a file to instruct the shell to read input from a file called "myfile1" instead of from the keyboard. Usethe **more** command to see the contents of myfile1.
3. Use the date and who commands in sequence (in one line) such that theoutput of date will display on the screen and the output of who will beredirected to a file called myfile2. Use the more command to check thecontents of myfile2
4. Write a sed command that swaps the first and second words in each linein a file.
5. Write a shell script that takes a command –line argument and reports on whether it is directory, a file, or something else.
6. Write a shell script that accepts one or more file name as arguments andconverts all of them to uppercase, provided they exist in the current directory.
7. Write a shell script that determines the period for which a specified useris working on the system.
8. Write a shell script that accepts a file name, starting and ending linenumbers as arguments and displays all the lines between the given linenumbers.
9. Write a shell script that deletes all lines containing a specified word in oneor more files supplied as arguments to it.
10. Write a shell script to perform the following string operations:
    * 1. To extract a sub-string from a given string
      2. To find the length of a given string
11. **LIST OF PROJECTS**
    * + 1. Case Study of Window OS
        2. Case Study of Linux OS
        3. Case Study of MAC OS
12. **RUBRICS**

|  |  |
| --- | --- |
| **Marks Distribution** | |
| **Continuous Evaluation(50 Marks)** | **End Semester Exam (20 Marks)** |
| Each experiment shall be evaluated for 10 marks and at the end of the semester proportional marks shall be awarded out of 50. | End semester practical evaluationincluding Mini project (if any)carries 20 marks. |
| Following is the breakup of 10 marks for each  **4 Marks**: Observation & conduct of experiment. Teacher may ask questions about experiment.  **3 Marks:** For report writing  **3 Marks:** For the 15 minutes quiz to be conducted in every lab. |

**Annexure 1**

**Operating Systems**

**(CSL 303)**

Lab Practical Report



Student: Aditya

Roll No.: 18csu011

Semester: 5th

Group: A1

Department of Computer Science and Engineering

NorthCap University, Gurugram- 122001, India

Session 2020-21

**INDEX**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S.No** | **Experiment** | **Page No.** | **Date of Experiment** | **Date of Submission** | **Marks** | **CO Covered** | **Signature** |
| 1. | Installation of cygwin | 18-28 | 1 sept 2020 |  |  |  |  |
| **2** | Write the Basic of Linux command with screenshot. | 29-53 |  |  |  |  |  |
| **3.** | Write the following shell programming.   1. Write a shell program to Hello world. 2. Write a shell program to find factorial of a number. 3. Write a shell program to find gross salary of an employee 4. Write a shell program to find Fibonacci series. 5. Write a shell program to find largest of three numbers 6. Write a shell program to find average of N numbers. | 54-64 |  |  |  |  |  |
| **4.** | Implement the following CPU scheduling Algorithms.   1. FCFS(with arrival time and without arrival time) | 65-71 |  |  |  |  |  |
| **5.** | Implement the i) SJF(with preemptive approach and non-preemptive approach) | 72-75 |  |  |  |  |  |
| **6.** | Implement the Priority Scheduling | 76-79 |  |  |  |  |  |
| **7.** | Implement the Round Robin Scheduling. | 80-81 |  |  |  |  |  |
| **8.** | Implement the Reader-Writer Problem**.** | 82-84 |  |  |  |  |  |
| **9.** | Implement the Banker’s Algorithm. | 85-90 |  |  |  |  |  |

**EXPERIMENT:-1**

Semester /Section: Vth / A

Link to Code:

Date: 6 august 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about Installation of cygwin.

**Program Outcome**

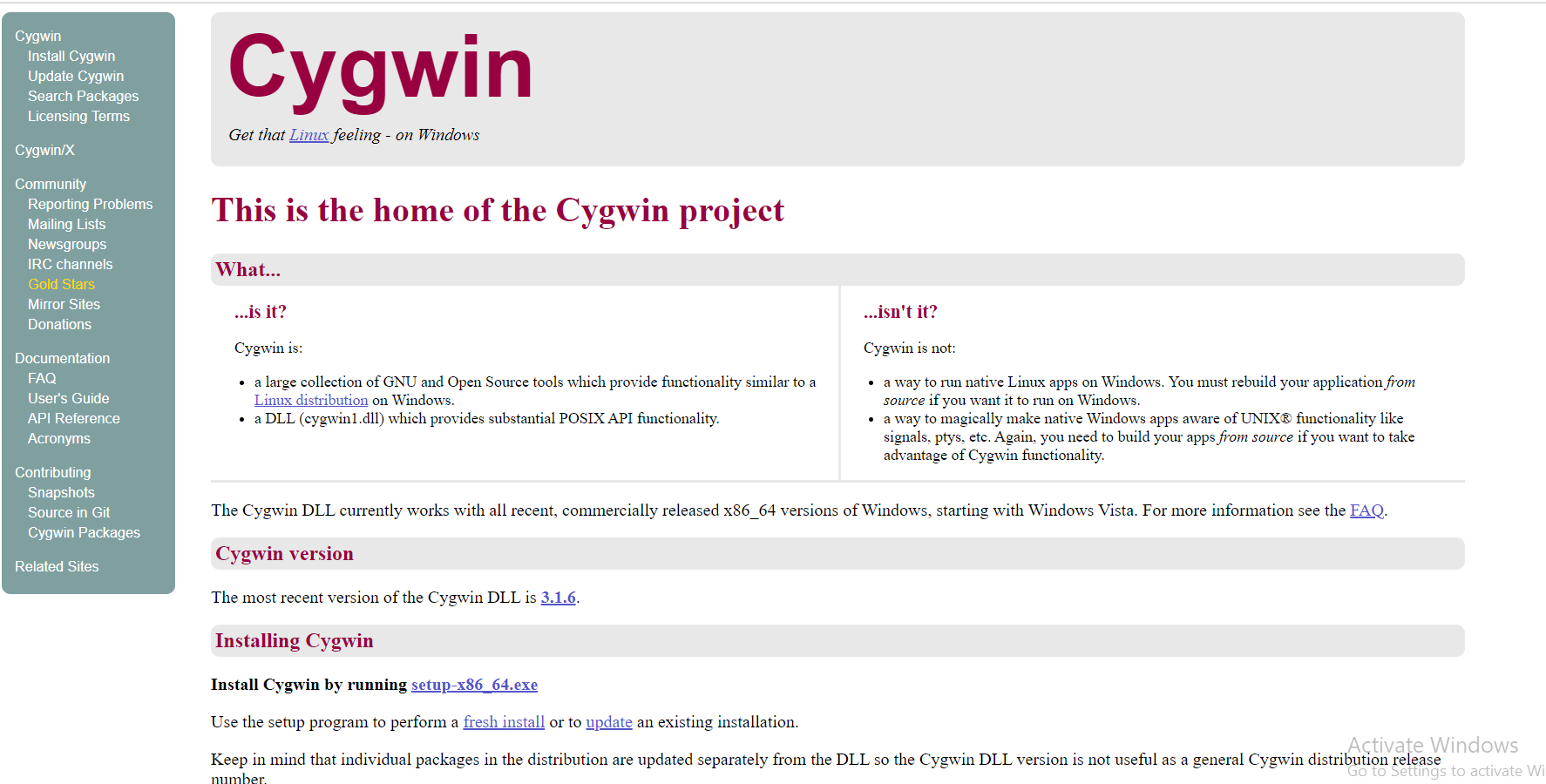
* The students will understand the installation of cygwin.

**Problem Statement**

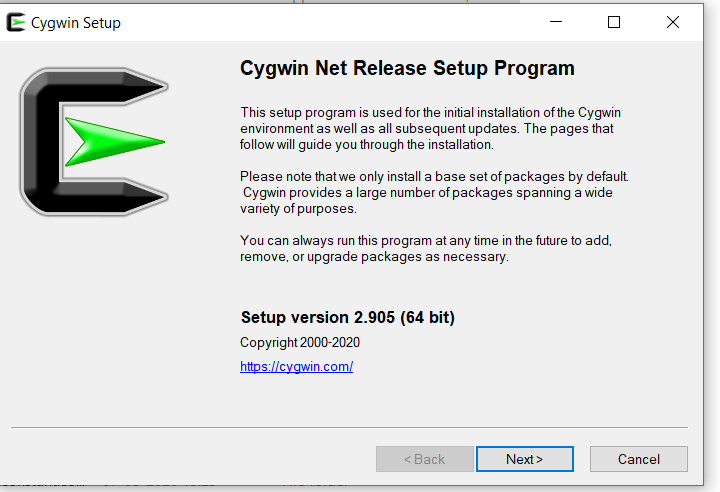
Installation of cygwin.

**Cygwin:-** Cygwin is open source tool that allow unix or linux application to be complied.

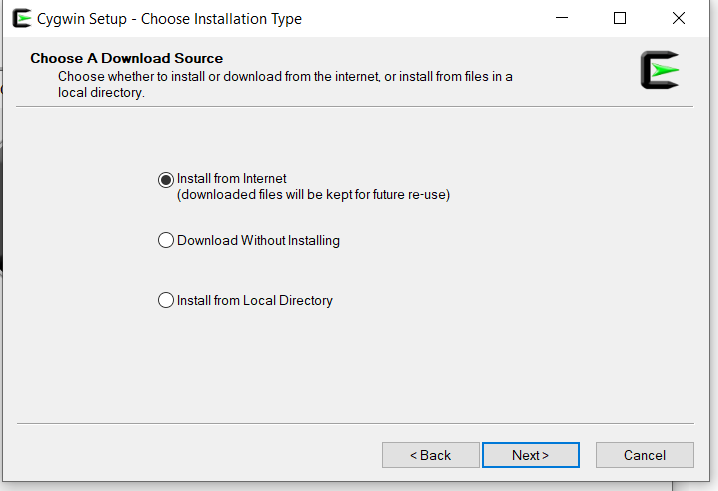
**Step 1:- Go to link: <https://www.cygwin.com/>**

****

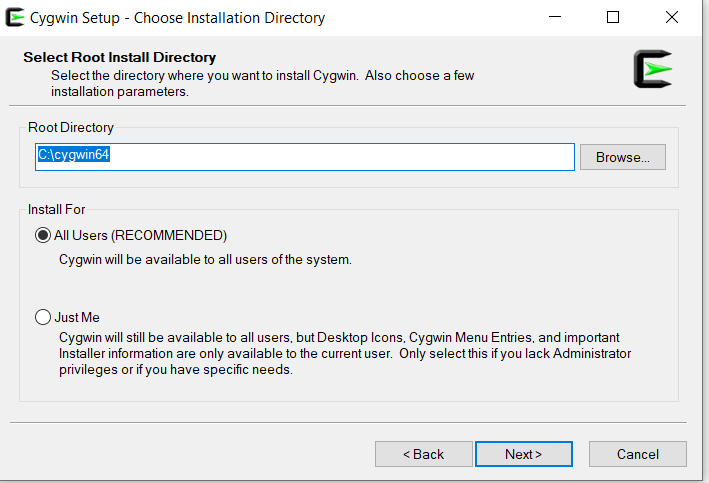
**Step 2:-**Accept all defaults, clicking "Next" until you get to "Choose a Download Site."

****

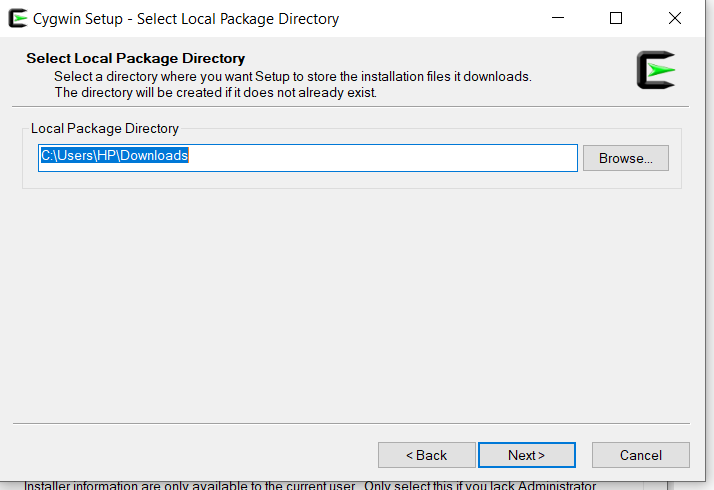
**Step3:** Now you will see three options. Choose install from internet option and click next to continue.

****

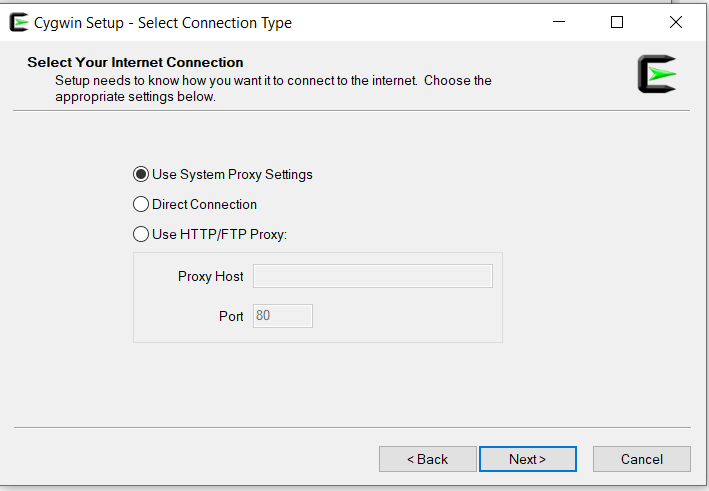
**Step 4:-** Now you need to select your root directory. You can change it according to your choice or keep it as default. In the install for option, select just me and hit next.

****

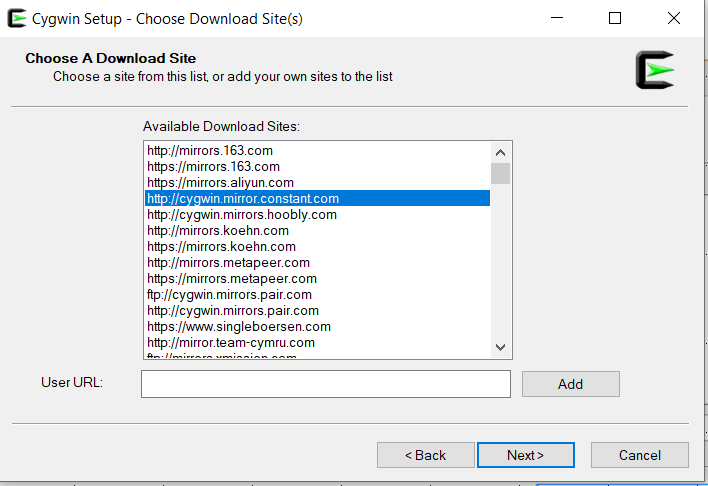
Step 5: Now browse to a directory for local packages. After selecting appropriate directory hit next.

****

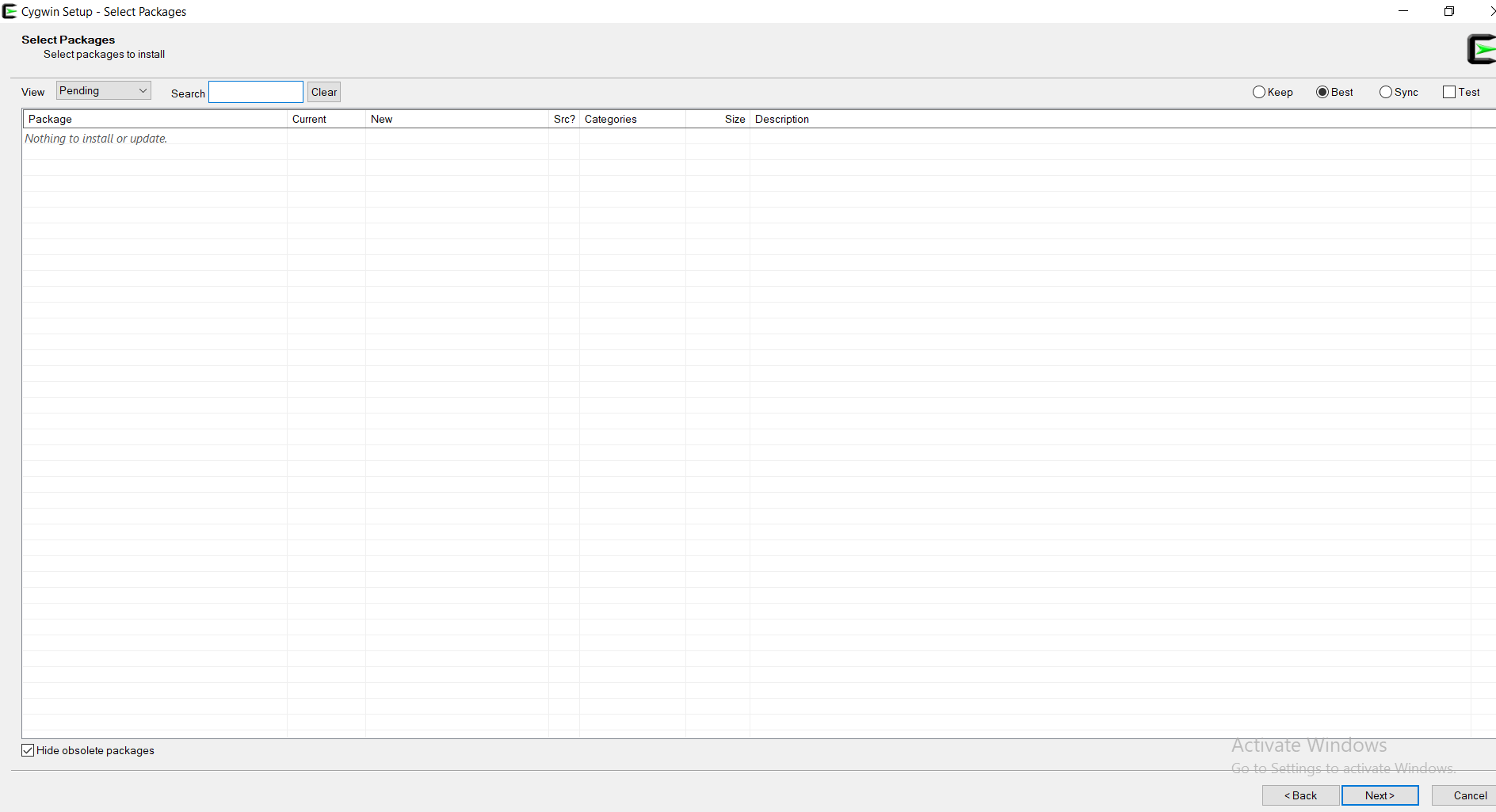
**Step 6:-** Now select your connection type. In my case I am keeping it as default.

****

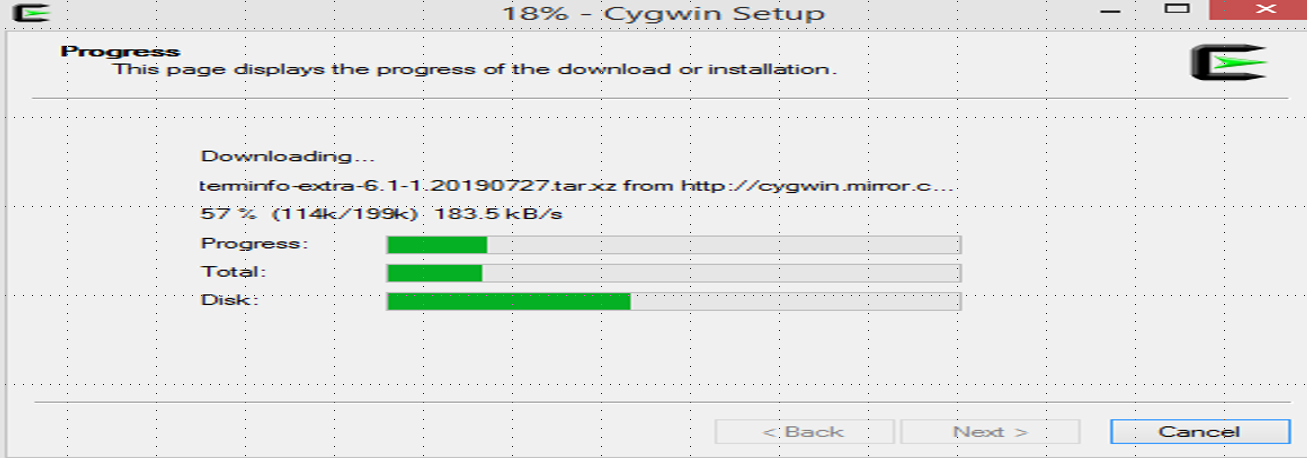
**Step 7:** Now you have to select site for downloading the packages. Select the third site as in image and click next to continue.

****

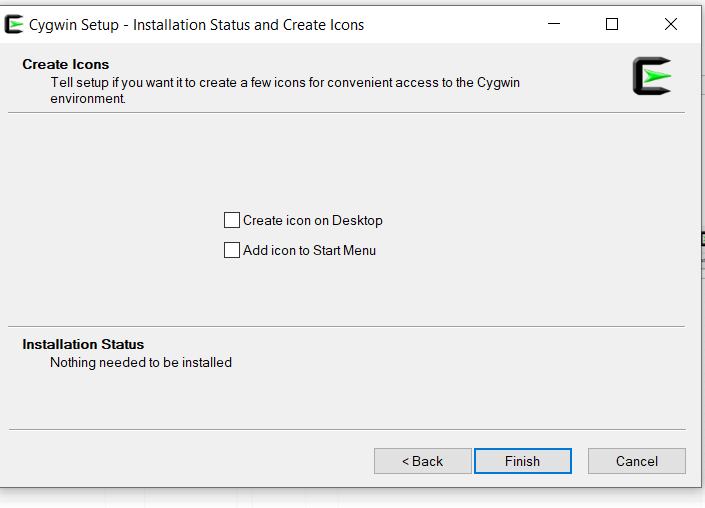
**Step 8:-** You will see a screen like the image given below. You can add additional packages you want to download or make no changes and click on next.

****

**Step 9:-** Installation will begin, it will take around 2 minutes. You can see the progress on the top where it is showing currently 57% - Cygwin setup

****

**Step10:-** After the above steps you will see options for icon on desktop and icon to start menu. Its your choice if you want icon on Desktop or start menu or in both places. Just click finish and your installation is complete

****

**EXPERIMENT -2**

Semester /Section: Vth / A

Link to Code: CSL 303

Date: 6 august 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about Basic of Linux command

**Program Outcome**

* The students will understand the Basic of Linux command

**Problem Statement**

Write the Basic of Linux command with screenshot.

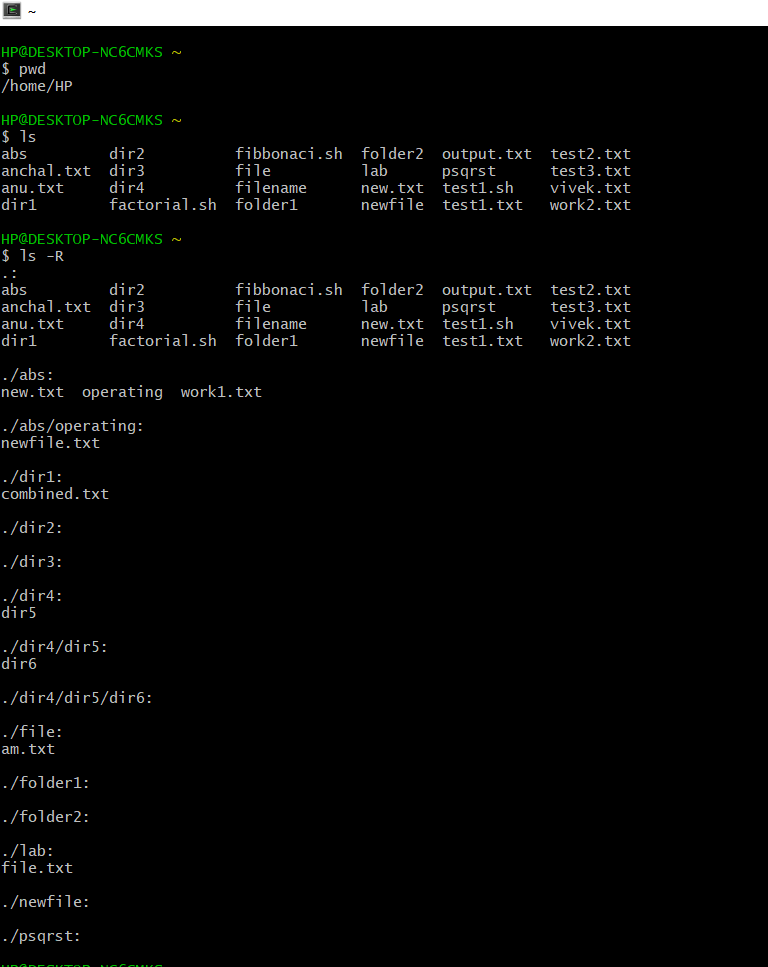
**Background Study**

* Linux is a Unix-Like operating system. All the Linux/Unix commands are run in the terminal provided by the Linux system.
* This terminal is just like the command prompt of Windows OS. Linux/Unix commands are case-sensitive. The terminal can be used to accomplish all Administrative tasks

**Commands:**

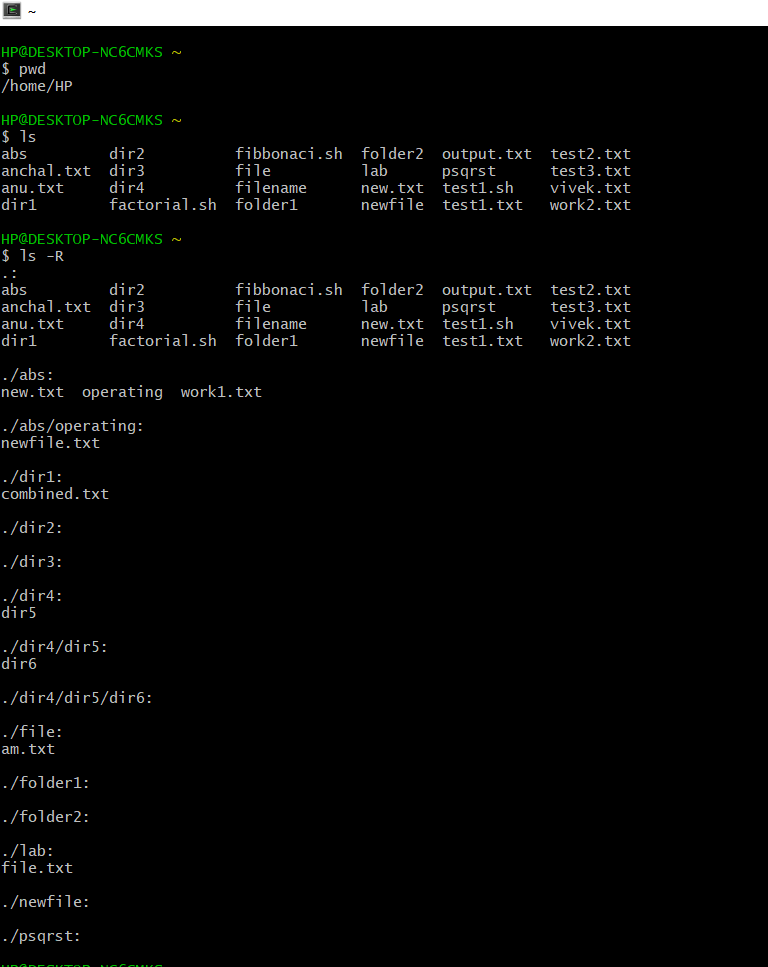
pwd commands:-

1. It give the current directory or absolute path.

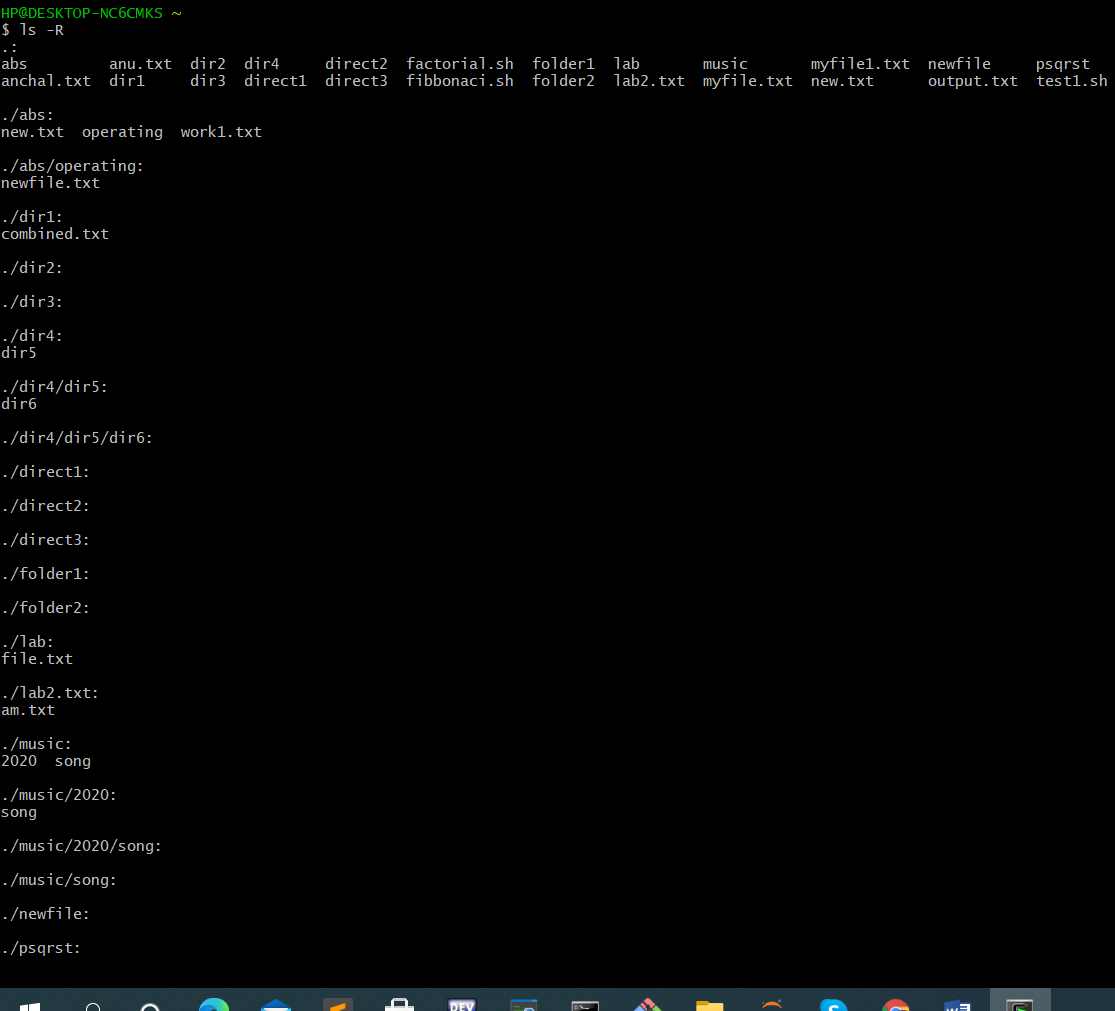


ls commands:-

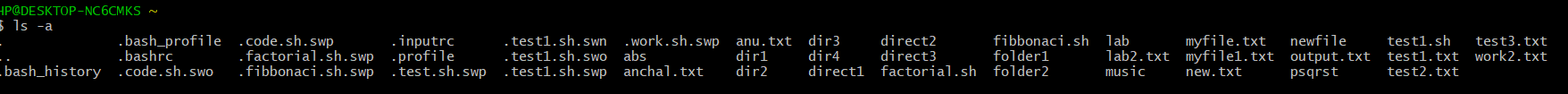
1. **ls:-**This command display the content of directory or file



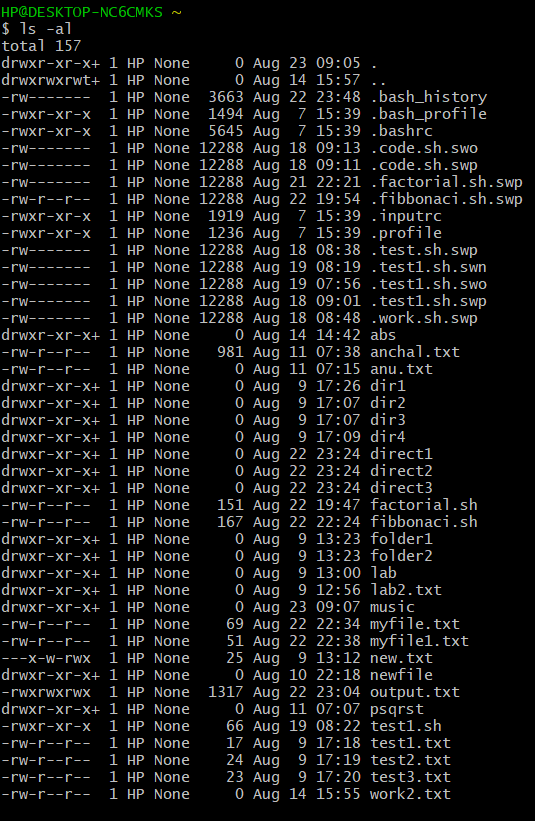
1. **ls –R:-**it will list all the file in the sub-directories as well.



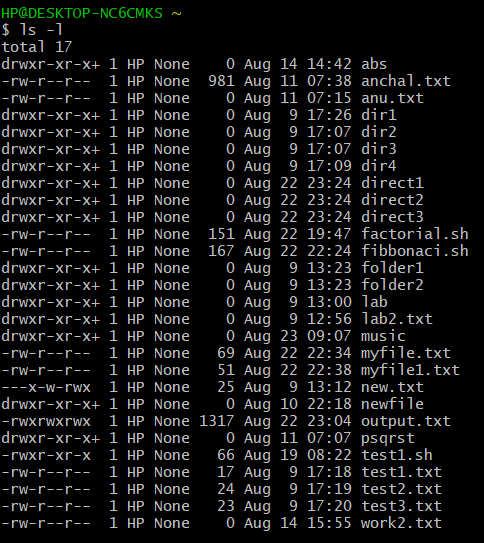
1. **ls –a:-** it will show the hidden file.



1. **ls –al:-**it will list the file and directories with detailed information like the permission,size,owner eetc.

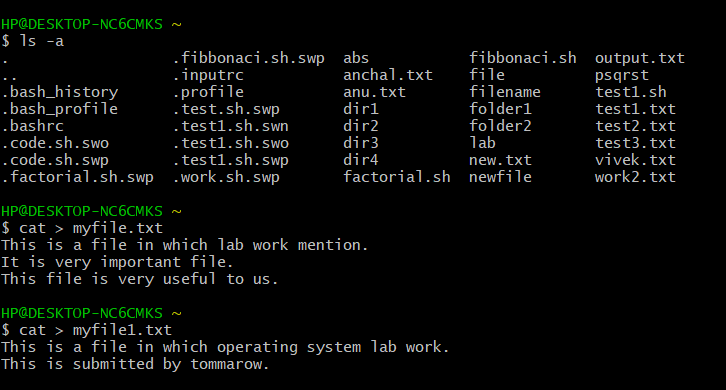


1. **ls –l**:- it give all privileges.

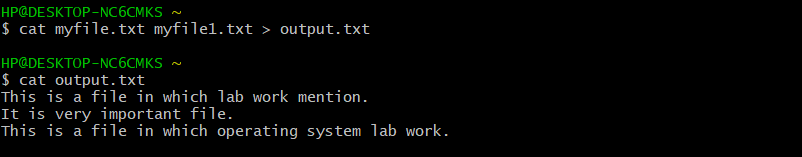


cat commands:-

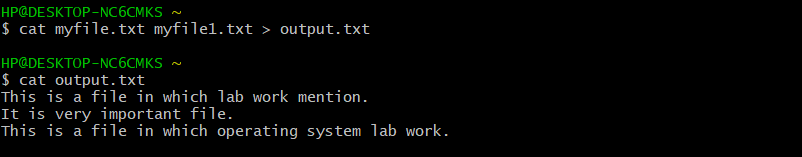
1. **cat > filename** :- Create a new file.



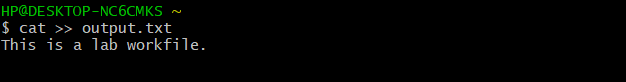
1. **cat file1 file2 >file3** :- it join two file (1and2) and store the output of them in a new file3.



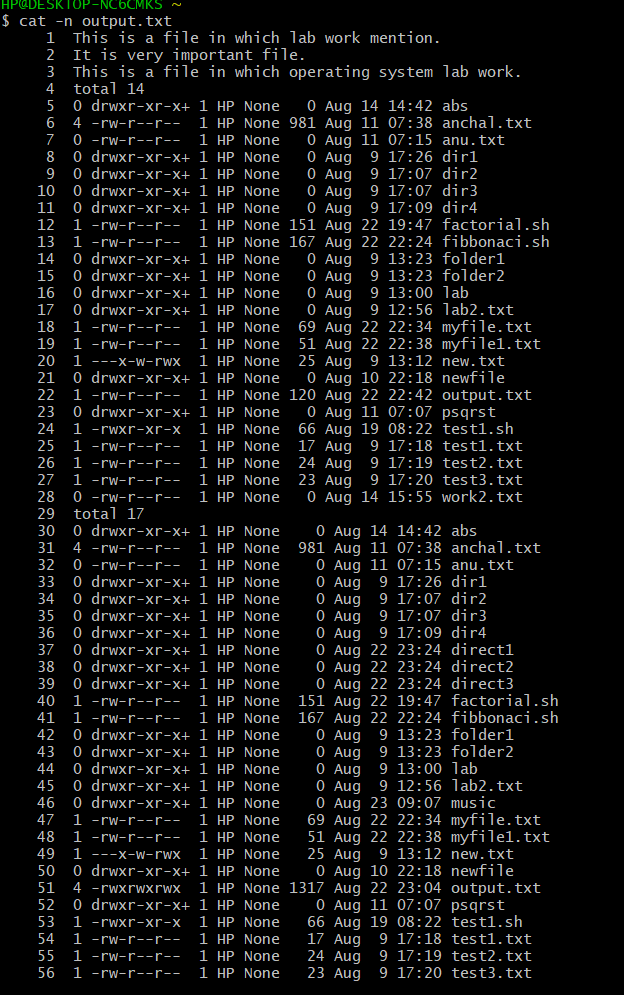
1. **cat filename**:- it Display the file content.



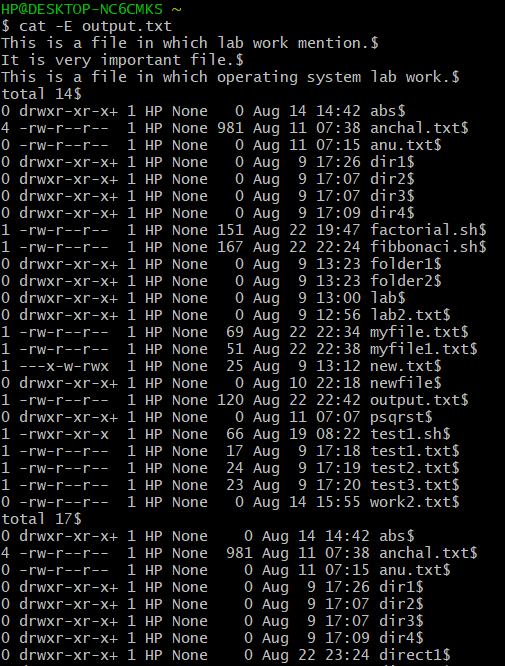
1. **cat >> newfile.txt:-**To append or make change in a file.(And again write you want to write next)



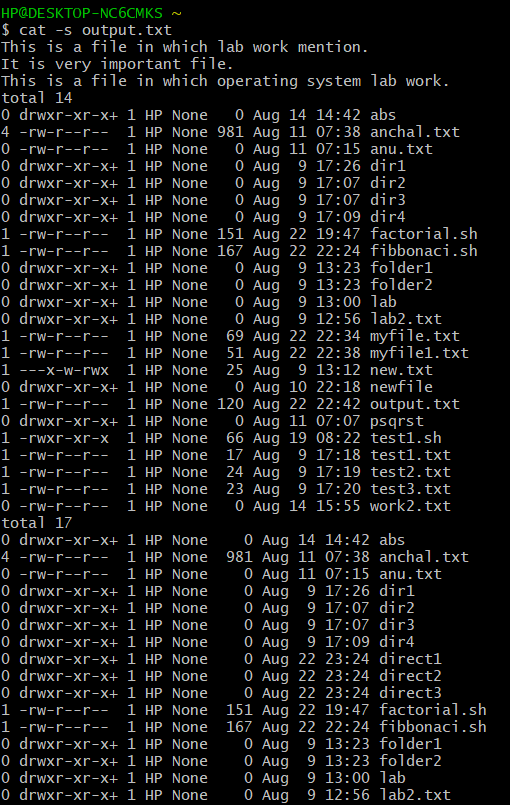
1. **cat –n newfile.txt**:- To show number of each line (even blank line).



1. **cat –E newfile.txt**:- To end each line with a $ symbol.

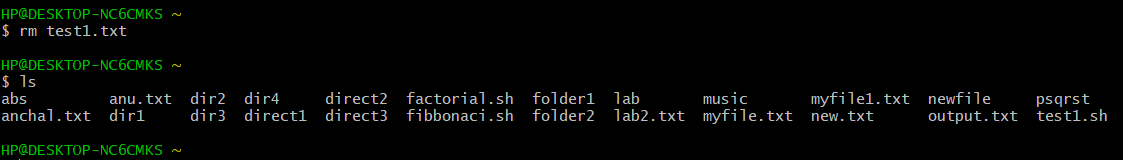


1. **cat –s newfile.txt:-** To squeeze the blank space (multiple space).

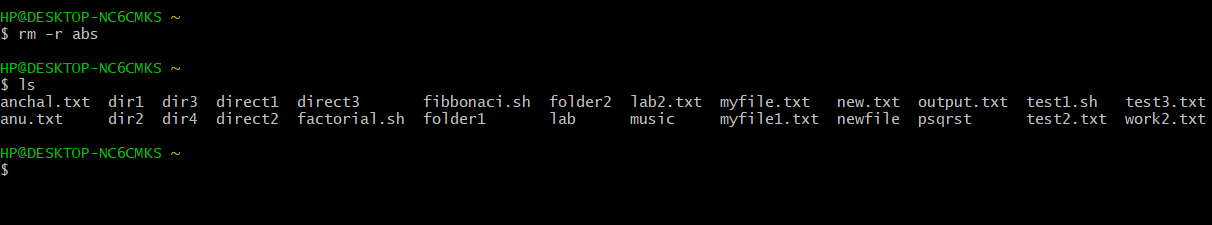


rm commands:-

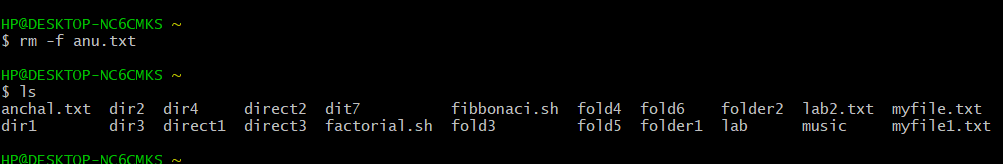
1. Use the **rm** command to delete files and directories.



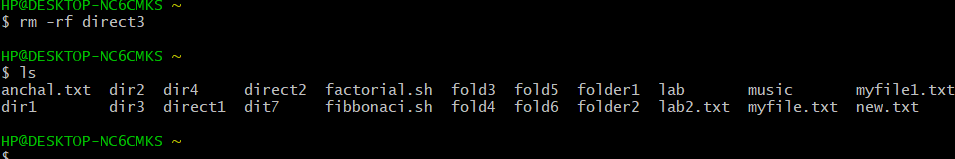
1. **rm –r :-**It delete both the file ad foler it contains when using only the rm commands.



1. rm-f:- file force to remove the file.

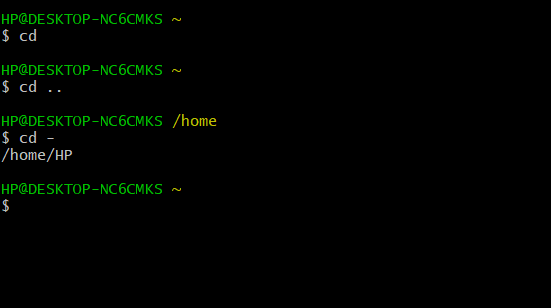


1. rm –rf dir:-force to remove the directory.



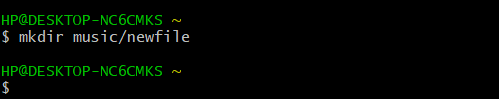
cd commands:-cd command to go straight to the home folder.

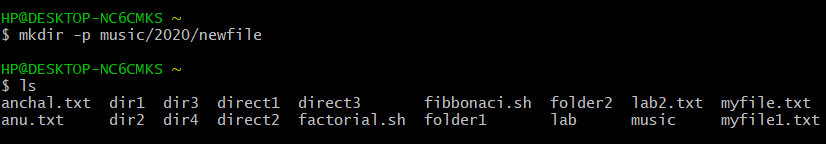
1. **cd .. :-** to move directory up
2. **cd -** :- to move your previous directory.

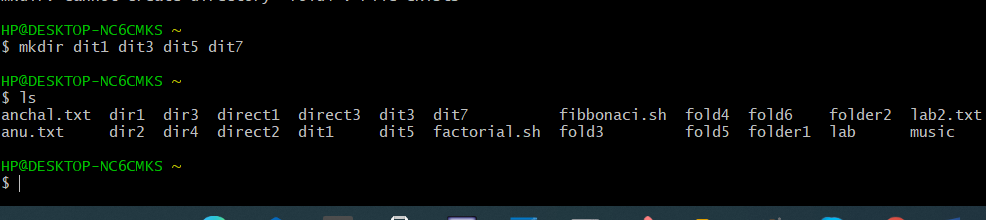


mkdir commands**:-**it use to make a new directory.

1. **mkdir Music/Newfile**:- To generate a new directory using another.

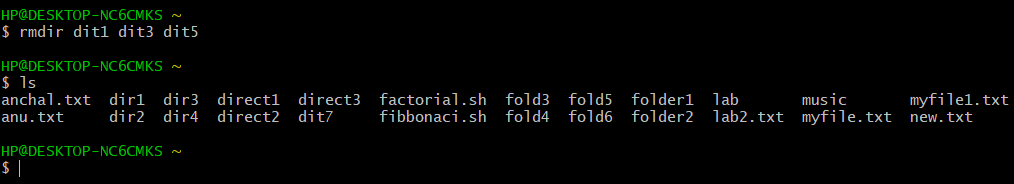


1. **mkdir -p music/2020/Newfile** :- use –p to generate new directory existing between two directory.
2. **mkdir dir1 dir2 dir3 dir4:-** it use to create a different - different directory.



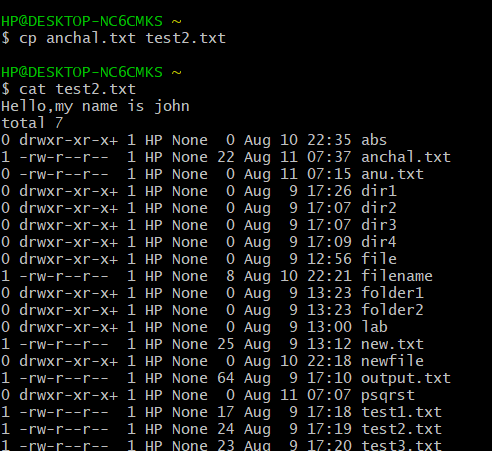
rmdir commands:-

1. **rmdir dir1:-** it use to delete only empty file.

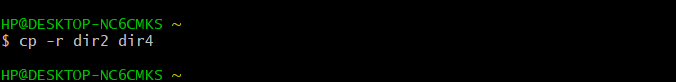
****

**cp commands:-**it use to copy file from the one file to file.

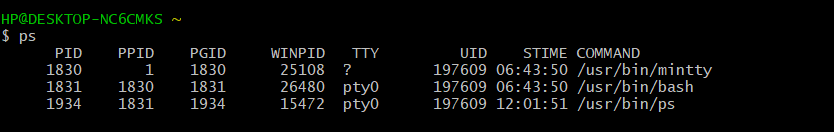
1. Cp :- Copy the content of file1 to file2.



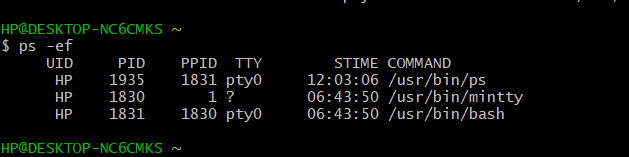
1. Cp –r dir1 dir2:- Copy dir1 to dir2;create dir2 if not present.



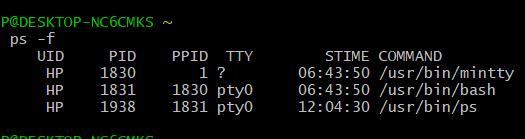
ps commands:- it display the process of current shell.



1. ps –ef:- To perform a full format listing.

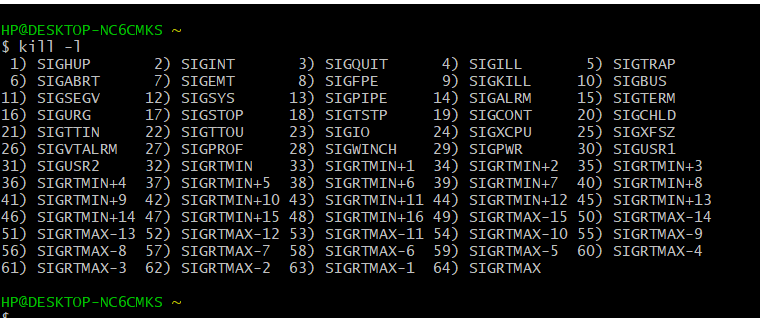


1. ps –f:-show process userids,ppids.

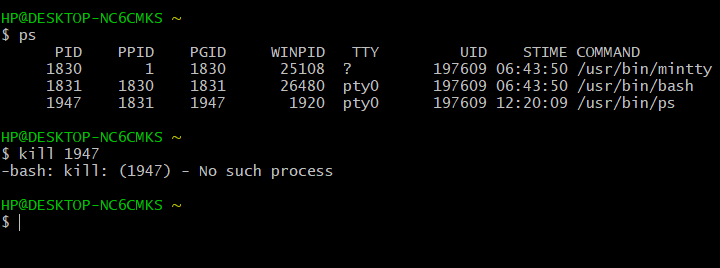


**Kill commands:-**To display all the available signals you can use below command option.

1. **kill –l:-**To display all the available signals you can use below command option.



**29.kill pid:-** To show how to use a PID with the kill command.



chmod commands:-To change access permissions, change mode.

1. chmod 777 sample.txt:-Allow everyone to read, write, and execute file.

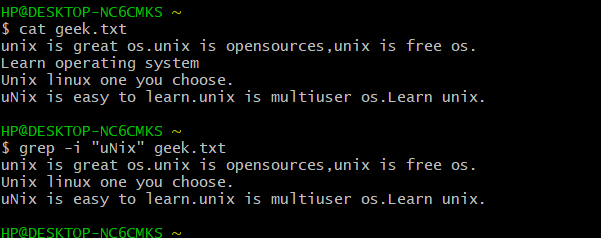


1. chmod a-x output.txt:- Deny execute permission to everyone.

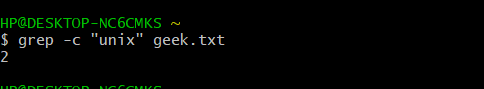


grep commands:-it will search the filter a file for a particular pattern of character and display all the line that contain pattern.

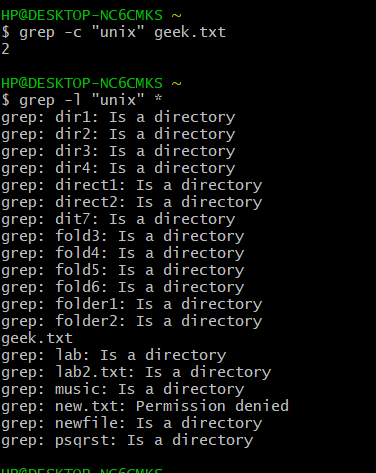
1. **grep –i case insensitive search: it enable to search for a string case insensitively in the given file.**(grep –i “uNix” geek.txt)



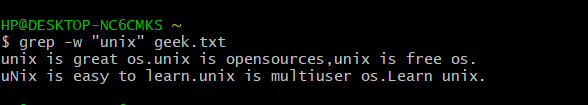
1. **grep –c :-Displaying the count of number of matches (grep –c “unix” geek.txt)**



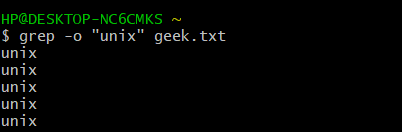
1. **grep –l:- Display the file name that matches the pattern(grep –l “unix” geek.txt)**



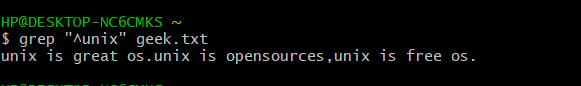
1. **grep –w:-** Checking for whole word in a file.

****

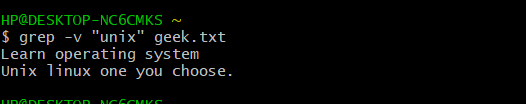
1. **grep –o** :- Displaying only the matching pattern**.**



1. **grep ^:-Matching the line start with string.**

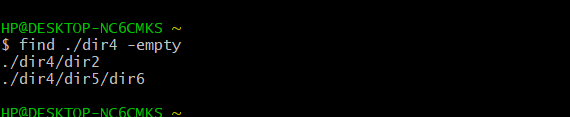


1. **grep –v: It will inverting the pattern match.**

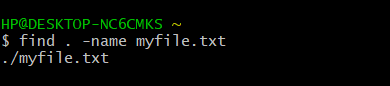


Find commands:-   It can be used to find file and directory and perform subsequent on them.

1. **Search for empty files and directories (find ./dir4 -empty)**



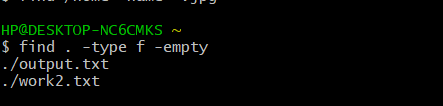
1. Find a file called myfile.txt in current and sub directories.



1. Find all .jpg file in the home and the sub-directories.

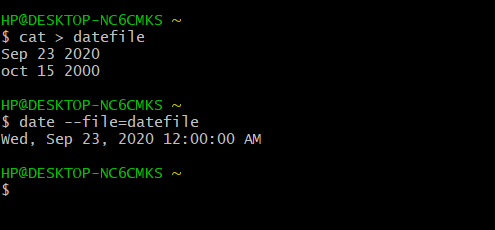


1. Find an empty file within current directory.

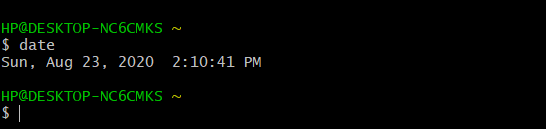


Date commands:-Date command is used to display the date in several format. It also allow to set time and date in a system.

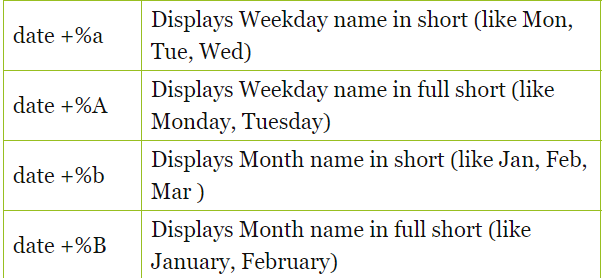
1. Read the date pattern from a file using- file option.

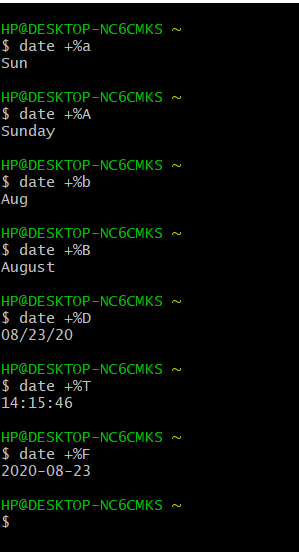


1. It will show current date and time.



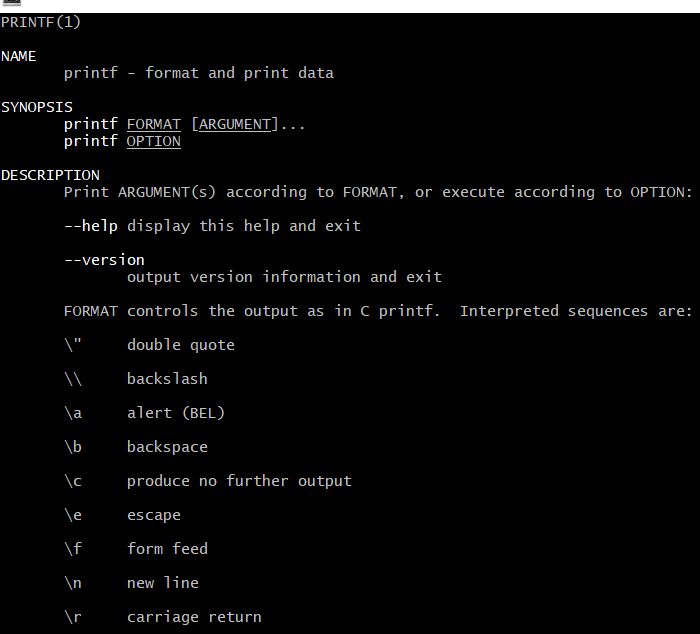
1. Various date commands:-





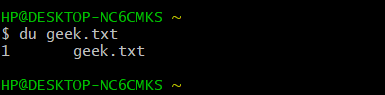
man command

1. It display the whole manual of the command.

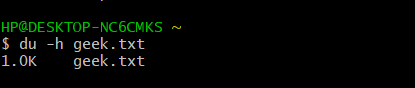


du commands:-

1. This command check how much space a file or directory takes.

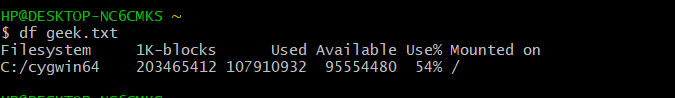


1. du –h :-It show size of file/directory in Kb,Mb.

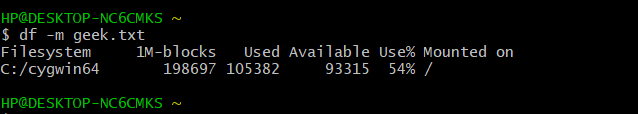
****

df commands:-

1. Use the df command to see the available disk space in each partition in system.

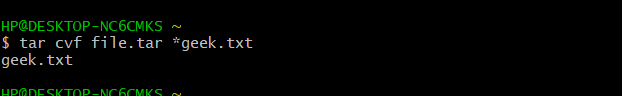


1. df –m:-it uses to show megabyte in available disk space.

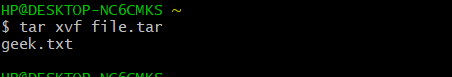


Tar commands:-

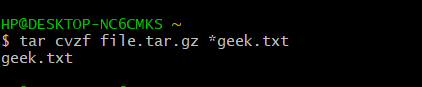
1. **This command create a tar file called file.tar**



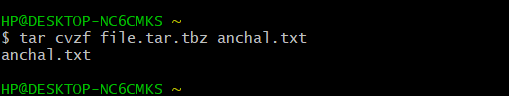
1. **This command extract file from Archives.**



1. This command create a tar file called file.tar.gz

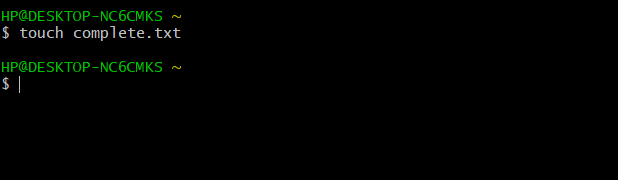


1. **This command from extract file from tar archived .tar**.gz files.

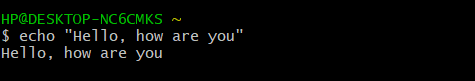


touch commands:-

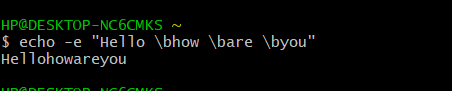
The touch command use to create a file.



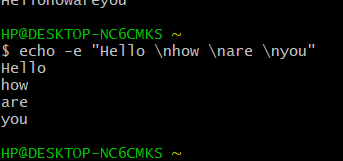
echo commands:-echo commanduse to display line of text/string that are passed as arguments.



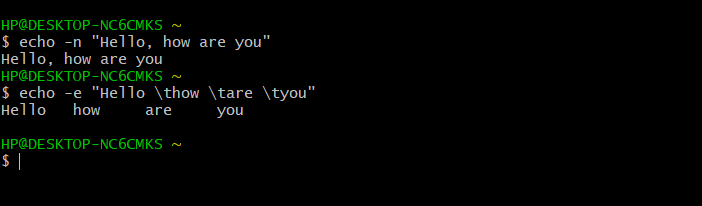
\b:- it will remove all the space between the text.



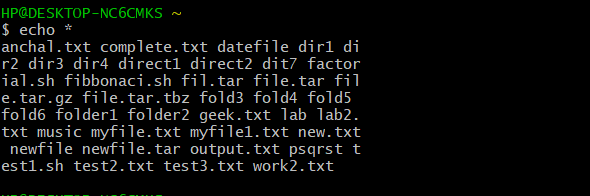
1. **\n: The option will create new line from where it used.**



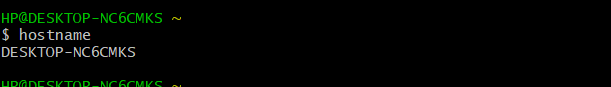
1. **\t:- this will create horizontal tab spaces.**.



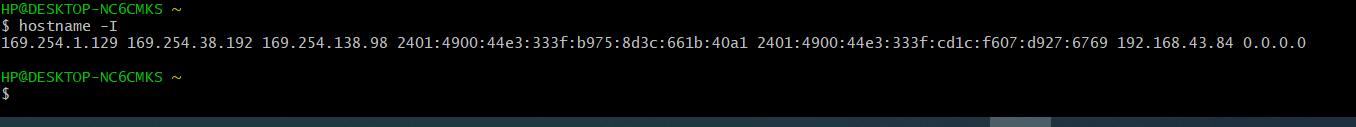
1. **echo \*:- The command will print all file and folder,similar to ls command.**



Hostname commands:-it is use to show your network/host name.

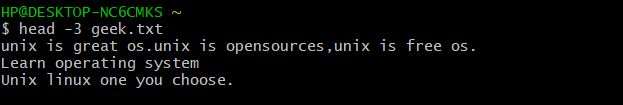


1. hostname –I:- It will display IP address of network.

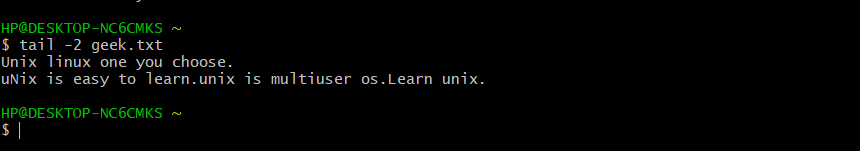


head commands:-it is used to view the starting line of text.

head -3 geek.txt



1. tail commands:-it is used to view the last some line of text.



**EXPERIMENT – 3**

Semester /Section: Vth / A

Link to Code: CSL 303

Date: 6 august 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about shell programming.

**Program Outcome**

* The students will understand the shell programming

**Problem Statement**

Write the following shell programming.

1. Write a shell program to Hello world.
2. Write a shell program to find factorial of a number.
3. Write a shell program to find gross salary of an employee
4. Write a shell program to find Fibonacci series.
5. Write a shell program to find largest of three numbers
6. Write a shell program to find average of N numbers.

**Background Study**

* The shell is the operating system's command-line interface (CLI) and interpreter for the set of commands that are used to communicate with the system.
* A shell script is usually created for command sequences in which a user has a need to use repeatedly in order to save time.

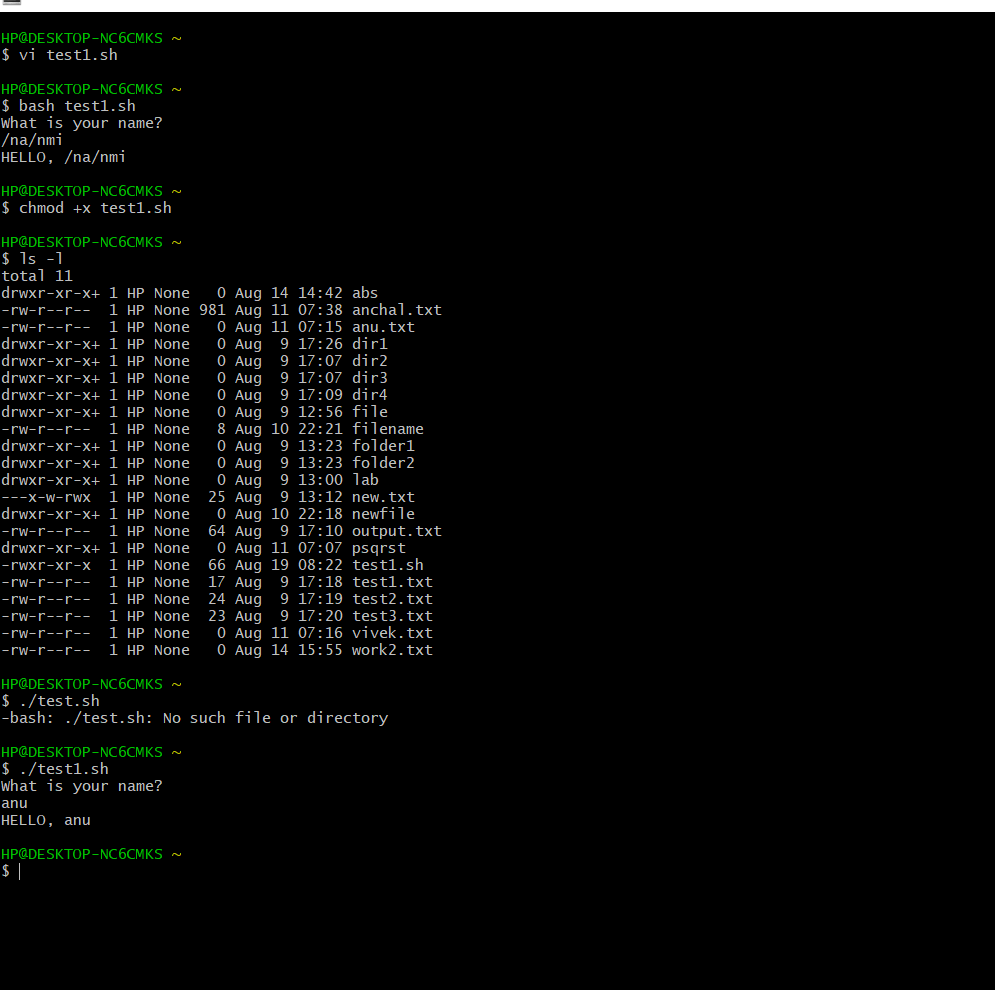
**Program 1: Write a shell program of “Hello World”.**

Step 1:- Create a file (vi test1.sh).

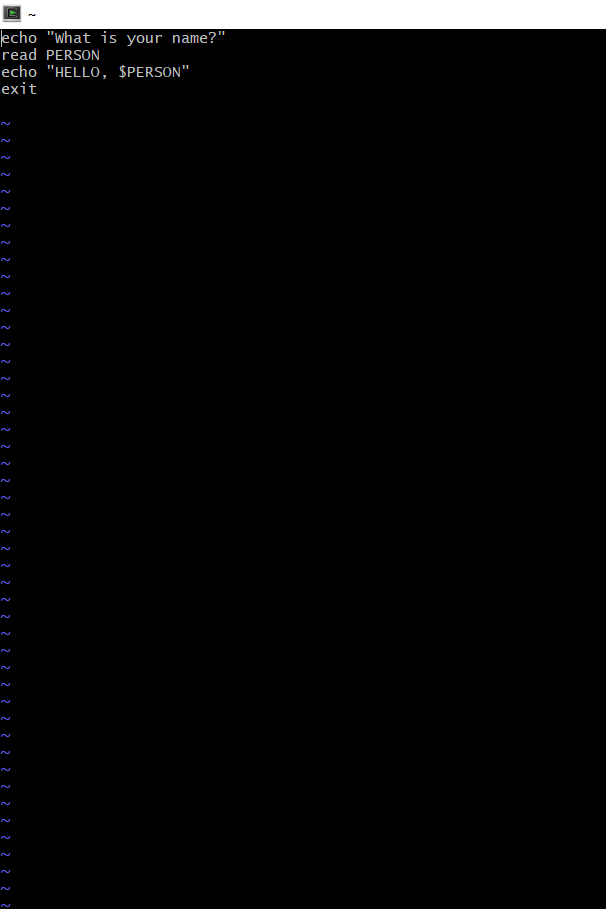
Step 2:- Then write a program in a shell.

Step 3:- Then execute a program with command

(bash test1.sh or ./test1.sh)

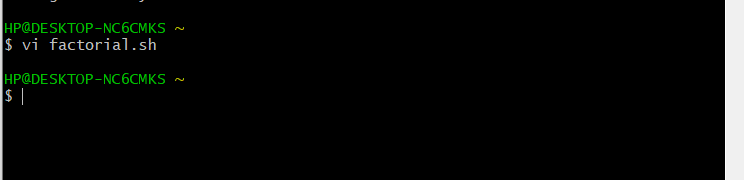


**Programs:**

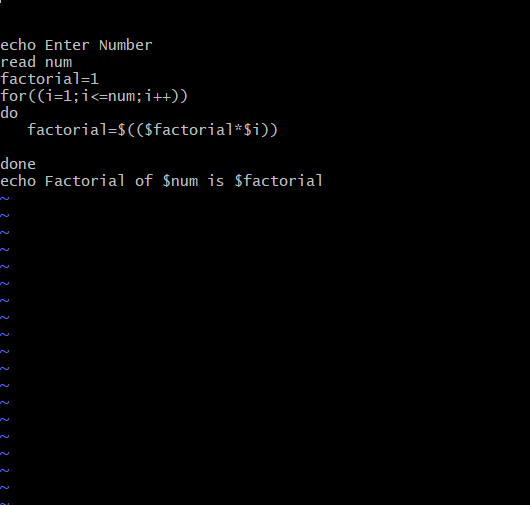


**Program 2: Write a shell programming to find factorial number.**

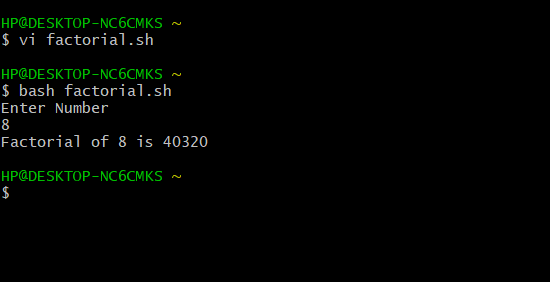
**Step 1:-Create file (factorial.sh)**

****

**Step 2:- Program**



**Output:-**

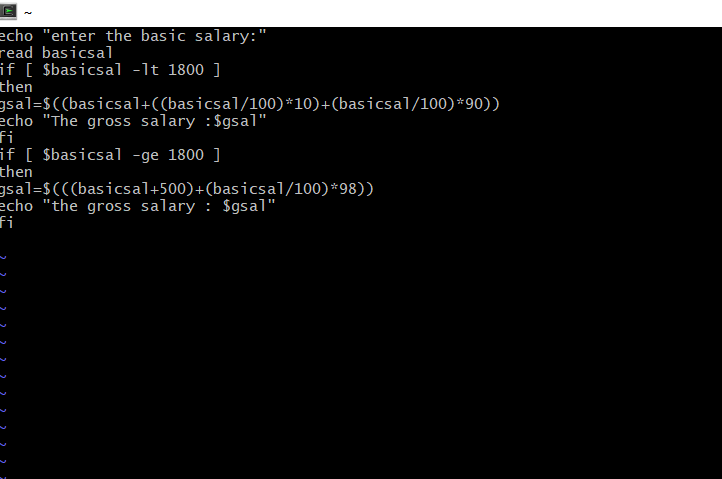


**Program 3:- Write a program to find gross – salary of an employee.**

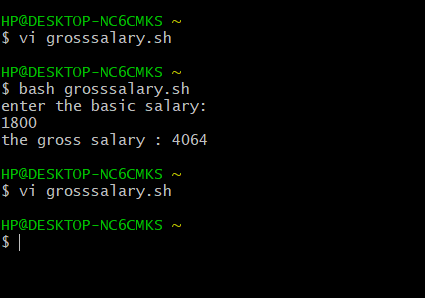
**Step 1: Create a file(grosssalary.sh)**

****

**Step 2:- Program of gross salary.**



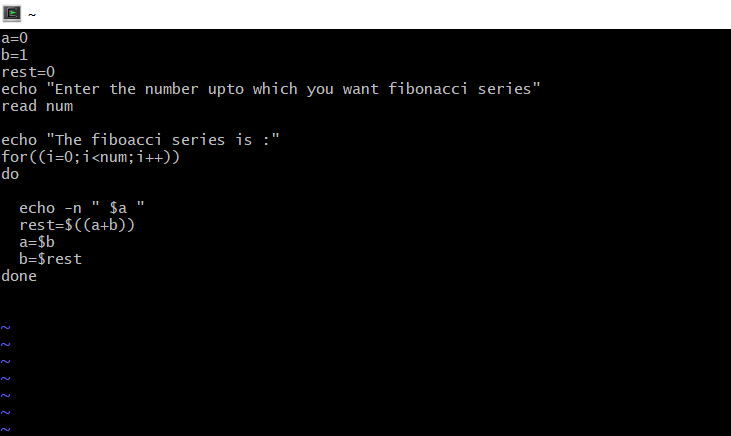
**Output:-**



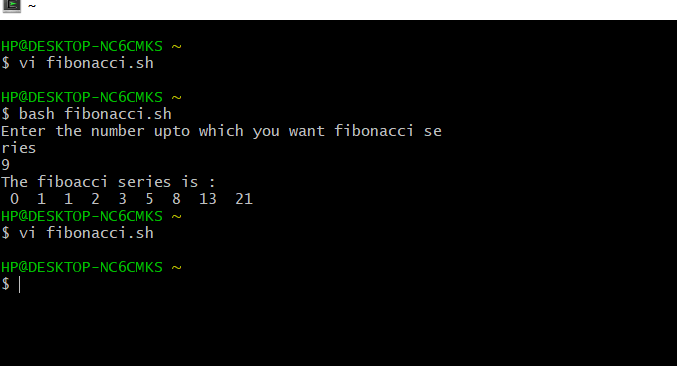
**Program 4: Write a program to find Fibonacci Series.**

**Step 1:- Create a file (fibonacci.sh)**

**Step 2:- Program**

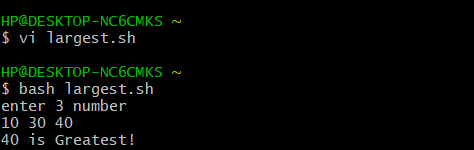


**Step 3:- Output**

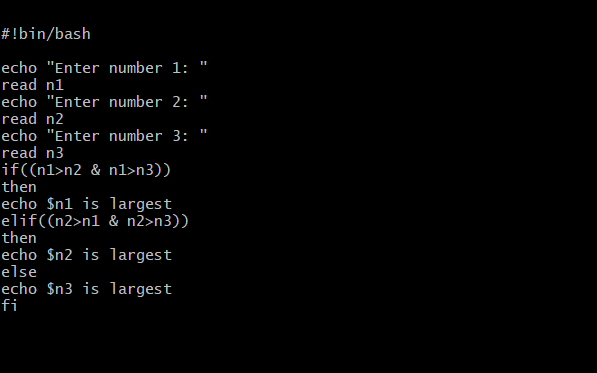


**Program 5: Write a program to find largest three number.**

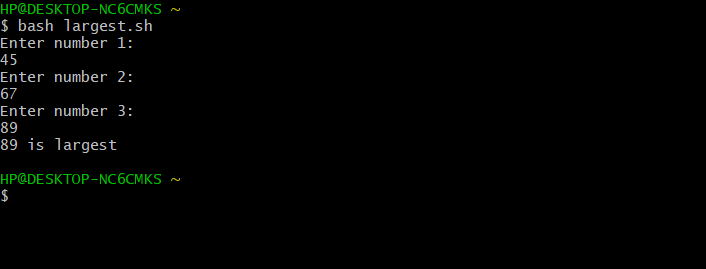
**Step 1: Create file(largest.sh)**



**Step 2: Program**

****

**Step 3:- Output**

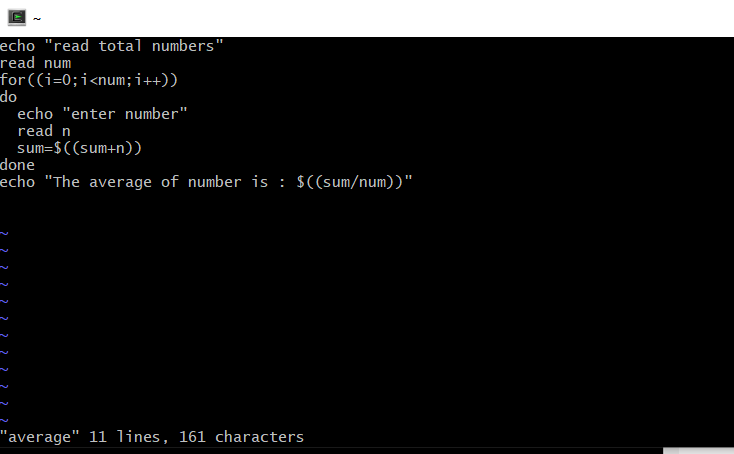


**Program 6:- Write a program to find average of N numbers.**

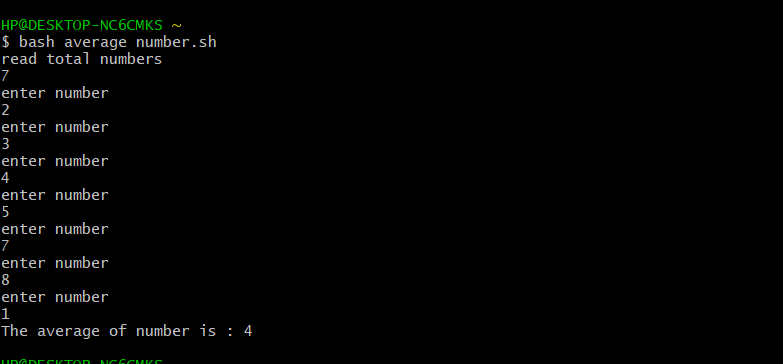
**Step 1:- Create a file(average.sh)**

****

**Step 2: Program**

****

**Output:-**

****

**Experiment No: 4**

Semester /Section: Vth / A

Link to Code:

Date: 1 September 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about CPU scheduling Algorithms.

**Program Outcome**

* The students will understand the First-cum-first-serve and shortest job first algorithm.

**Problem Statement**

Implement the following CPU scheduling Algorithms.

1. FCFS(with arrival time and without arrival time)

**Background Study:**

**FCFS**

* The simplest CPU-scheduling algorithm is the first-come, first-served (FCFS) scheduling algorithm. With this algorithm, processes are assigned the CPU in the order they request it.
* There is a single queue of ready processes.
* The implementation of the FCFS policy is easily managed with a FIFO queue. When a process enters the ready queue, its PCB is linked onto the tail of the queue.
* The average waiting time under the FCFS policy, however, is often quite long.

**FCFS with arrival time**

**Algorithm**

1. Input the process along with brust time [bt] and arrival time [at].
2. Find the waiting time for all other process for

Wt[i] = (bt[0]+bt[1] +……bt[i-1] –at[i])

1. Now find the turnaround time

= waiting time – burst time

1. Then find the average waiting time

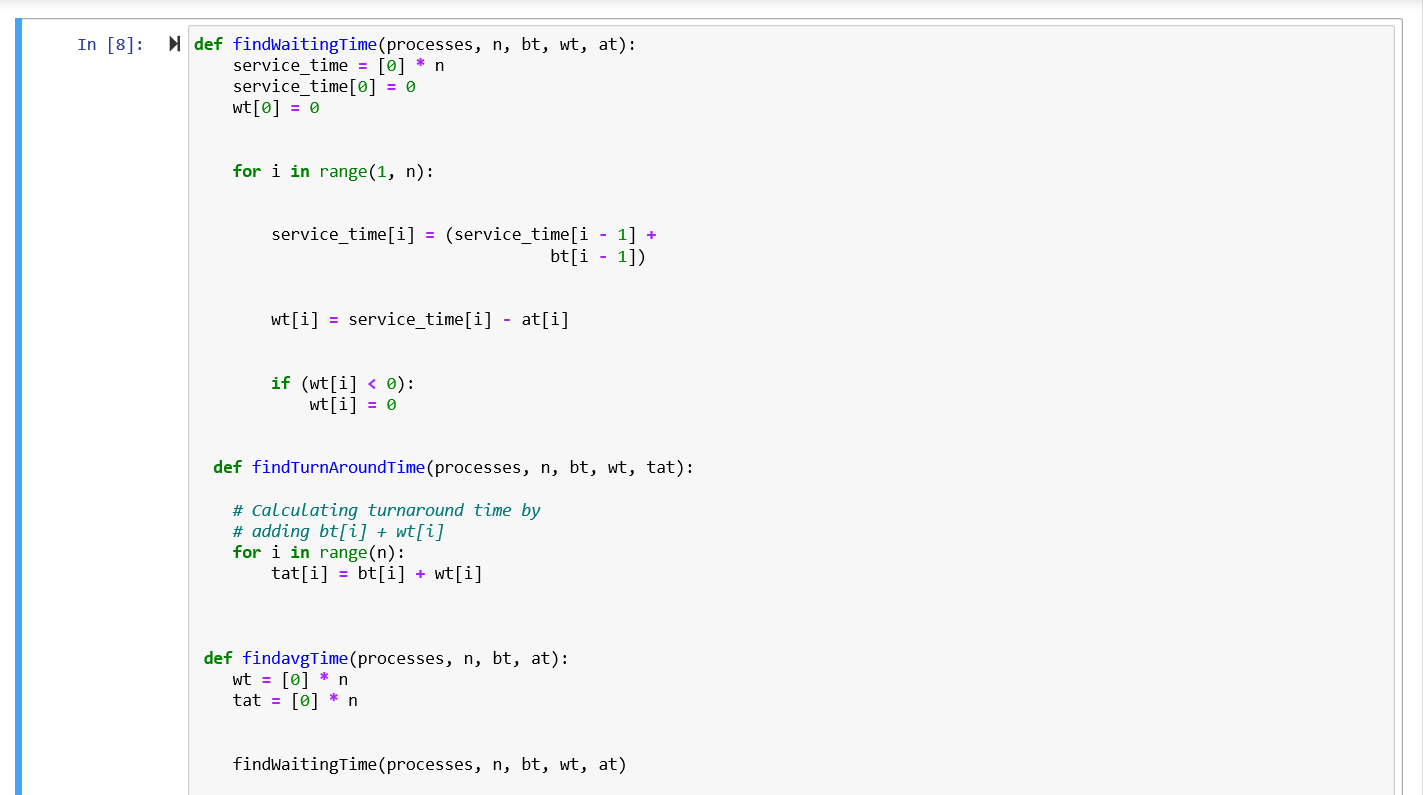
= total waiting time/ no. of process

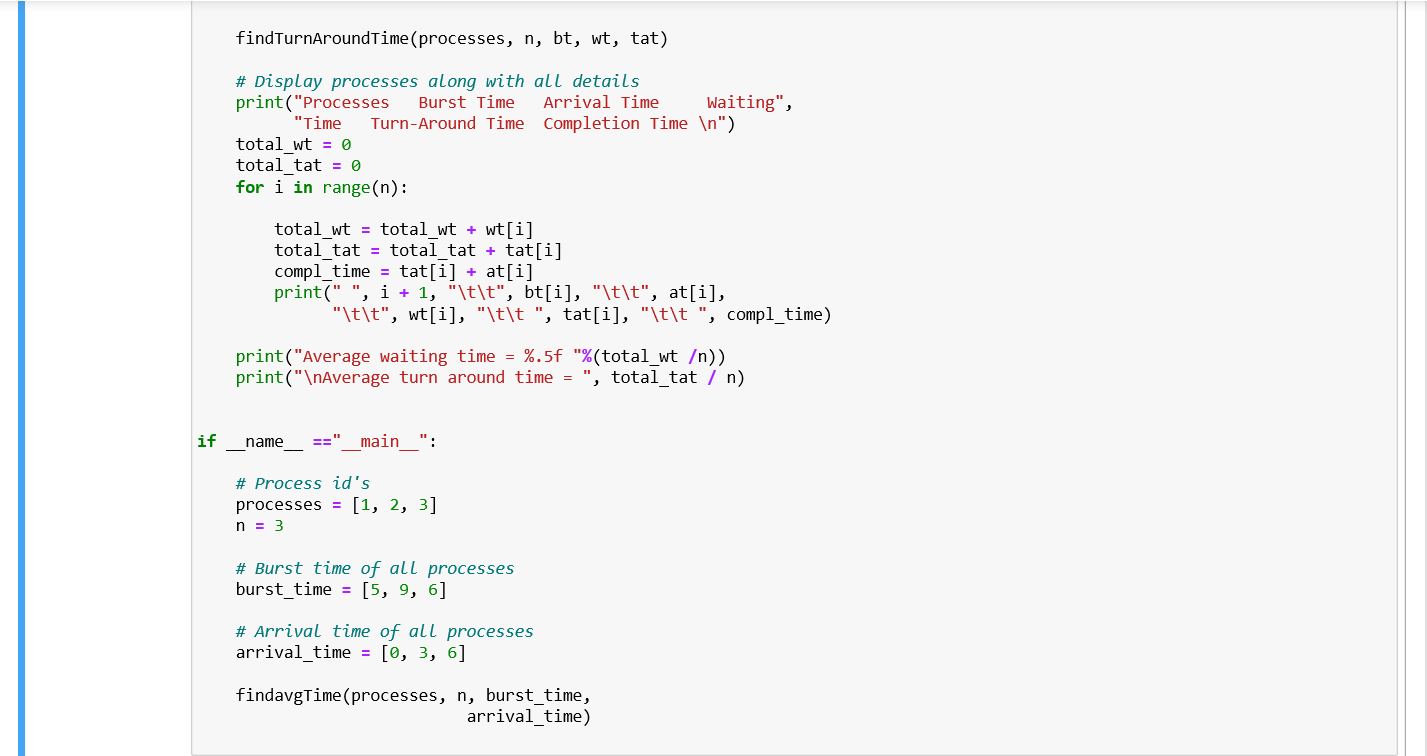
1. Find average turnaround time

=total turn around time/ no.of process

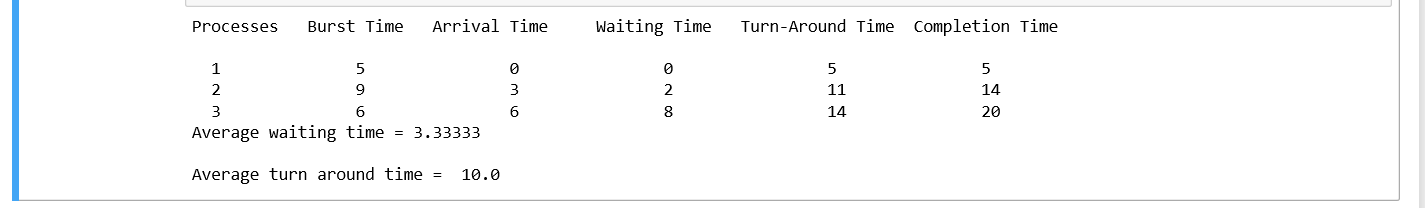
**Code:Screenshot**

**With arrival time :**

****

****

**Output: Screenshhots**

****

**FCFS without arrival time:**

**Algorithm:**

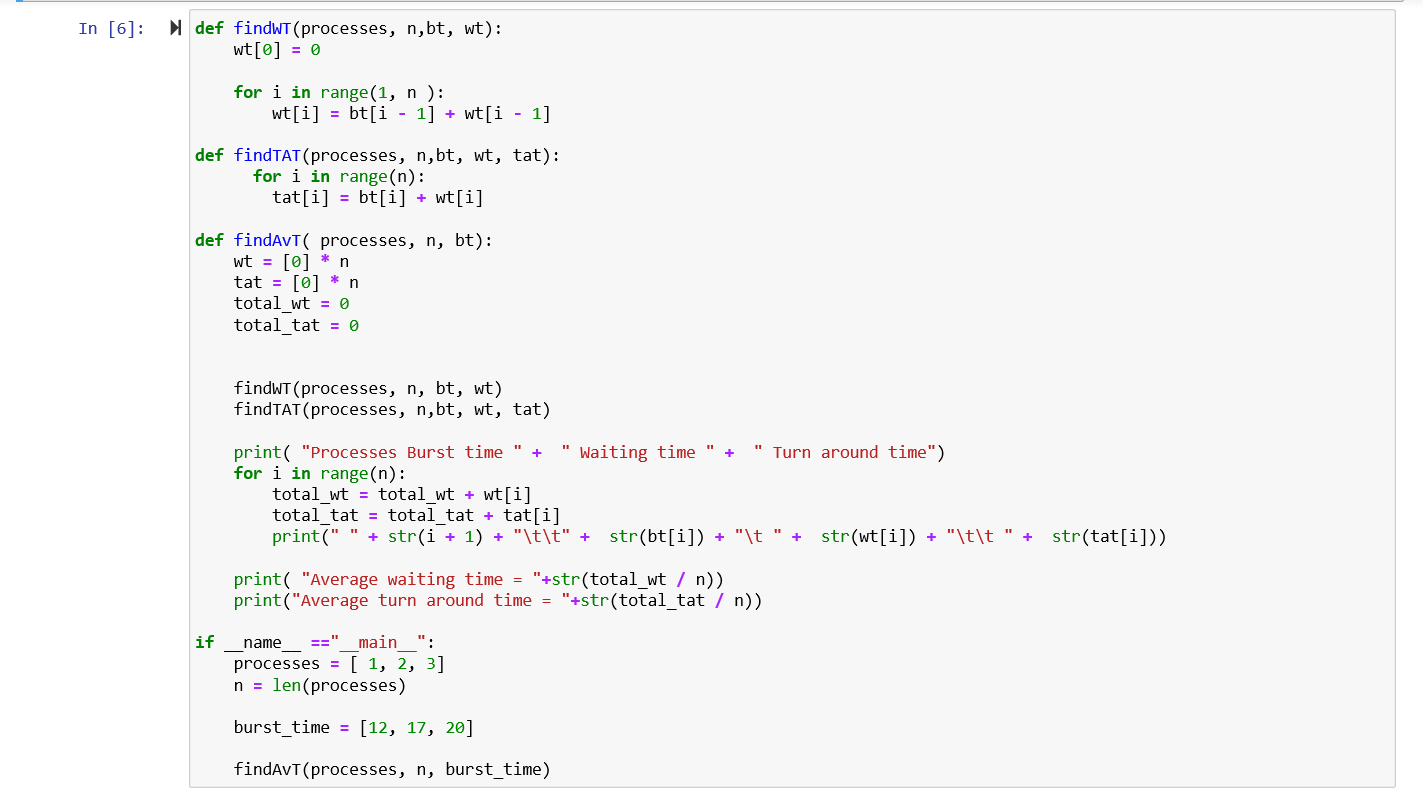
1. Input the processes along with their burst time(bt)
2. Find waiting time (wt) for all processes.
3. As first process that comes need to wait so waiting time for process 1 will be 0 i.e. wt[0] = 0.
4. Find waiting time for all other processes for processes i ->

wt[i] = bt[i - 1] + wt[i - 1]

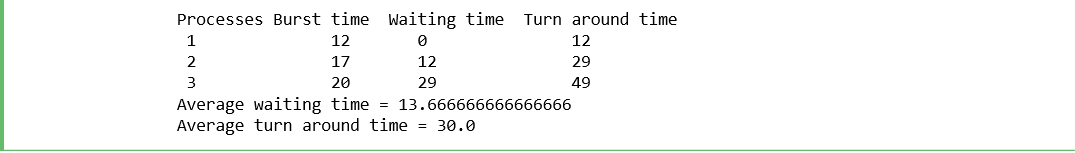
1. Find turnaround time = burst time – waiting time ,for all processes.
2. Find average waiting time= total weighting time/no.of processes
3. Similarly, find average

turnaround time = total turnaround time/no.of process.

**Code: Screenshot**

****

**Output: Screenshot**

****

**Suggested Question Bank**

**Preparatory Questions**

1. Which module gives control of the CPU to the process selected by the short-term scheduler?
   1. dispatcher
   2. interrupt
   3. scheduler
   4. none of the mentioned
2. The processes that are residing in main memory and are ready and waiting to execute are kept on a list called
   1. job queue
   2. ready queue
   3. execution queue
   4. process queue
3. The interval from the time of submission of a process to the time of completion is termed as
   1. waiting time
   2. turnaround time
   3. response time
   4. throughput
4. Which scheduling algorithm allocates the CPU first to the process that requests the CPU first?
   1. first-come, first-served scheduling
   2. shortest job scheduling
   3. priority scheduling
   4. none of the mentioned
5. In priority scheduling algorithm
   1. CPU is allocated to the process with highest priority
   2. CPU is allocated to the process with lowest priority
   3. equal priority processes can not be scheduled
   4. none of the mentioned
6. Process are classified into different groups in
   1. shortest job scheduling algorithm
   2. round robin scheduling algorithm
   3. priority scheduling algorithm
   4. multilevel queue scheduling algorithm
7. Which one of the following can not be scheduled by the kernel?
   1. kernel level thread
   2. user level thread
   3. process
   4. none of the mentioned
8. CPU scheduling is the basis of \_\_\_\_\_\_\_\_\_\_\_\_.
   1. multiprocessor systems
   2. multiprogramming operating systems
   3. larger memory sized systems
   4. None of these
9. With multiprogramming, \_\_\_\_\_\_ is used productively.

a) time b) space c) money d) All of these

1. The two steps of a process execution are : (choose two)

a) I/O Burst b) CPU Burst c) Memory Burst d) OS Burst

1. An I/O bound program will typically have :
   1. a few very short CPU bursts
   2. many very short I/O bursts
   3. many very short CPU bursts
   4. a few very short I/O bursts
2. A process is selected from the \_\_\_\_\_\_ queue by the \_\_\_\_\_\_\_\_ scheduler, to be executed.

a) blocked, short term b) wait, long term c) ready, short term d) ready, long term

1. In the following cases non – preemptive scheduling occurs : (Choose two)
   1. When a process switches from the running state to the ready state
   2. When a process goes from the running state to the waiting state
   3. When a process switches from the waiting state to the ready state
   4. When a process terminates
2. Dispatch latency is :
   1. the speed of dispatching a process from running to the ready state
   2. the time of dispatching a process from running to ready state and keeping the CPU idle
   3. the time to stop one process and start running another one
   4. None of these
3. Scheduling is done so as to :
   1. increase the throughput
   2. decrease the throughput
   3. increase the duration of a specific amount of work
   4. None of these
4. Turnaround time is :
   1. the total waiting time for a process to finish execution
   2. the total time spent in the ready queue
   3. the total time spent in the running queue
   4. the total time from the completion till the submission of a process
5. Scheduling is done so as to :
   1. increase the turnaround time
   2. decrease the turnaround time
   3. keep the turnaround time same
   4. there is no relation between scheduling and turnaround time

**Experiment No: 5**

Semester /Section: Vth / A

Link to Code:

Date: 1 September 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about CPU scheduling Algorithms.

**Program Outcome**

* The students will understand the Shortest job first algorithm.

**Problem Statement**

Implement the following CPU scheduling Algorithms.

i) SJF(with preemptive approach and non-preemptive approach)

**Background Study:**

**SHORTEST JOB FIRST**

* This algorithm associates with each process the length of the process's next CPU burst.
* When the CPU is available, it is assigned to the process that has the smallest next CPU burst. If the next CPU bursts of two processes are the same, FCFS scheduling is used.
* The SJF scheduling algorithm gives the minimum average waiting time for a given set of processes
* The real difficulty with the SJF algorithm knows the length of the next CPU request.
* Shortest Job first has the advantage of having minimum average waiting time among all scheduling algorithms.
* It is a Greedy Algorithm.
* It may cause starvation if shorter processes keep coming. This problem can be solved using the concept of aging.

**SJF with preemptive:**

**Algorithm:**

1. Transverse until all process gets completely executed.

* Find process with minimum remaining time at every single time lap.
* Reduce it time by 1.
* Check if it remaining time becomes 0.
* Increment the counter of process completion.
* Completion time of current process = current\_time +1;
* Calculate the waiting time for each completed process: wt[i] = Completion time-arrival time-burst time.
* Increment time lap by one.

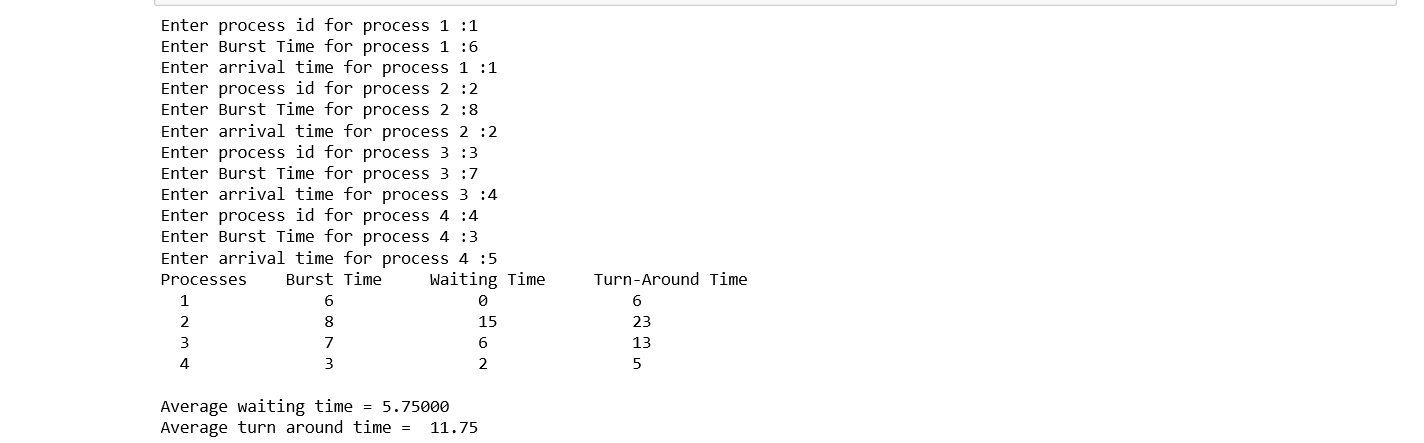
1. Find turnaround time (waiting time + burst time).
2. Find average waiting time= total weighting time/no.of processes
3. Similarly, find average turnaround time = total turnaround time/no.of process.

**Code: Screenshot**





**Output: Screenshot**



**SJF with non-preemptive**

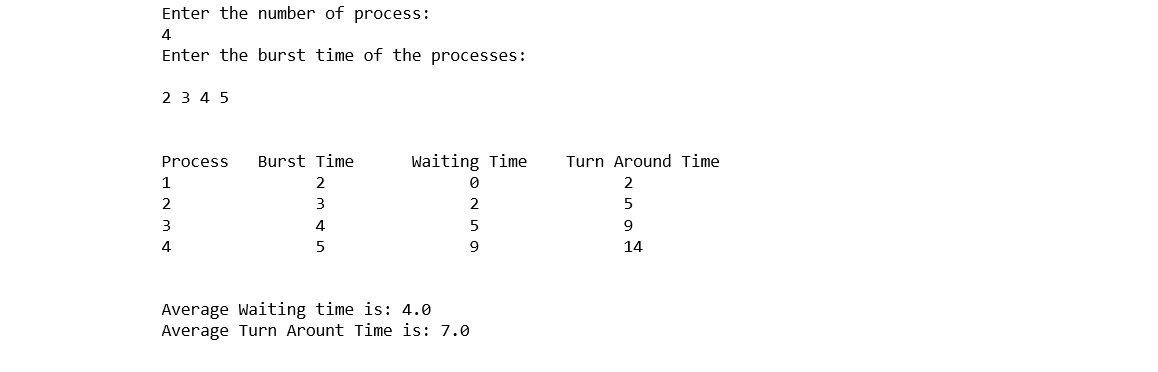
**Algorithm:**

1. Sort all the process according to their arrival time.
2. Then select that process which has minimum arrival time and minimum burst time.
3. After completion of process make a pool of process which after till the completion of previous process
4. Select that process among the pool which is having minimum burst time.

**Code: Screenshot**

****

**Output: Screenshot**

****

**Experiment No: 6**

Semester /Section: Vth / A

Link to Code: CSL 303

Date: 22 september 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about CPU scheduling Algorithms.

**Program Outcome**

* The students will understand the Priority scheduling algorithm.

**Problem Statement**

Implement the following CPU scheduling Algorithms.

i) Priority Scheduling

**Background Study:**

**Priority Scheduling**

**Study Background Priority Scheduling:-**

* Priority Scheduling is a method of scheduling processes that is based on priority. In this algorithm , the scheduler select the task to workas per the priority.

**Characteristics of Priority Scheduling:-**

* A CPU algorithm that schedules process based on priority.
* Lower the number , higher is the priority.
* It is used in operating system for performing batch process.

**Advantage:-**

* Easy to use scheduling method**.**
* Processes are executed on the basis of Priority so high priority does not need to wait for long which saves time.

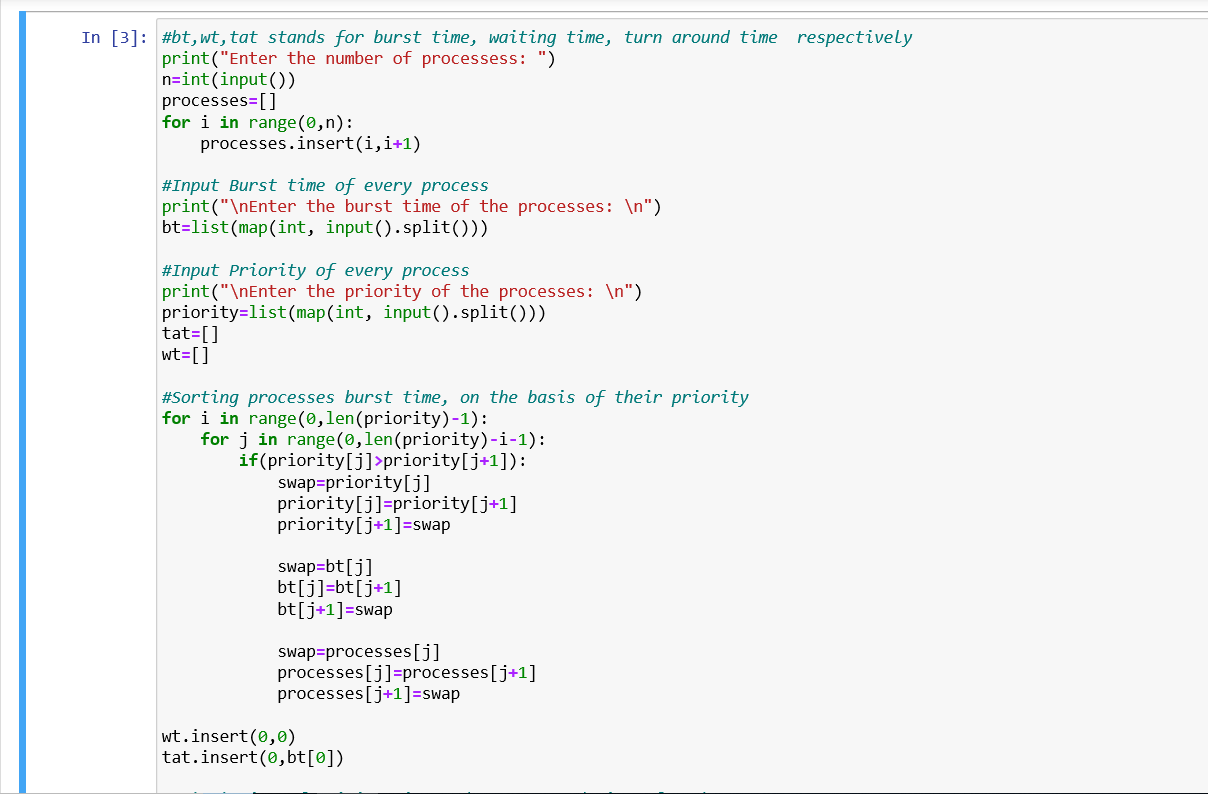
**Disadvantage:-**

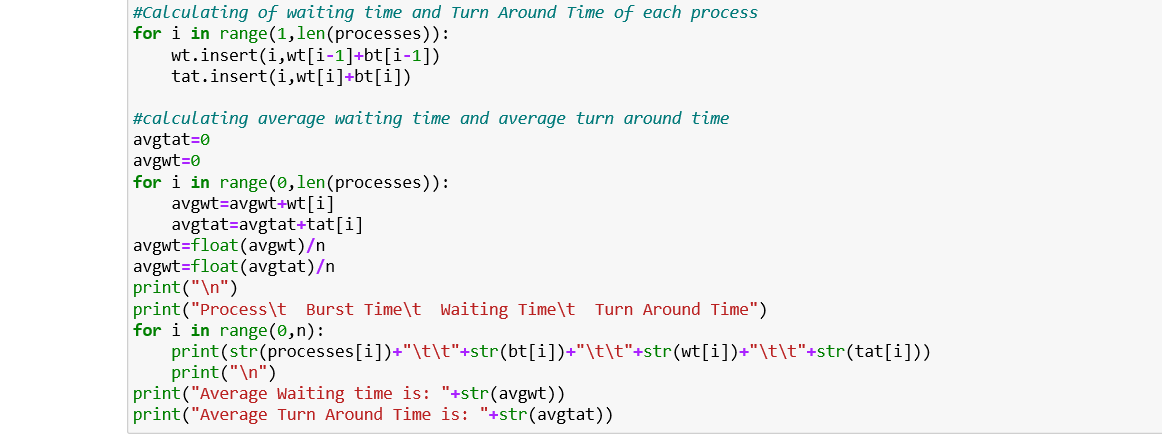
* If System eventually crashes, all low priority process get lost.
* The Scheduling Algorithm may leave some low priority processes may starve and will be postponed for an indefinite time.

**Algorithm:**

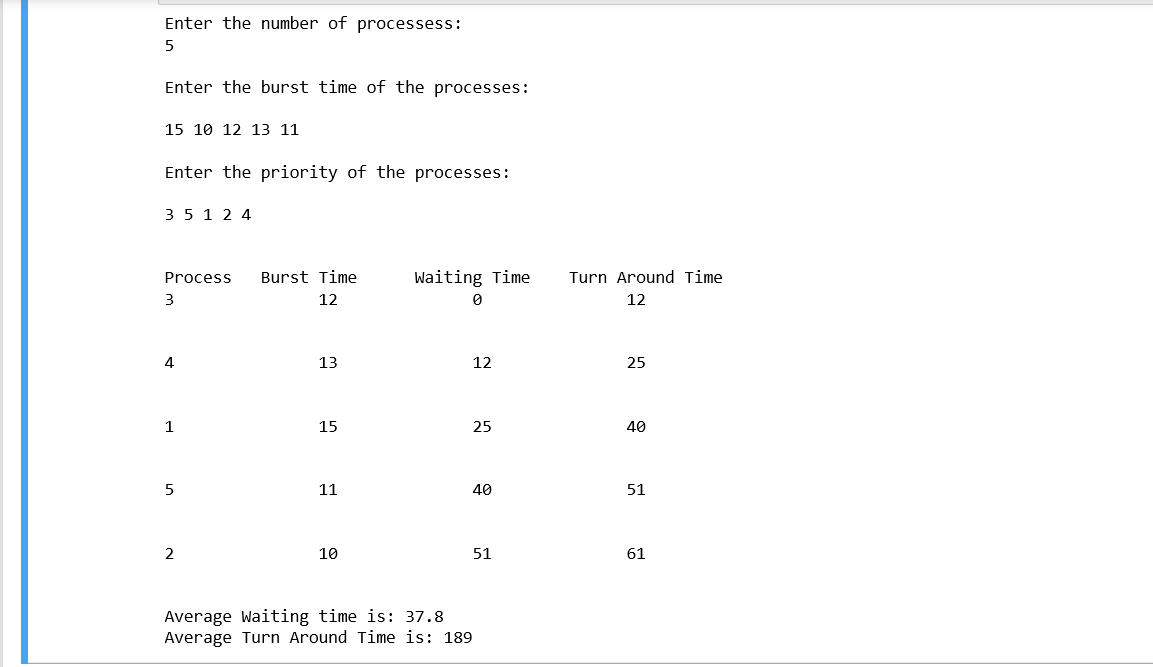
1. First input the processes with their burst time and priority.
2. Sort the processes, burst time and priority according to the priority.
3. Input the processes along with their burst time (bt).
4. Find waiting time (wt) for all processes.
5. As first process that comes need not to wait so waiting time for process 1 will be 0 i.e. wt[0] = 0.
6. Find **waiting time** for all other processes i.e. for
   1. process i ->
   2. wt[i] = bt[i-1] + wt[i-1] .
7. Find **turnaround time** = waiting\_time + burst\_time for all processes.
8. Find **average waiting time** = total\_waiting\_time / no\_of\_processes.
9. Similarly, find **average turnaround time** =total\_turn\_around\_time / no\_of\_processes.

**Code: Screenshot**





**Output: Screenshot**



**Experiment No: 7**

Semester /Section: Vth / A

Link to Code: CSL 303

Date: 29 september 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about CPU scheduling Algorithms.

**Program Outcome**

* The students will understand the Round Robins scheduling algorithm.

**Problem Statement**

Implement the following CPU scheduling Algorithms.

i) Round Robin Scheduling

**Background Study:**

* In Round Robin scheduling, each ready task runs turn by turn only in a cyclic queue for a limited time slice. This algorithm also offers starvation free execution of processes.

**Characteristics:-**

* Round Robin is a pre-emptive algorithm.
* The CPU is a shifted to the next process after fixed interval time, which is called time quantum.

**Advantage:-**

* It doesn’t face the issues of starvation on convoy effect.
* It deals with all process without any priority.
* It doesn’t depend on burst time.

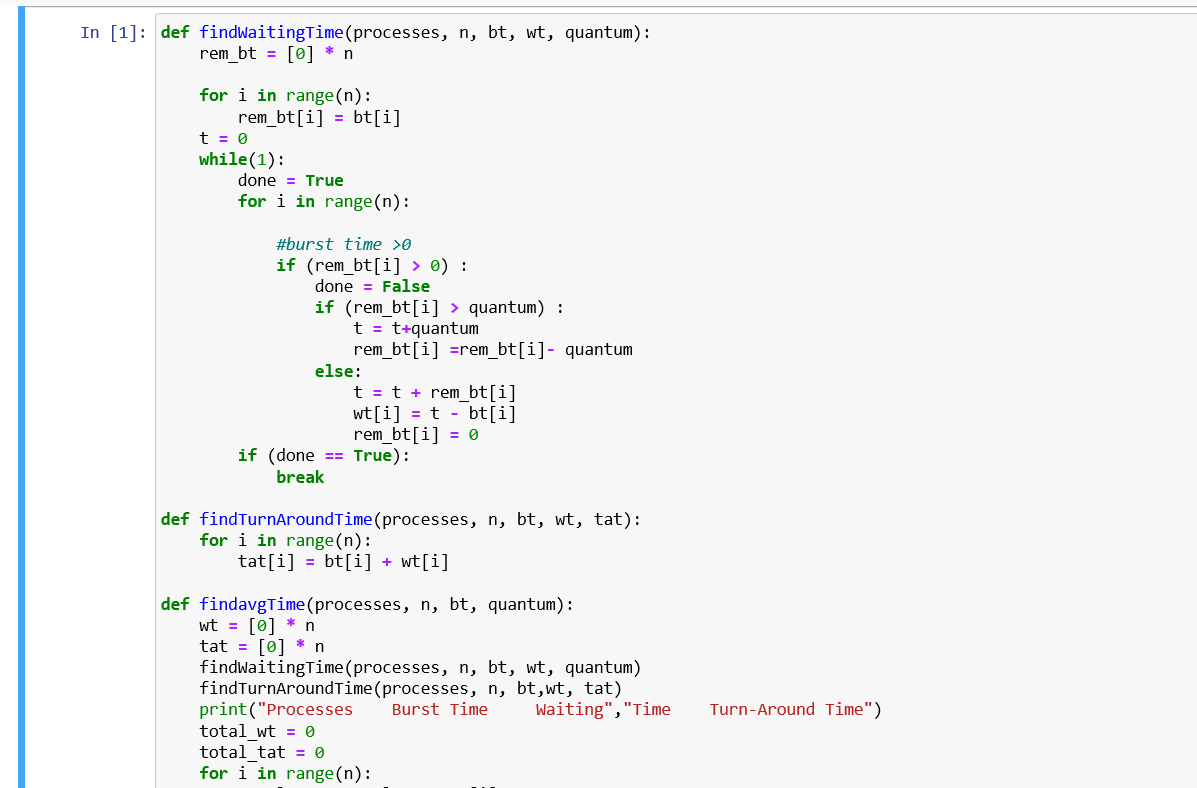
**Disadvantage:-**

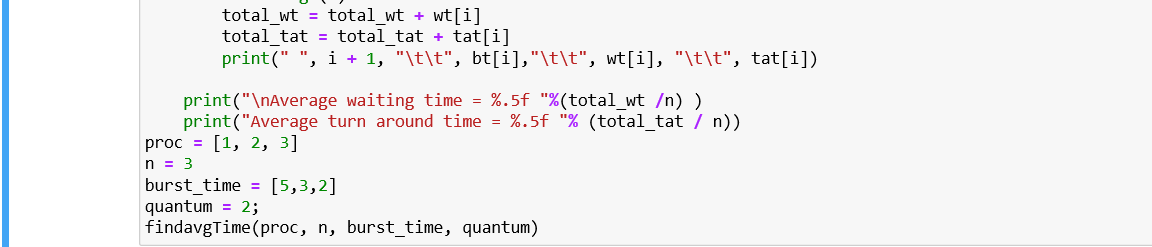
* If slicing time is OS is low, the processor output will be reduced.
* This method spends more time on context-switching.
* Its performance heavily depends on time quantum.

**Algorithm:-**

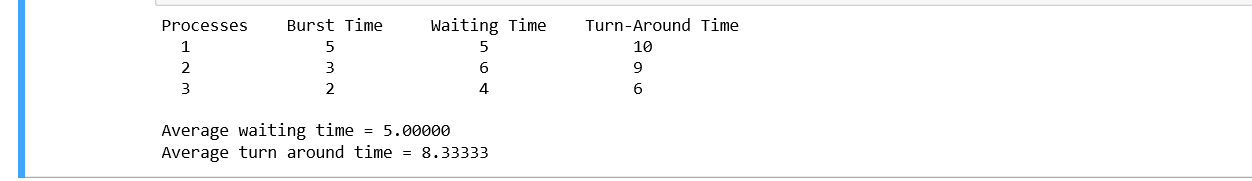
1. Create an array **rem\_bt[]** to keep track of remaining burst time of processes. This array is initially a copy of bt[] (burst times array)
2. Create another array **wt[]** to store waiting times of processes. Initialize this array as 0.
3. Initialize time: t = 0
4. Keep traversing all processes while all processes are not done. Do following for i'th process if it isnot done yet.
   1. If rem\_bt[i] > quantum
      1. t = t + quantum
      2. bt\_rem[i] -= quantum;
   2. Else // Last cycle for this process
      1. t = t + bt\_rem[i];
      2. wt[i] = t - bt[i]
      3. bt\_rem[i] = 0;

**Code: Screenshot**

****

****

**Output: Screenshot**

****

**Experiment No: 8**

Semester /Section: Vth / A

Link to Code: CSL 303

Date: 27 october 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about Reader-Writer Problem.

**Program Outcome**

* The students will understand the Reader-Writer Problem.

**Problem Statement**

Implement the following Reader-Writer Problem.

i) Reader-Writer Problem.

**Background Study:**

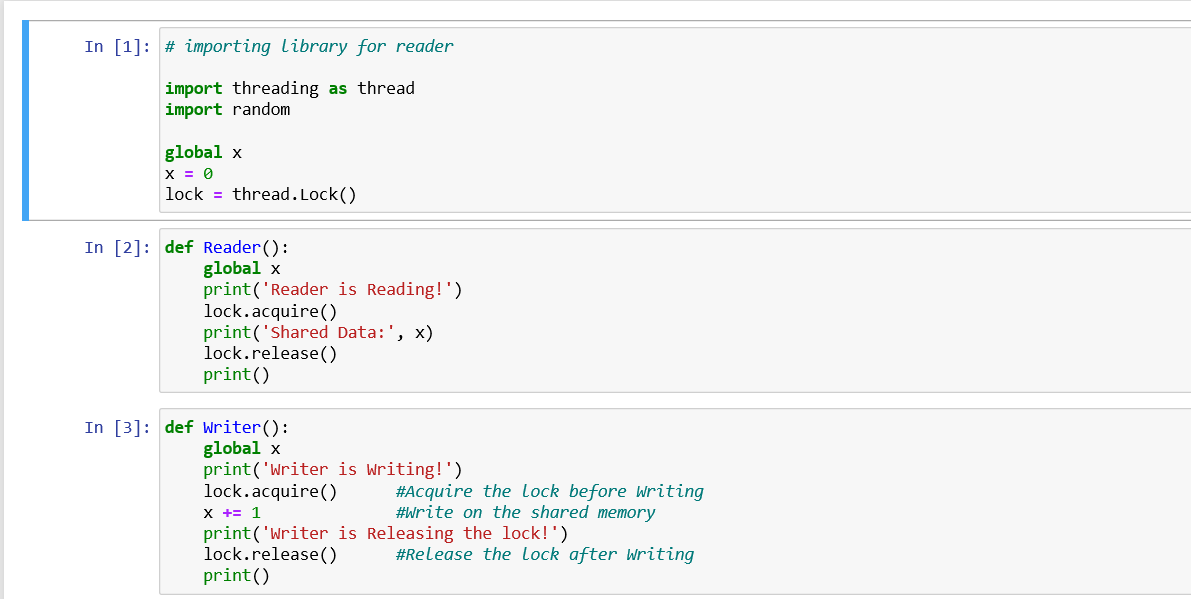
* The readers-writers problem relates to an object such as a file that is shared between multiple processes.
* To solve this situation, a writer should get exclusive access to an object i.e. when a writer is accessing the object, no reader or writer may access it.

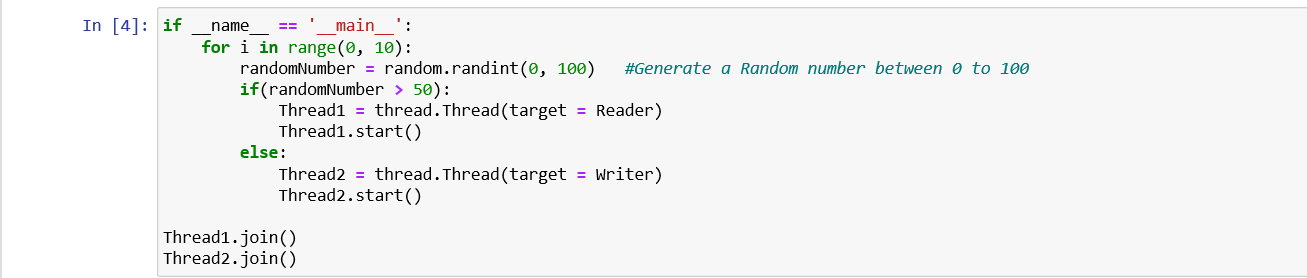
**Algorithm:**

1. Reader will run after Writer because of read semaphore.
2. Writer will stop writing when the write semaphore has reached 0.
3. Reader will stop reading when the read semaphore has reached 0.

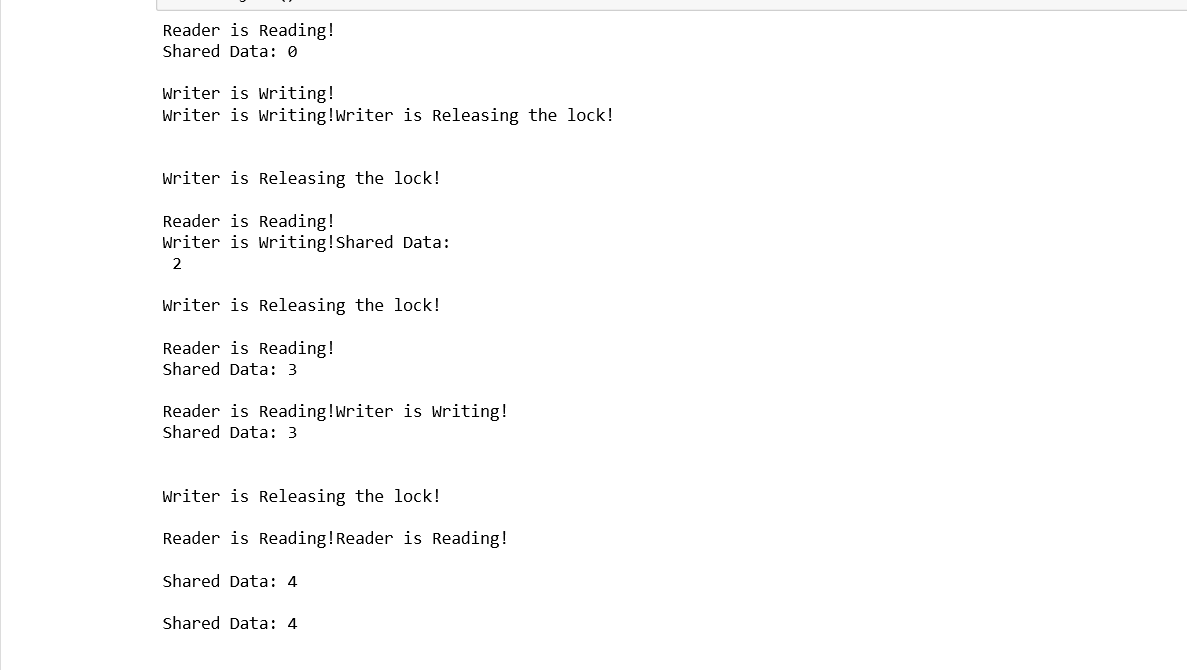
In writer, the value of write semaphore is given to read semaphore and in reader, the value of read is given to write on completion of the loop.

**Code: Screenshot:**

****

****

**Output: Screenshot**

****

**Experiment No: 9**

Semester /Section: Vth / A

Link to Code: CSL 303

Date: 10 november 2020

Faculty Signature:

Remarks:

**Objective**

To familiarize the students about Banker’s Algorithm.

**Program Outcome**

* The students will understand the Banker’s Algorithm Problem.

**Problem Statement**

Implement the following Banker’s Algorithm Problem.

i) Banker’s Algorithm .

**Background Study:**

* The banker’s algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an “s-state” check to test for possible activities, before deciding whether allocation should be allowed to continue.
* Banker's Algorithm is used majorly in the banking system to avoid deadlock. It helps you to identify whether a loan will be given or not. This algorithm is used to test for safely simulating the allocation for determining the maximum amount available for all resources.

**Drawbacks**:  
  
It requires the number of processes to be fixed; no additional processes can start while it is executing. It requires that the number of resources remain fixed; no resource may go down for any reason without the possibility of deadlock occurring.

**Algorithm:**

**Safety Algorithm:**

1) Let Work and Finish be vectors of length ‘m’ and ‘n’ respectively.  
Initialize: Work = Available  
Finish[i] = false; for i=1, 2, 3, 4….n

2) Find an i such that both  
a) Finish[i] = false  
b) Needi <= Work  
if no such i exists goto step (4)

3) Work = Work + Allocation[i]  
Finish[i] = true  
goto step (2)

4) if Finish [i] = true for all i  
then the system is in a safe state

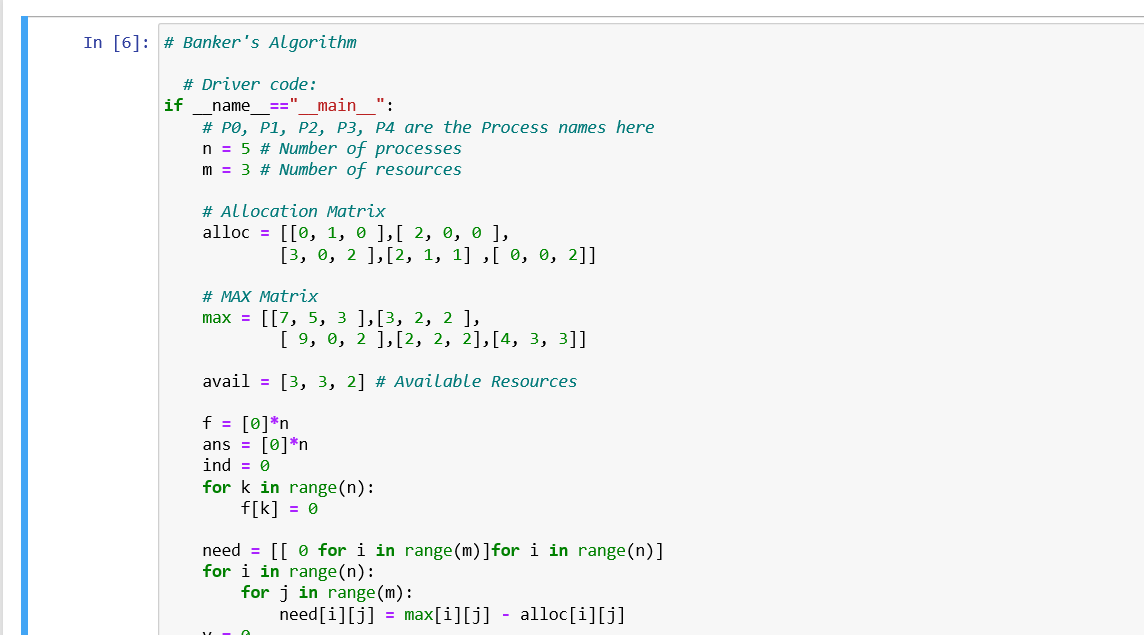
**Resource-Request Algorithm:**

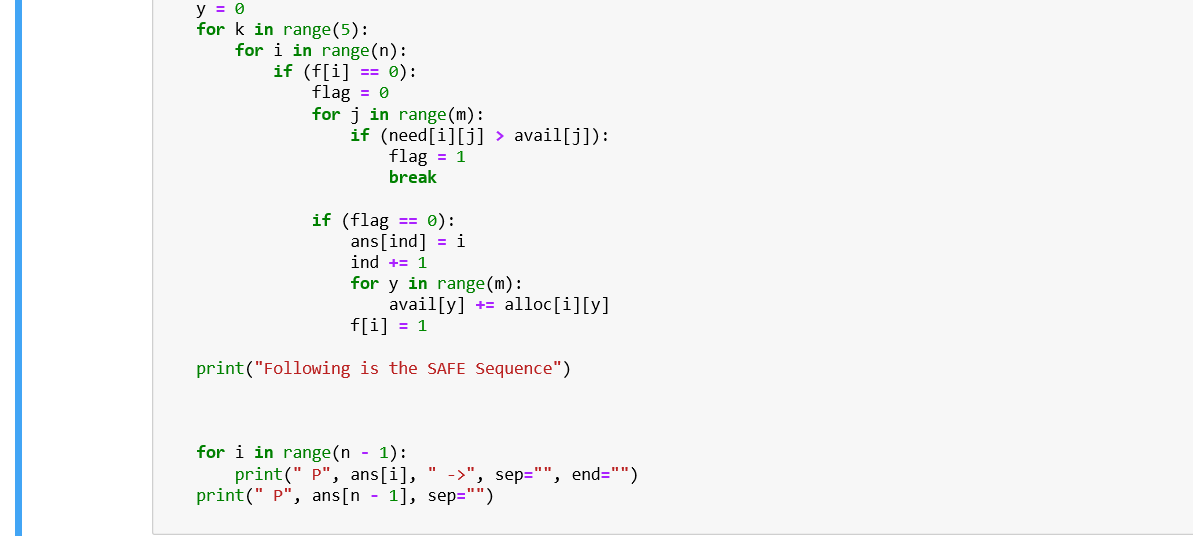
1) If Requesti <= Needi  
Goto step (2) ; otherwise, raise an error condition, since the process has exceeded its maximum claim.

2) If Requesti <= Available  
Goto step (3); otherwise, Pi must wait, since the resources are not available.

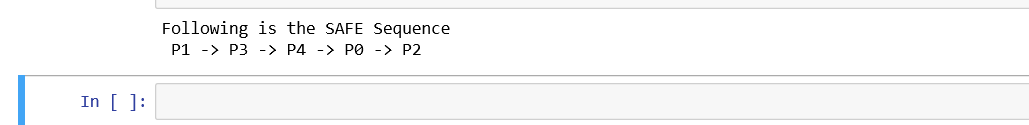
3) Have the system pretend to have allocated the requested resources to process Pi by modifying the state as  
follows:  
Available = Available – Requesti  
Allocationi = Allocationi + Requesti  
Needi = Needi– Requesti

**Code: Screenshot**

****

****

**Output: Screenshot**

****

**Experiment No: (Mini Project)**

Student Name and Roll Number: adityas

Semester /Section:

Link to Code:

Date:

Faculty Signature:

Remarks:

**Project Title:**

**Description of Project:**

**Problem Statement:**

**Problem Analysis:**

**Program Design:**

**Programming Requirements:**

**Data/Input Output Description:**

**Algorithmic Approach/Algorithm/DFD/ER diagram/Program Steps**

**Implementation and Testing (stage/module wise)**

**Output (Screenshots)**