**FUNDAMENTALS OF SHELL SCRIPTING:**

1. To know the default shell- **$0**.
2. Supported shells are present - **/etc/shells**.
3. **#!/bin/bash** – shebang line ( select the interpreter).
4. Run using-  
   ./<script name>.sh  
   /path/<script name.sh  
   bash ./<script name>.sh  
   sh <script name>.sh
5. Comments- # use for single line comment.  
   for multi line comment <<comment ………….. commnent.
6. Using variables – var-name= <value> and for string “<string name >”  
   \* variable has the property of overwrite itself.
7. Var used to store the output of a command.  
   HOST=$(hostname)  
   echo “Name of the machine is $HOST”
8. Variable is on read only mode – not overwrite  
   readonly var\_name= <value>.
9. Arrays - define an array   
   myarray=( 1 2 Hello “hey man”)  
     
   how to get values from an array-  
   echo “ ${myarray[0]}”  
   echo “ ${myarray[1}”  
   getting all arrays - echo “ ${myarray[\*]}”
10. Get the length of an array - echo “ ${#myarray[\*]}”
11. Get array from slicing  
    echo “ ${#myarray[\*]:2:2}”  
      
      
     two values from defined index.  
     starting from 2nd index .
12. Update array by adding some values -  
    myarray+= ( <value1> <value2> <value3> ).
13. Store the key value pairs-  
    declare -A <array name>  
    <array name> = ( [key]=” pair” [key]=” pair” )  
    echo “${myarray[key]}”
14. String Operations -   
    find length of a string abc = “……………”  
    abc\_length=${#abc}  
    echo “$ abc\_ length”  
    echo “for upper case ${abc^^}”  
    echo “for lower case ${abc ,,}”  
    echo “for replacement ${ abc/<old text>/<new text> )  
     echo “ for slicing ${abc:2:2}”  
      
      
     two values from defined index.  
     starting from 2nd index .
15. User interactions- read <var name>  
    (manual input from the user)
16. Using comment in read command   
    read -p <your text> <var name>
17. Airthmetic operations -   
    Using let command :- let a++ let a=5\*10.  
    using brackets (( a++ )) $(( a=5\*10 )).
18. Conditional Statements-  
    using if-else :  
    if [[ $marks -gt 40 ]]; then  
    do   
     echo “………”  
    done  
    fi (to end the if else statement , use opposite of if )
19. Using if-elif statement   
    if [[ $marks -gt 40 ]]; then  
    do   
     echo “………”  
     elif [[ ]]; then  
     echo “………..”  
    done  
    fi
20. Case :-   
    echo “…………..”  
      
    read choice   
    case $choice in   
    1) command or anything ;;  
    2) command or anything ;;   
    3) command or anything ;;  
    \*) echo “invalid output”   
    esac (opposite of case to indicate the end of syntx)
21. Case :-   
    echo “…………..”  
      
    read choice   
    case $choice in   
    1)   
     commands   
     echo “…………”  
     ;;  
    2) command or anything ;;   
    3) command or anything ;;  
    \*) echo “invalid output”   
    esac (opposite of case to indicate the end of syntx)
22. Using for loop -  
    with number range {1..20}.
23. Always take a file path in variable   
    file=”/path/”
24. Use variables with a command like-  
    var=$(cat $file).
25. Make a infinite loop with while  
    while true  
    do   
     echo “……..”  
    done
26. For (( ;; ))  
    do   
     echo “……..”  
    done
27. To read a content with CSV File :  
    while IFS=”,” read name fullname  
    do   
     echo “your name is $name”  
     echo “The fullname is $fullname”  
    done
28. #!/bin/bash
29. **Making a binary calculator** --

calc() {

echo "please choose a method for operations:"

echo "1. Addition"

echo "2. Subtraction"

echo "3. Multiplication"

echo "4. Division"

read -p "choose your method:" choice

if [[ $choice -gt 4 ]]; then

echo "please select choice index from above options"

else

read -p "choose first number:" num1

read -p "choose second number:" num2

case $choice in

1) result=$(( num1 + num2 ))

echo "your answer is $result"

;;

2) result=$(( num1 - num2 ))

echo "your answer is $result"

;;

3) result=$(( num1 \* num2 ))

echo "your answer is $result"

;;

4) result=$(( num1 / num2 ))

echo "your answer is $result"

;;

\*) echo "Invalid chocie"

esac

fi }.

1. Arguments in Script :-  
   Access the arguments  
   1. To get no of arguments $#  
   2. To display all arguments $@  
   3. To use or display an argument $1, $2….
2. Use exit to quit the command between loop.
3. basename /scripts/cfs/file1.txt gives only filename here .
4. dirname /scripts/cfs/file1.txt gives only directory names .
5. realpath <filename> to print complete path of that location.
6. **using string comparison**-

\* str1 = str2 equal to

\* str1 != str2 not equal to

\* str1 > str2 greater than

\* str1 < str2 less than

\* -n str if string has length greter than 0

\* -z str if string length is zero

**using file comparison** -

\* -d file check if file exist and is a directory.

\* -e file check if file exist

\* -f file check if file exist and is a file.

\* -r file check if file exist and is a readable.

\* -w file check if file exist and is a writable

\* -s file check if file exist and is not empty.

\* -x file check if file exist and is a executable.

\* -O file check if file exist and is owned by current user.

\* -G file check if file exist and default group is same as current user.

\* file1 -nt file2 check if file 1 is newer than file2.

\* file1 -ot file2 check if file 1 is older than file2.

36) redirect >  
append >>   
dev/null (vanished the output) &> send the error and output in dev/null file.

37 ) the name of the script - ${0}

38) Logging in Shell scripting - Using logger for print log messages   
  
logger “………..”  
then see the log in /var/log/messages.

39) Debugging in shell script --- use **set -x**  
  
#!/bin/bash  
  
set -x   
<……………………………………………  
……………………………>

40) if command fail it will exit **set -e**

41) for running in script in background **nohup ./<script name> &**

42) for scheduling the command only onetime -----  
using AT Command  
  
at 12.09 PM < date > (optional)   
then provide your script or command --- bash <script name >  
ctrl +d for save   
  
atq to check the jobs  
atrm <job id > to remove the job.