**Assignment 2**

**Part A:**

**What will the commands do**

* Echo “Hello, World!”
* In shell scripting this command is just used to display some message to the user screen, this displays the content mentioned in “” , but it is not compulsory that to mention double quotes.
* Name=”Productive”
* Its string variable called name who has assigned a word Productive.
* Touch file.txt
* It creates the text file only, not open it.
* Ls -a
* List all file in current directories including hidden files.
* rm file.txt
* It removes / delete the file whichever is mentioned next to word rm.
* cp file1.txt file2.txt
* Copies the content of file1 to file2
* mv file.txt /path/to/directory/
* Moves the file.txt or whichever is mentioned to the specified location.
* chmod 755 script.sh
* changing the permission levels of script.sh file to Owner can R,W,X,

Group can R,X, and Other’s can R,X.

* grep "pattern" file.txt
* grep is used to find specific pattern in file.txt
* kill PID
* To kill / Terminates particular process PID.
* mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt
* A new directory named ‘mydir’ is created, navigate into ‘mydir’ directory, a file named file.txt, the text “Hello, World!” is written to ‘file.txt and then the content of file.txt is displayed showing “Hello, World!”.
* ls -l | grep ".txt"
* The command is used to list files and directories in the current directory and filter the results to show only those that contain .txt in their names.
* cat file1.txt file2.txt | sort | uniq
* Combines the content of file1 and file2 then sort them then removes the duplicate lines and keep only unique lines.
* ls -l | grep "^d"
* First lists the files and directories with detailed information and then filters the output to show only directories.
* grep -r "pattern" /path/to/directory/
* Recursively searches for the specific pattern within all files in the specified directory.
* cat file1.txt file2.txt | sort | uniq –d
* Displays the duplicate lines, shows lines that appear more than one across ‘file1’ and ‘file2’.
* chmod 644 file.txt
* This is setting file permission to like owner can able to edit the file, but others should only be able to view it.
* cp -r source\_directory destination\_directory
* Copies the content of sources directory to destination directory recursively.
* find /path/to/search -name "\*.txt"
* This command searches for file with a ‘.txt’ extension in the specified directory and its subdirectories.
* chmod u+x file.txt
* It modifies the permissions of the file ‘file.txt’ to add executable permission for the file’s owner (User).
* echo $PATH
* displays the current value of the ‘path’ environment variable in your shell.

**Part B:**

**Identify True or False:**

1. **ls** is used to list files and directories in a directory.

**-True**

1. mv is used to move files and directories.

- **True**

1. cd is used to copy files and directories.

-**False.**

1. pwd stands for "print working directory" and displays the current directory.

-**True**

1. grep is used to search for patterns in files.

-**True.**

6. chmod 755 file.txt gives read, write, and execute permissions to the owner, and read and execute permissions to group and others.

-**True**

7. mkdir -p directory1/directory2 creates nested directories, creating directory2 inside directory1 if directory1 does not exist.

-**True**

8. rm -rf file.txt deletes a file forcefully without confirmation.

-**True**

**Identify the Incorrect Commands:**

1. **chmodx** is used to change file permissions.

**- Incorrect.**

2. **cpy** is used to copy files and directories.

- **Incorrect.**

3. **mkfile** is used to create a new file.

- **Incorrect.**

4. **catx** is used to concatenate files.

- **Incorrect.**

5. **rn** is used to rename files.

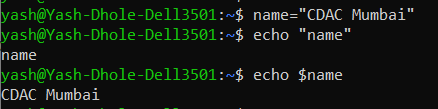
- **Incorrect.**

**Part C:**

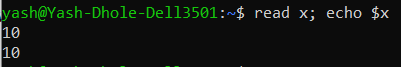
**Question 1:** Shell script to print “Hello, World!”



**Question 2:**  Display the value of a variable after assigning string “CDAC Mumbai”.



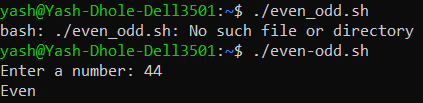
**Question 3:** Take input from user and display it.



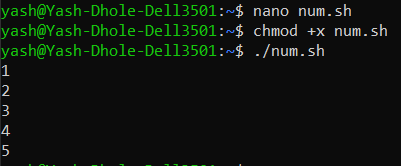
**Question 4:** Script of Addition of two number and print the result.



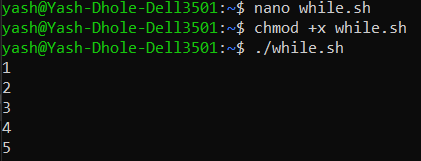
**Question 5:** Script to print even or odd number of user input.



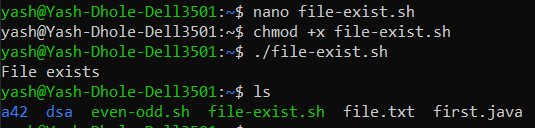
**Question 6:** Script of for loop from 1-5 and print it.



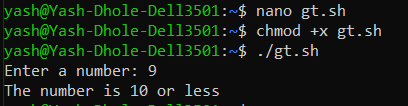
**Question 7:**  Script of While loop from 1-5 and print it.



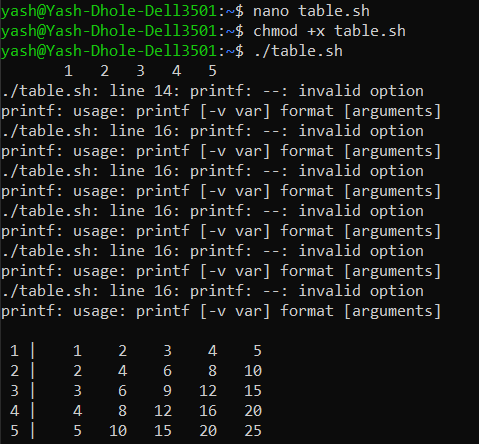
**Question 8:** Check the file availability in current directory and print available if it is else print not available.



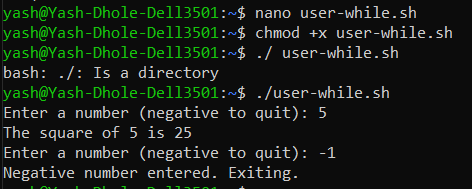
**Question 9:** Check the number is greater than 10 and print according to that.



**Question 10:**  Script to print table from 1-5.



**Question 11:** Script to read number from user till he enters the negative number and square of each positive number.



**Part D:**

**Interview Question**

1. What is an operating system, and what are its primary functions?

-An operating system is a software that acts as an intermediary between computer hardware and users. It manages resources like memory, storage, and processing power, and helps in running applications smoothly. The OS also handles tasks such as managing files, controlling processes, and ensuring that devices like printers and keyboards work properly.

1. Explain the difference between process and thread ?

-A process is an independent program in execution with its own memory space, while a thread is a smaller unit within a process that can run independently but shares the same memory and resources with other threads in the same process. Processes are isolated from each other, making them more secure, but they require more resources. Threads, on the other hand, are lighter and allow for faster communication within a process but can lead to issues like data corruption if not handled properly.

1. What is virtual memory and how does it work ?

-Virtual memory is a memory management technique used by operating systems to extend the apparent amount of available RAM by using a portion of the hard drive as additional memory. When the actual RAM is full, the OS temporarily moves data that isn't actively being used to a space on the hard drive called the "swap file" or "page file." This frees up RAM for other tasks. When the data is needed again, it's swapped back into RAM. This process allows computers to run larger applications or multiple programs simultaneously, even if the physical RAM is limited.

1. Describe the difference between multiprogramming, multitasking, and multiprocessing.

-Multiprogramming is a method where multiple programs are loaded into memory at the same time, and the CPU switches between them to maximize efficiency, but only one program is executed at any given moment.

Multitasking is an extension of multiprogramming where the CPU switches between tasks so quickly that it appears multiple tasks are running simultaneously. It allows users to interact with multiple applications at the same time, like browsing the web while listening to music.

Multiprocessing involves the use of two or more CPUs within a single computer system to execute multiple processes simultaneously. This allows for true parallel execution of processes, leading to better performance, especially in systems that handle complex computations or high workloads.

1. What is file system, and what are its component?

-A file system is a way the operating system organizes and manages data on storage devices. It includes components like files and directories, which hold the data and organize it, and metadata, which stores details about the files. Additionally, structures like the File Allocation Table or keep track of where the data is located on the disk, ensuring efficient storage and retrieval.

6. What is a deadlock, and how can it be prevented?

-A deadlock is a situation in operating systems where two or more processes are unable to proceed because each is waiting for the other to release a resource. This creates a cycle of dependency that halts all involved processes. To prevent deadlocks, techniques like avoiding circular wait by ordering resource allocation, using resource allocation protocols or implementing timeout mechanisms can be applied. These methods help ensure that processes don’t get stuck waiting indefinitely.

7. Explain the difference Kernel and Shell

-The kernel is the core part of an operating system that directly interacts with the hardware, managing resources like CPU, memory, and devices. It handles essential tasks like process management, memory management, and device control. The shell, on the other hand, is an interface between the user and the kernel. It allows users to interact with the operating system by typing commands, which the shell interprets and passes on to the kernel for execution. In simple terms, the kernel manages the system, while the shell provides a way for users to communicate with it.

8. What is CPU scheduling, and why is it important?

-CPU scheduling is the process of deciding which task or process the CPU should work on next. It’s important because it helps ensure that the CPU is used efficiently, allowing multiple tasks to run smoothly and improving overall system performance.

9. How does system call works ?

-A system call is a way for a program to request a service from the operating system, like reading a file or creating a process. When a program makes a system call, it triggers a software interrupt that switches the CPU from user mode to kernel mode, where the operating system can safely perform the requested action. After the service is completed, the CPU switches back to user mode, and the program continues running.

10. What is the purpose of device drivers in an operating system?

-Device drivers are specialized software that allow the operating system to communicate with hardware devices like printers and keyboards. Their purpose is to translate the OS’s general commands into specific instructions that the hardware can understand, enabling the proper functioning of devices and ensuring they work correctly with the system.

11. Explain the role of the page table in virtual memory management

-The page table is crucial in virtual memory management as it maps virtual addresses used by applications to physical addresses in the computer's memory. When a program accesses a memory location, the page table translates the virtual address into a physical address, allowing the system to find and use the correct data. This mapping helps manage and optimize the use of memory, enabling efficient use of both RAM and disk space through techniques like paging.

12. What is thrashing, and how can it be avoided?

-Thrashing is a situation where the operating system spends more time swapping data between RAM and disk than executing processes, leading to severe performance degradation. It typically happens when there are too many processes or when processes require more memory than is available. To avoid thrashing, you can increase the amount of physical memory, optimize the number of processes running simultaneously, and improve memory management techniques to ensure efficient use of available resources.

13. Describe the concept of a semaphore and its use in synchronization.

-A semaphore is a synchronization tool used in operating systems to manage access to shared resources and prevent conflicts between processes. It is essentially a counter that controls access to resources by using two main operations: wait and signal. The wait operation decreases the semaphore's count and can block a process if the count is zero, while the signal operation increases the count and can unblock a waiting process. Semaphores help ensure that only a limited number of processes access a resource at a time, preventing issues like race conditions and ensuring smooth coordination between processes.

14. What is the purpose of an interrupt in operating systems?

-Interrupts are signals sent to the CPU by hardware or software indicating that an immediate attention is needed. The purpose of an interrupt is to alert the CPU to stop its current activities and address an urgent task, such as handling input from a keyboard or responding to a hardware failure. This allows the operating system to manage multiple tasks efficiently and ensures timely processing of critical events without having to constantly check for them.

15. How does an operating system handle process synchronization?  
 -An operating system handles process synchronization using mechanisms like mutexes, semaphores, and monitors. These tools help coordinate the activities of multiple processes or threads to avoid conflicts and ensure data consistency. For example, mutexes provide exclusive access to shared resources, semaphores manage access with counters, and monitors use condition variables to control process execution. By managing these synchronization tools, the OS ensures that processes run smoothly and data integrity is maintained.

16. Explain the concept of file descriptor ?

-A file descriptor is an integer that serves as a reference to an open file or other I/O resource, such as a socket or pipe, in an operating system. When a file is opened or a resource is created, the OS assigns a unique file descriptor to it. This number is used by the operating system and applications to perform operations on the file or resource, such as reading, writing, or closing it. File descriptors provide a way to efficiently manage and interact with multiple I/O resources.

17. How does a system recover from a system crash?

-When a system crashes, it restarts and reinitializes, checking for file corruption and repairing it. Lost data is recovered from backups or logs, and critical processes and services are restarted. This process helps restore the system to normal operation with minimal data loss and disruption.

**Part E:**

**Question1:** Consider the following processes with arrival times and burst times:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

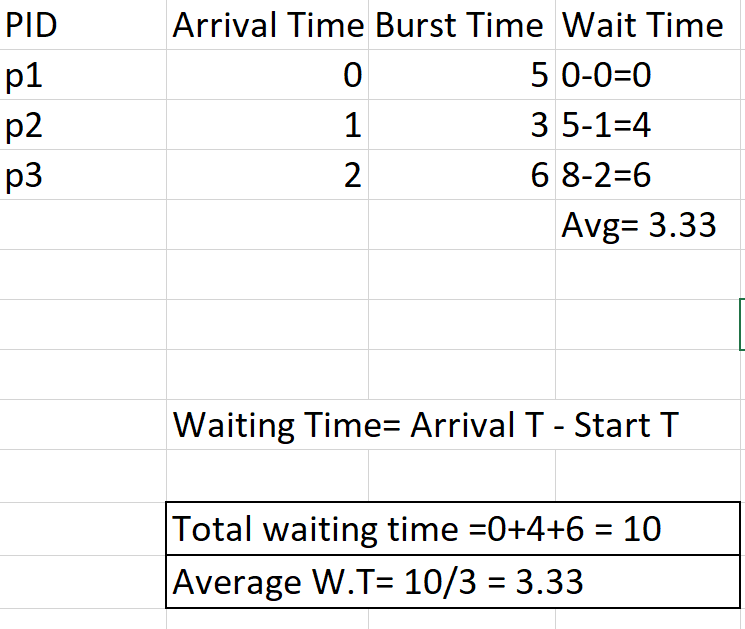
| P1 | 0 | 5 |

| P2 | 1 | 3 |

| P3 | 2 | 6 |

Calculate the average waiting time using First-Come, First-Served (FCFS) scheduling.

Solution :



**Question 2:** Consider the following processes with arrival times and burst times:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

| P1 | 0 | 3 |

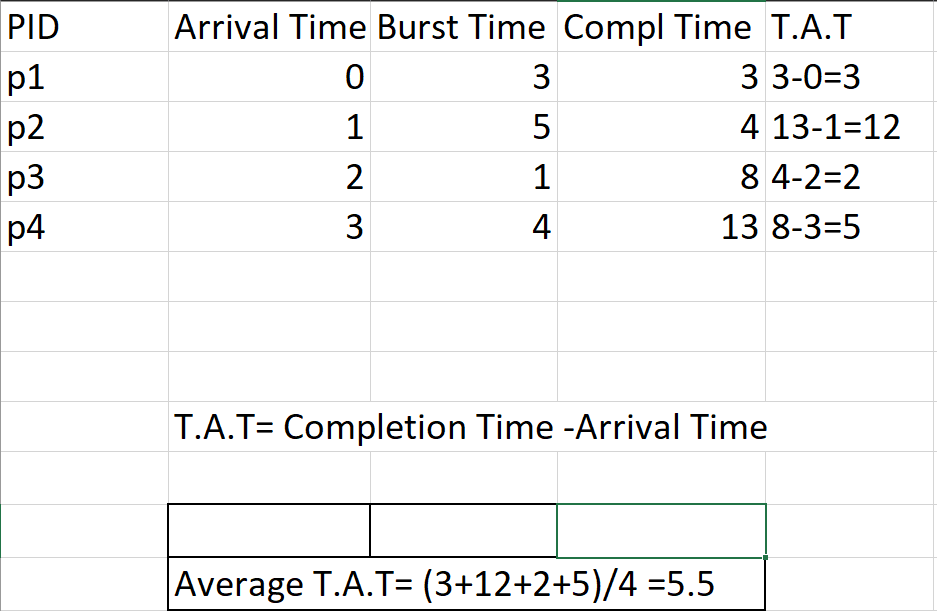
| P2 | 1 | 5 |

| P3 | 2 | 1 |

| P4 | 3 | 4 |

Calculate the average turnaround time using Shortest Job First (SJF) scheduling.

Solution:



**Question3:** Consider the following processes with arrival times, burst times, and priorities (lower number

indicates higher priority):

| Process | Arrival Time | Burst Time | Priority |

|---------|--------------|------------|----------|

| P1 | 0 | 6 | 3 |

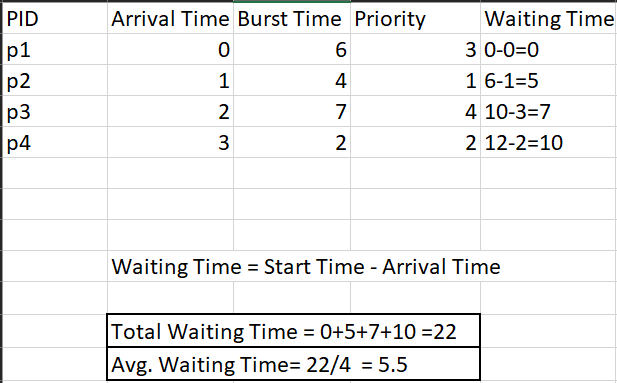
| P2 | 1 | 4 | 1 |

| P3 | 2 | 7 | 4 |

| P4 | 3 | 2 | 2 |

Calculate the average waiting time using Priority Scheduling.

Solution:



**Question 4:** Consider the following processes with arrival times and burst times, and the time quantum for

Round Robin scheduling is 2 units:

| Process | Arrival Time | Burst Time |

|---------|--------------|------------|

| P1 | 0 | 4 |

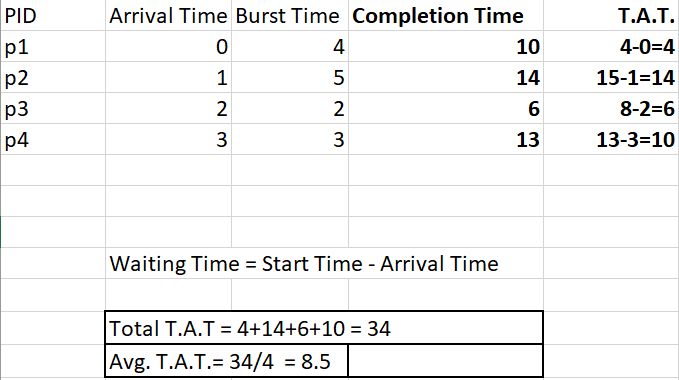
| P2 | 1 | 5 |

| P3 | 2 | 2 |

| P4 | 3 | 3 |

Calculate the average turnaround time using Round Robin scheduling.

Solution :



**Question 5** : Consider a program that uses the fork() system call to create a child process. Initially, the parent

process has a variable x with a value of 5. After forking, both the parent and child processes

increment the value of x by 1.

What will be the final values of x in the parent and child processes after the fork() call?

Solution:

-Before fork call:

Parent process X = 5

-After fork both the child processes have their own copy of X with the value 5.

-Parent Process:

Original X = 5

After incrementing: X = 5+1 = 6

-Child Process:

Original X = 5

After Increment: X =5 +1 = 6

Finally Parent Process **X = 6 &** Child Process **X = 6.**