**1.Write a shell script to perform basic floating point arithmetic operations between two nos.**

Program:

#!/bin/bash  
#1.Write a shell script to perform basic floating point arithmetic operations between two nos.  
echo "Enter two numbers"  
read a b  
echo "Press 1 for addition"  
echo "Press 2 for subtraction"  
echo "Press 3 for multiplication"  
echo "Press 4 for division"  
read ch  
case $ch in  
 1) s=$(echo "scale=2;($a+$b)"|bc);;  
 2) s=$(echo "scale=2;($a-$b)"|bc);;  
 3) s=$(echo "scale=2;($a\*$b)"|bc);;  
 4) s=$(echo "scale=2;($a/$b)"|bc);;  
 \*) echo "Invalid choice!"  
esac  
echo "Result : $s"

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./calculator.sh  
Enter two numbers  
4 25  
Press 1 for addition  
Press 2 for subtraction  
Press 3 for multiplication  
Press 4 for division  
1  
Result : 29  
user@Gigabyte:/mnt/d/documents/assignment$ ./calculator.sh  
Enter two numbers  
2 3.7  
Press 1 for addition  
Press 2 for subtraction  
Press 3 for multiplication  
Press 4 for division  
2  
Result : -1.7  
user@Gigabyte:/mnt/d/documents/assignment$ ./calculator.sh  
Enter two numbers  
4.2 100  
Press 1 for addition  
Press 2 for subtraction  
Press 3 for multiplication  
Press 4 for division  
3  
Result : 420.0  
user@Gigabyte:/mnt/d/documents/assignment$ ./calculator.sh  
Enter two numbers  
4 8  
Press 1 for addition  
Press 2 for subtraction  
Press 3 for multiplication  
Press 4 for division  
4  
Result : .50  
user@Gigabyte:/mnt/d/documents/assignment$

**2.Write a shell script / program to check whether the number is prime or not.**

Program:

#!/bin/bash  
#2.Write a shell script / program to check whether the number is prime or not.  
echo "Enter a number"  
read n  
c=0  
for ((i=1;i<=n;i++))  
do  
 if (($n%$i==0))  
 then  
 c=$(($c+1))  
 fi  
done  
if [ $c -eq 2 ]  
then  
 echo "Prime number"  
else  
 echo "Not a prime number"  
fi

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./prime\_check.sh  
Enter a number  
17  
Prime number  
user@Gigabyte:/mnt/d/documents/assignment$ ./prime\_check.sh  
Enter a number  
6  
Not a prime number  
user@Gigabyte:/mnt/d/documents/assignment$

**3.Write a shell script program to check if the Number is a Palindrome or not.**

Program:

#!/bin/bash  
#3.Write a shell script program to check if the Number is a Palindrome or not.  
echo "Enter a number"  
read n  
s=0  
w=$n  
while [ $w -ne 0 ]  
do  
 d=$(($w%10))  
 s=$(($s\*10+$d))  
 w=$(($w/10))  
done  
if [ $s -eq $n ]  
then  
 echo "Palindrome number"  
else  
 echo "Not a palindrome number"  
fi

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./palindrome\_check.sh  
Enter a number  
12521  
Palindrome number  
user@Gigabyte:/mnt/d/documents/assignment$ ./palindrome\_check.sh  
Enter a number  
2453  
Not a palindrome number

user@Gigabyte:/mnt/d/documents/assignment$

**4.Write a shell script to find a number within an array using binary search.**

Program:

#!/bin/bash  
#4.Write a shell script to find a number within an array using binary search.  
declare -a arr  
echo "Enter the number of array elements"  
read n  
echo "Enter array elements"  
for ((i=0;i<n;i++))  
do  
 read arr[$i]  
done  
for ((i=0;i<n-1;i++))  
do  
 for ((j=0;j<n-i-1;j++))  
 do  
 if [ ${arr[$j]} -gt ${arr[$j+1]} ]  
 then  
 temp=${arr[$j]}  
 arr[$j]=${arr[$j+1]}  
 arr[$j+1]=$temp  
 fi  
 done  
done

echo "Elements in sorted order"  
for ((i=0;i<n;i++))  
do  
 echo "${arr[$i]}"  
done  
echo "Enter the number to search"  
read s  
lb=0

ub=$(($n-1))  
flag=0  
while [ $lb -le $ub ]  
do  
 mid=$((($ub+$lb)/2))  
 if [ $s -gt ${arr[$mid]} ]  
 then  
 lb=$(($mid+1))  
 elif [ $s -lt ${arr[$mid]} ]  
 then  
 ub=$(($mid-1))  
 else  
 echo "$s is present at position $(($mid+1))"

flag=1  
 break  
 fi  
done  
if [ $flag -eq 0 ]  
then  
 echo "$s is not present"  
fi

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./binary\_search.sh  
Enter the number of array elements  
5  
Enter array elements  
8  
4  
2  
7  
0  
Elements in sorted order  
0  
2  
4  
7  
8  
Enter the number to search  
7  
7 is present at position 4  
user@Gigabyte:/mnt/d/documents/assignment$ ./binary\_search.sh  
Enter the number of array elements  
5  
Enter array elements  
3  
2  
5  
1  
9  
Elements in sorted order  
1  
2  
3  
5  
9  
Enter the number to search  
10  
10 is not present  
user@Gigabyte:/mnt/d/documents/assignment$

**5.Write a shell script to sort a set of integer numbers using bubble sort.**

Program:

#!/bin/bash  
#5.Write a shell script to sort a set of integer numbers using bubble sort.  
declare -a arr  
echo "Enter the number of array elements"  
read n  
echo "Enter array elements"  
for ((i=0;i<n;i++))  
do  
 read arr[$i]  
done  
for ((i=0;i<n-1;i++))  
do  
 for ((j=0;j<n-i-1;j++))  
 do  
 if [ ${arr[$j]} -gt ${arr[$j+1]} ]  
 then  
 temp=${arr[$j]}  
 arr[$j]=${arr[$j+1]}  
 arr[$j+1]=$temp  
 fi  
 done  
done  
echo "Elements in sorted order"  
for ((i=0;i<n;i++))  
do  
 echo "${arr[$i]}"  
done

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./bubble\_sort.sh  
Enter the number of array elements  
6  
Enter array elements  
11  
23  
4  
16  
2  
10  
Elements in sorted order  
2  
4  
10  
11  
16  
23  
user@Gigabyte:/mnt/d/documents/assignment$

**6.Write a shell script to find the frequency of vowels in a given string. String should be taken as input at runtime.**

Program:

#!/bin/bash  
#6.Write a shell script to find the frequency of vowels in a given string. String should be taken as input at runtime.  
echo "Enter a string"  
read str  
count=$(echo $str | grep -o -i "[aeiou]" | wc -l)  
echo "Frequency of vowels in '$str' : $count"

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./vowel\_frequency.sh  
Enter a string  
Hello World! I am Agnibhu.  
Frequency of vowels in 'Hello World! I am Agnibhu.' : 8  
user@Gigabyte:/mnt/d/documents/assignment$

**7.Write a shell script that takes a command line argument and reports on whether it is directory, a file, or something else.**

Program:

#!/bin/bash  
#7.Write a shell script that takes a command line argument and reports on whether it is directory, a file, or something else.  
if [ $# -ne 1 ]  
then  
 echo "Invalid number of arguments"  
 exit  
fi  
if [ -f $1 ]  
then  
 echo "$1 is a file"  
elif [ -d $1 ]  
then  
 echo "$1 is a directory"  
else  
 echo "$1 is something else"  
fi

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./directory\_or\_file.sh TODO.txt  
TODO.txt is a file  
user@Gigabyte:/mnt/d/documents/assignment$ ./directory\_or\_file.sh test  
test is a directory  
user@Gigabyte:/mnt/d/documents/assignment$ ./directory\_or\_file.sh Not\_present.txt  
Not\_present.txt is something else  
user@Gigabyte:/mnt/d/documents/assignment$

**8.Write a shell script to accept a string in the command line and reverse the same string.**

Program:

#!/bin/bash  
#8.Write a shell script to accept a string in the command line and reverse the same string.  
if [ $# -ne 1 ]  
then  
 echo "Invalid number of arguments"  
 exit  
fi  
str=""  
for ((i=0;i<${#1};i++))  
do  
 ch=${1:i:1}  
 str=$ch$str  
done  
echo "Reversed string : $str"

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./reverse\_string.sh Hello\_World!  
Reversed string : !dlroW\_olleH  
user@Gigabyte:/mnt/d/documents/assignment$

**9.Write a shell script to display a name in abbreviation form using command line arguments.**

Program:

#!/bin/bash  
#9.Write a shell script to display a name in abbreviation form using command line arguments.  
nstr=""  
if [ $# -eq 0 ]  
then  
 echo "No arguments given"  
 exit  
fi  
for item in $@  
do  
 ch=${item:0:1}  
 nstr=$nstr$ch.  
done  
echo "Abbreviated form : $nstr"

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./name\_abbreviation.sh Steven Paul Jobs  
Abbreviated form : S.P.J.  
user@Gigabyte:/mnt/d/documents/assignment$

**10.Develop an interactive script that asks for a word and a file name and then tells how many times that word occurred in the file.**

Program:

#!/bin/bash  
#10.Develop an interactive script that asks for a word and a file name and then tells how many times that word occurred in the file.  
echo "Enter filename"  
read file  
if [ ! -f $file ]  
then  
 echo "File does not exist"  
else  
 echo "Enter word"  
 read word  
 echo -n "Number of occurrences of '$word' in $file : "  
 grep -o -i $word $file | wc -l  
fi

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./file\_word\_frequency.sh  
Enter filename  
unix.txt  
Enter word  
unix  
Number of occurrences of 'unix' in unix.txt : 5  
user@Gigabyte:/mnt/d/documents/assignment$ ./file\_word\_frequency.sh  
Enter filename  
not\_present.txt  
File does not exist

user@Gigabyte:/mnt/d/documents/assignment$

*Contents of unix.txt:*

Unix (trademarked as UNIX) is a family of multitasking, multiuser computer operating systems.  
Unix derives from the original AT&T Unix, whose development started in 1969.  
Unix was developed at the Bell Labs research center by Ken Thompson, Dennis Ritchie, and others.

**11. Write a shell script that receives two file names as arguments. It should check whether the two file contents are same or not. If they are the same then the second file should be deleted.**

Program:

#!/bin/bash  
#11. Write a shell script that receives two file names as arguments. It should check whether the two file contents are same or not. If they are the same then the second file should be deleted.  
if [ $# -ne 2 ]  
then  
 echo "Invalid number of arguments"  
 echo "Usage: ./del\_duplicate\_file.sh file1.txt file2.txt"  
 exit  
fi  
if [ $1 -ef $2 ]  
then  
 echo "Same file given as arguments"  
elif [ ! -f $1 ] || [ ! -f $2 ]  
then  
 echo "File doesn't exist"  
elif cmp -s $1 $2  
then  
 echo "Contents of $1 and $2 are same"  
 rm $2  
 echo "File $2 has been deleted"  
else  
 echo "Contents of $1 and $2 are different"  
fi

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ls \*.txt  
data.txt t1.txt t2.txt t3.txt TODO.txt unix.txt  
user@Gigabyte:/mnt/d/documents/assignment$ ./del\_duplicate\_file.sh t2.txt t3.txt  
Contents of t2.txt and t3.txt are same  
File t3.txt has been deleted

user@Gigabyte:/mnt/d/documents/assignment$ ls \*.txt  
data.txt t1.txt t2.txt TODO.txt unix.txt  
user@Gigabyte:/mnt/d/documents/assignment$ ./del\_duplicate\_file.sh t1.txt t2.txt  
Contents of t1.txt and t2.txt are different  
user@Gigabyte:/mnt/d/documents/assignment$ ./del\_duplicate\_file.sh t1.txt t1.txt  
Same file given as arguments  
user@Gigabyte:/mnt/d/documents/assignment$ ./del\_duplicate\_file.sh t1.txt t3.txt  
File doesn't exist  
user@Gigabyte:/mnt/d/documents/assignment$ ./del\_duplicate\_file.sh  
Invalid number of arguments  
Usage: ./del\_duplicate\_file.sh file1.txt file2.txt  
user@Gigabyte:/mnt/d/documents/assignment$

*Contents of t1.txt:*

Dolor  
Sit  
Amet

*Contents of t2.txt:*

Lorem  
Ipsum

*Contents of t3.txt:*

Lorem  
Ipsum

**12. Write a shell script for menu based system to insert records for employees with an employee ID, name, designation, salary in a data file, also display records when necessary. Display salary for the employee asked.**

Program:

#!/bin/bash  
#12. Write a shell script for menu based system to insert records for employees with an employee ID, name, designation, salary in a data file, also display records when necessary. Display salary for the employee asked.

file="data.txt"  
if [ ! -s $file ]  
then  
 echo "Employee ID|Name|Designation|Salary" >> $file  
fi  
while [ true ]  
do  
 echo "===== MENU ====="  
 echo "1. Insert record"  
 echo "2. Display records"  
 echo "3. Display salary"  
 echo "4. Exit"  
 echo "Enter choice"  
 read ch  
 case $ch in  
 1)  
 echo "Enter employee ID"  
 read id  
 echo "Enter employee name"  
 read name  
 echo "Enter designation"  
 read des  
 echo "Enter salary"  
 read sal  
 echo "$id|$name|$des|$sal" >> $file  
 echo "Record added : $id|$name|$des|$sal";;  
 2)  
 cat $file;;  
 3)  
 echo "Enter employee ID"  
 read eid  
 total=`cat $file | wc -l`  
 flag=0  
 for ((i=2;i<=total;i++))  
 do  
 record=`cat $file | head -$i | tail -1`

match=`echo $record | grep -wci $eid`  
 if [ $match -eq 1 ]  
 then  
 emp\_name=`echo $record | cut -d "|" -f2`  
 emp\_sal=`echo $record | cut -d "|" -f4`  
 echo "Salary of $emp\_name : Rs.$emp\_sal"  
 flag=1  
 break  
 fi  
 done  
 if [ $flag -eq 0 ]  
 then  
 echo "No records found"  
 fi;;  
 4)  
 echo "Program terminated"  
 exit;;  
 \*)  
 echo "Invalid choice"  
 esac

done

Output:

user@Gigabyte:/mnt/d/documents/assignment$ ./employee\_database.sh  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
1  
Enter employee ID  
2315690  
Enter employee name  
David  
Enter designation  
Game Designer  
Enter salary  
45000  
Record added : 2315690|David|Game Designer|45000  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
1  
Enter employee ID  
2347619  
Enter employee name  
John  
Enter designation  
Software Developer  
Enter salary  
60000  
Record added : 2347619|John|Software Developer|60000  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
1  
Enter employee ID  
9175670  
Enter employee name  
Jane  
Enter designation  
Graphic Artist  
Enter salary  
65000  
Record added : 9175670|Jane|Graphic Artist|65000  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
1  
Enter employee ID  
5432178  
Enter employee name  
Mary  
Enter designation  
Data Analyst  
Enter salary  
75000  
Record added : 5432178|Mary|Data Analyst|75000  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
2  
Employee ID|Name|Designation|Salary  
2315690|David|Game Designer|45000  
2347619|John|Software Developer|60000  
9175670|Jane|Graphic Artist|65000  
5432178|Mary|Data Analyst|75000  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
3  
Enter employee ID  
9175670  
Salary of Jane : Rs.65000  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice

3  
Enter employee ID  
1234567  
No records found  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
0  
Invalid choice  
===== MENU =====  
1. Insert record  
2. Display records  
3. Display salary  
4. Exit  
Enter choice  
4  
Program terminated  
user@Gigabyte:/mnt/d/documents/assignment$

*Contents of data.txt:*

Employee ID|Name|Designation|Salary  
2315690|David|Game Designer|45000  
2347619|John|Software Developer|60000  
9175670|Jane|Graphic Artist|65000  
5432178|Mary|Data Analyst|75000