CDAC MUMBAI

Concepts of Operating System

Assignment 2

Part A

What will the following commands do?

1.echo "Hello, World!"

ans: It prints “Hello World” to terminal

2. name="Productive"

ans: assigns the value “Productive” to the variable ‘name’



3. touch file.txt

ans: creates a new file as file.txt in same directory



4. ls -a

ans: Lists all files and directories in the current directory, including hidden ones

5. rm file.txt

ans:Removes the file named file.txt

6. cp file1.txt file2.txt

ans: it copies the content of file1.txt to file2.txt

7. mv file.txt /path/to/directory/

ans: it moves the file.txt to the given path.

8. chmod 755 script.sh

ans: this command give the permission of read, write and execute to the owner and only read and execute to the group and other.

9. grep "pattern" file.txt

ans: searches for “pattern” in the file.txt and prints the specific patterns

10. kill PID

ans: Sends a signal to terminate the process with the process Id PID

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

ans: mkdir mydir : Creates a directory named mydir.

cd mydir : Changes the current directory to mydir.

touch file.txt: Creates an empty file named file.txt.

echo "Hello, World!" > file.txt: Writes the string "Hello, World!" to file.txt, overwriting any existing content.

cat file.txt: Displays the contents of file.txt.

12. ls -l | grep ".txt"

ans: display all files in current directories in long form and filters it to the .txt file, it will show only .txt files.

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

ans: concatenates of file1.txt and file2.txt ,sorts the combined output and then displays only the duplicate lines that are in both files

14. ls -l | grep "^d"

ans: Lists all files and directories ,show only lines that start with "d" i.e shows detailed information only for directories.

15. grep -r "pattern" /path/to/directory/

ans: finds and shows all lines containing "pattern" in any file inside the specified directory and its subdirectories.

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

ans: shows the lines that are common in both file1.txt and file2.txt.

17. chmod 644 file.txt

ans: allows the file's owner to read and write the file, while everyone else can only read it.

18. cp -r source\_directory destination\_directory

ans: copy the source\_directory and everything inside it to the destination\_directory.

19. find /path/to/search -name "\*.txt"

ans: finds and lists all text files “.txt” in the specified directory and any folders inside it.

20. chmod u+x file.txt

ans: allows the file's owner to run file.txt as a program or script.

21. echo $PATH

ans: shows the directories system searches for executables when you run a command.

Part B

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

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,nano,cat

4. catx is used to concatenate files. - cat

5. rn is used to rename files. - mv

Part C

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

Answer:

#! /bin/bash

echo "Hello World"

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano hello.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash hello

Hello World

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

value of the variable

Answer:

#! /bin/bash

name="CDAC Mumbai"

echo "The value of variable name is $name"

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question2.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question2.sh

The value of variable name is CDAC Mumbai

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

Answer:

#! /bin/bash

echo Enter the number

read number

echo "you entered the $number number"

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question3.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question3.sh

Enter the number

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

result.

Answer:

shell script:-

#! /bin/bash

echo please enter num1

read num1

echo please enter num2

read num2

sum=$(( num1 + num2 ))

echo "the addition of two numbers is $sum"

terminal command:-

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question4.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question4.sh

please enter num1

5

please enter num2

6

the addition of two numbers is 11

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

prints "Odd".

Answer:-

shell script:-

#! /bin/bash

echo Enter a Number

read Num

if [ $((Num % 2)) -eq 0 ]

then

echo "even"

else

echo "odd"

fi

terminal command:-

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question5.sh

Enter a Number

2

even

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

Answer:-

shell script:-

#!/bin/bash

for i in {1..5}

do

echo $i

done

terminal command:-

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question6.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question6.sh

1

2

3

4

5

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

Answer:-

shell script:-

#!/bin/bash

i=1

while [ $i -le 5 ]

do

echo $i

i=$((i + 1))

done

terminal commands:-

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question7.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question7.sh

1

2

3

4

5

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".

Answer:-

shell script:-

#!/bin/bash

if [ -f "file.txt" ]; then

echo "File exists"

else

echo "File does not exist"

fi

terminal commands:-

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question8.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question8.sh

File does not exist

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.

Answer:-

shell script :-

#!/bin/bash

num=15

if [ $num -gt 10 ]; then

echo "The number is greater than 10"

else

echo "The number is not greater than 10"

fi

terminal commands:-

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question9.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question9.sh

The number is greater than 10

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.

Answer:-

shell script:-

#!/bin/bash

for i in {1..5}

do

for j in {1..5}

do

printf "%4d" $((i \* j))

done

echo

done

terminal commands:-

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question10.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question10.sh

1 2 3 4 5

2 4 6 8 10

3 6 9 12 15

4 8 12 16 20

5 10 15 20 25

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.

Answer:

shell script:

#! /bin/bash

while((1==1))

do

echo Enter a Number

read n

if(($n>=0))

then

echo $((n\*n))

else

break

fi

done

terminal command:

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ nano question11.sh

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$ bash question11.sh

Enter a Number

5

25

Enter a Number

-5

shubham@shubham-Lenovo-E41-15:~/LinuxAssignment/Assignment2$

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.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Process | Arrival Time | Burst time | Completion Time | Turn Around Time | Waiting Time |
| P1 | 0 | 5 | 5 | 5 | 0 |
| P2 | 1 | 3 | 8 | 7 | 4 |
| P3 | 2 | 6 | 14 | 12 | 6 |

1. 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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Process | Arrival Time | Burst Time | Completion Time | Turn Around Time |
| P1 | 0 | 3 | 3 | 3 |
| P2 | 1 | 5 | 13 | 12 |
| P3 | 2 | 1 | 4 | 2 |
| P4 | 3 | 4 | 8 | 5 |

Average Turn Around Time = 22/4 = 5.5

1. 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.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Process | Arrival Time | Burst Time | Priority | Completion Time | Turn Around Time | Waiting Time |
| P1 | 0 | 5 | 3 | 12 | 12 | 0 |
| P2 | 1 | 4 | 1 | 5 | 4 | 0 |
| P3 | 2 | 7 | 4 | 19 | 17 | 10 |
| P4 | 3 | 2 | 2 | 7 | 4 | 2 |

1. 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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Process | Arrival Time | Burst Time | Completion Time | Turn Around Time |
| P1 | 0 | 4 | 10 | 10 |
| P2 | 1 | 5 | 14 | 13 |
| P3 | 2 | 2 | 6 | 4 |
| P4 | 3 | 3 | 13 | 1 |

Average Turn Around Time = 37/4 = 9.25

1. 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?

The final value of x in both the parent and child processes will be 6.