**CDAC MUMBAI**

Concepts of Operating System

**Assignment 2**

**\*Part A**

**Q.What will the following commands do?**

• echo "Hello, World!"

* Prints Hello,World

• name="Productive"

* Here, name is variable and Productive is its value

• touch file.txt

* It creates a file named as file.txt

• ls -a

* Lists all files,directories and it also lists the hidden ones

• rm file.txt

* It removes file.txt file

• cp file1.txt file2.txt

* It copies the file1.txt file into file2.txt

• mv file.txt /path/to/directory/

* Moves file.txt file into given directory

• chmod 755 script.sh

* It changes mod to 755 where 7 is read+write +execute and 5 is read and execute
* It allows everyone to execute here

• grep "pattern" file.txt

* Searches pattern word into file.txt file

• kill PID

* kills the process with given id

• mkdir mydir && cd mydir && touch file.txt && echo "Hello, World!" > file.txt && cat file.txt

* Created mydir directory and enter into mydir and created a file named as file.txt and printed Hello, World text into file.txt and displayed file.txt contents

• ls -l | grep ".txt"

* Listed in long format and searches only for “.txt” files

• cat file1.txt file2.txt | sort | uniq

* Concatenates file1.txt and file2.txt
* Then sort them
* Removes duplicates

• ls -l | grep "^d" • grep -r "pattern" /path/to/directory/

* Listed files in long format and searches only directories and
* searches recursively for pattern in all files and directories

• cat file1.txt file2.txt | sort | uniq –d

* Concatenqates file1.txt and file2.txt
* Sort them alphabetically
* And seperates only duplicated ones

• chmod 644 file.txt

* Changes mode and gives 644 permissions
* Where 6 stands for read and write and 4 stands for read only
* So everyone have the permission to read here and only owner can write and read

• cp -r source\_directory destination\_directory

* Copies all files and subfolders into destination directory from source directory

• find /path/to/search -name "\*.txt"

* Searches all files of this path which ends with ‘.txt’

• chmod u+x file.txt

* Adds execute permission to only user of file.txt

• echo $PATH P

* Prints the system path of this envt. Var

**\*Part B**

**Q. Identify True or False:**

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

-> **True**

2. mv is used to move files and directories.

-> **True**

3. cd is used to copy files and directories.

-> **False**

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

-> **True**

5. 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**

**Q.Identify the Incorrect Commands:**

1. chmodx is used to change file permissions.

-> chmod

2. cpy is used to copy files and directories.

-> cp

3. mkfile is used to create a new file.

-> touch

4. catx is used to concatenate files.

-> cat

5. rn is used to rename files.

-> mv oldname newname

**\*Part C**

**Question 1:** Write a shell script that prints "Hello, World!" to the terminal.

* echo “Hellp,World!”

**Question 2:** Declare a variable named "name" and assign the value "CDAC Mumbai" to it. Print the value of the variable.

* name = “CDAC Mumbai”
* echo “the name is $name”

Question 3: Write a shell script that takes a number as input from the user and prints it.

* read -p “enter a number - ” num
* echo “you entered the number is $num”

Question 4: Write a shell script that performs addition of two numbers (e.g., 5 and 3) and prints the result.

* a=5
* b=3
* sum=$ ((a+b))
* echo “the sum of $a and $b is $sum”

Question 5: Write a shell script that takes a number as input and prints "Even" if it is even, otherwise prints "Odd".

* read -p “Enter a num - ” num
* if((num %2 == 0 ));then
* echo “Even”
* else
* echo “Odd”
* fi

Question 6: Write a shell script that uses a for loop to print numbers from 1 to 5.

* for i in {1…5}
* do
* echo”$i”
* done

Question 7: Write a shell script that uses a while loop to print numbers from 1 to 5.

* i=1
* while [ $i -le 5 ]
* do
* echo "$i"
* ((i++))
* done

Question 8: Write a shell script that checks if a file named "file.txt" exists in the current directory. If it does, print "File exists", otherwise, print "File does not exist".

* if [ -f "file.txt" ]; then
* echo "File exists
* Else
* echo "File does not exist"
* fi

Question 9: Write a shell script that uses the if statement to check if a number is greater than 10 and prints a message accordingly.

* read -p "Enter a number: " num
* if [ $num -gt 10 ]; then
* echo "The number is greater than 10"
* else
* echo "The number is not greater than 10"
* fi

Question 10: Write a shell script that uses nested for loops to print a multiplication table for numbers from 1 to 5. The output should be formatted nicely, with each row representing a number and each column representing the multiplication result for that number.

* for i in {1..5}
* do
* for j in {1..5}
* do
* printf "%4d" $((i \* j))
* done
* echo
* done

Question 11: Write a shell script that uses a while loop to read numbers from the user until the user enters a negative number. For each positive number entered, print its square. Use the break statement to exit the loop when a negative number is entered.

* while true
* do
* read -p "Enter a number (negative to exit): " num
* if [ $num -lt 0 ]; then
* echo "Negative number entered. Exiting...
* break
* fi
* square=$((num \* num))
* echo "Square of $num is: $square”
* done

**\*Part D**

**Common Interview Questions (Must know)**

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

-> OS is an intermediator of computer hardware and user. OS controls the hardware and coordinates its use among various application program for the various users

-> the primary functions are, process,memory,file system, device managements and security & user interface

2. Explain the difference between process and thread.

-> process is an independent pgm in execution and it has its own memory

-> thread shares its memory with other threads

3. What is virtual memory, and how does it work?

-> it gives an illusion of having more memory than RAM

-> Virtual memory works by giving each program its virtual address space which is mapped to physical RAM through a page table . if data is needed in RAM then it works directly , if data is not needed then fault page occurs and OS loads it from disk into RAM and if the RAM is full then OS swaps the less used pages to make more space.

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

-> Multiprogramming – multiple programs in a memory and CPU switches when one waits for i/o

-> Multitasking – multiple tasks running at the same time

-> Multiprocessing -system with multiple CPUs running in parallel

5. What is a file system, and what are its components?

-> A file system is a method used by an operating system (OS) to store, organize, retrieve, and manage data on storage devices like hard drives, SSDs, CDs, or USBs.

Its components are Files + Directories + Metadata + Storage management + Security.

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

-> Deadlock is a situation where set of processes are blocked because each process is holding a resource & waiting for another resource aquired by some other process.

Deadlock = I will go. No I will go and nobody goes

To prevent the deadlocks we have to allow the preemption,no hold and wait,no circular wait and reduce the mutual exclusion.

7. Explain the difference between a kernel and a shell.

-> SHELL :1) Shell is an intermediator of kernel and user.

2) It’s a command line interface

3) Shell is an outer level of an OS

-> KERNEL : 1)Kernel is a brain of OS.

2) It’s a low level programme

3) Kernel is an inner level of an OS

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

-> CPU Scheduling: Decides which process runs on the CPU when multiples are ready.

Its important because it can have a big effect on resources utilization and the overall performance of the system.

9. How does a system call work?

-> System call is an interface b/w user process and kernel

->The processor executes a process in the user mode until a system call interrupts it -> Then on a priority basis, the system call is executed in the kernel mode -> After the completion of system call execution control returns to user mode ->  
The execution resumes in Kernel mode.

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

-> Device drivers are programs that act as translators between OS and hardware. They hide hardware details, provide a uniform interface, and allow the OS to control devices.

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

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

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

14. How does an operating system handle process synchronization?

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

16. Explain the concept of a file descriptor.

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

18. Describe the difference between a monolithic kernel and a microkernel.

19. What is the difference between internal and external fragmentation?

20. How does an operating system manage I/O operations?

21. Explain the difference between preemptive and non-preemptive scheduling. 22. What is round-robin scheduling, and how does it work?

23. Describe the priority scheduling algorithm. How is priority assigned to processes?

24. What is the shortest job next (SJN) scheduling algorithm, and when is it used? 25. Explain the concept of multilevel queue scheduling.

26. What is a process control block (PCB), and what information does it contain? 27. Describe the process state diagram and the transitions between different process states.

28. How does a process communicate with another process in an operating system?

29. What is process synchronization, and why is it important?

30. Explain the concept of a zombie process and how it is created

. 31. Describe the difference between internal fragmentation and external fragmentation.

32. What is demand paging, and how does it improve memory management efficiency?

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

34. How does a memory management unit (MMU) work?

35. What is thrashing, and how can it be avoided in virtual memory systems?

36. What is a system call, and how does it facilitate communication between user programs and the operating system?

37. Describe the difference between a monolithic kernel and a microkernel.

38. How does an operating system handle I/O operations?

39. Explain the concept of a race condition and how it can be prevented.

40. Describe the role of device drivers in an operating system. 41. What is a zombie process, and how does it occur? How can a zombie process be prevented? 42. Explain the concept of an orphan process. How does an operating system handle orphan processes?

43. What is the relationship between a parent process and a child process in the context of process management?

44. How does the fork() system call work in creating a new process in Unix-like operating systems?

45. Describe how a parent process can wait for a child process to finish execution. 46. What is the significance of the exit status of a child process in the wait() system call?

47. How can a parent process terminate a child process in Unix-like operating systems?

48. Explain the difference between a process group and a session in Unix-like operating systems.

49. Describe how the exec() family of functions is used to replace the current process image with a new one.

50. What is the purpose of the waitpid() system call in process management? How does it differ from wait()?

51. How does process termination occur in Unix-like operating systems?

52. What is the role of the long-term scheduler in the process scheduling hierarchy? How does it influence the degree of multiprogramming in an operating system?

53. How does the short-term scheduler differ from the long-term and medium-term schedulers in terms of frequency of execution and the scope of its decisions?

54. Describe a scenario where the medium-term scheduler would be invoked and explain how it helps manage system resources more efficiently.

**\*Part E**

1. 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.

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.

3. 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.

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.

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?