12a)

```
#!/bin/bash
#solicite al usuario 2 numeros, imprima la
suma,resta,multiplicacion, y cual
#es el mayor de los dos
echo -n "Ingrese primer numero: "
read n1
echo -n "Ingrese segundo numero: "
read n2
echo -n "La suma de $n1 mas $n2 es: "
echo `expr $n1 + $n2`
echo -n "La resta de $n1 menos $n2 es: "
echo `expr $n1 - $n2`
echo -n "La multiplicacion entre $n1 por $n2 es: "
echo `expr $n1 '*' $n2`
if [ $n1 -lt $n2 ]
then
        echo "El numero mayor es $n2"
else
        echo "El numero mayor es $n1"
fi
```

12b)

```
#!/bin/bash
# 2 numeros como parametro, imprima la
suma,resta,multiplicacion, y cual
#es el mayor de los dos
if [ $# -ne 0 ]
then
        echo -n "La suma de $1 mas $2 es: "
        echo `expr $1 + $2`
        echo -n "La resta de $1 menos $2 es: "
        echo `expr $1 - $2`
        echo -n "La multiplicacion entre $1 por $2 es: "
        echo `expr $1 '*' $2`
        if [ $1 -lt $2 ]
        then
                echo "El numero mayor es $2"
        else
                echo "El numero mayor es $1"
        fi
else
        echo "No pasaste ningun argumento"
fi
```

13a)

```
#!/bin/bash
#visualizar en pantalla los numeros del 1 al 100 asi
como sus cuadrados
for ((i=0;i<=100;i++))
do
        echo "Numero $i y su cuadrado `expr $i '*' $i`"
done
```

13b)

```
#!/bin/bash
#mostrar 3 opciones, segun la elegida listo, pwd, y
quien
echo "1) Listar el contenido del directorio actual"
echo "2) El nombre del directorio actual"
echo "3) Quien esta logeado al sistema"
echo -ne "Ingrese numero de opcion: "
read opcion
case $opcion in
        1)
                echo "El contenido del directorio
actual es `ls -l`"
        ;;
        2 )
                echo "La ruta del directorio actual es
`pwd`"
        ;;
        3)
                echo "Estas logeado al sistema como
`whoami`"
        ;;
        *)
                echo "Opcion incorrecta"
        ;;
esac
```

13c)

```
#!/bin/bash
#recibe como parametro un nombre de archivo e
informa si existe o no.
#si exsite muestra si es directorio o archivo
#si no existe el archivo/directorio crea un directorio
con el nombre recibido como parametro
if [ $# -ne 0 ]
then
        if [ -f $1 ]
        then
```

```
                echo "Existe y es un archivo"
        elif [ -d $1 ]
        then
                echo "Existe y es un directorio"
        else
                echo "No existe archivo/directorio.
Creando directorio $1"
        fi

else
        echo "Debe pasar un nombre de
archivo/directorio como parametro"
fi
```

14)

```
#!/bin/bash
#dados dos vectores de longitud igual, pero no se
conocen, sumar elemento a elemento e imprimir
vector1=(1 15 9 20)
vector2=(3 1 10 40)

for((i=0; i<${#vector1[@]}; i++))
do
        echo "La suma de los elementos de la posicion
$i es : `expr ${vector1[$i]} + ${vector2[$i]}`"
done
```

14)

```
#!/bin/bash
#renombra archivos, recibe el directorio pasado como
parametro
#-a CADENA al final
#-b CADENA al principio
recorrer ()
{
        cd $1
        for i in `ls`
        do
                if [ -f "$i" ]
                then
                        renombrar $2 $3 "$i"
                fi

    done

}

renombrar()
{
        if [ $1 = "-a" ]
        then
                mv $3 ${3}${2}
        else
                mv $3 ${2}${3}
        fi
}

if [ $# -eq 3 ]
then
        recorrer $1 $2 $3
else
        echo "Cantidad incorrecta de parametros"
fi
```

15)

```
#!/bin/bash
#cut
#ls -l | cut -d " " -f 1,2 muestra columnas 1 y 2
delimitados por " "
#ls -l | tr -s " " | cut -d " " -f 1,2 el tr me elimina " "
repetidos y me deja solo uno
        echo `man cut`
```

16)

```
#!/bin/bash
#recivo por parametro una extension, y hace un
reporte con 2 columnas, el nombre de user y la
cantidad de archivos
#que posee con esa extension. Se guarda en un archivo
llamado reporte.txt
# ls -l | cut -d " " -f 3  -> se queda con la columna de
usuario tras un ls -l
#awk '{print $3,$4}' ls.txt imprime la columna 3 y 4 del
archivo ls.txt

#`ls -l > ls.txt`
        #echo `awk '{print $3}' ls.txt | wc -l`
        #echo "$3"

# ls -l | cut -d " " -f 3  -> se queda con la columna de
usuario tras un ls -l
```

```
#ls -l | tr -s " " | cut -d " " -f 3 te elimina los doble
espacio "  "
#ls -l | grep '.sh' | wc -l me quedo con la cantidad de
lineas .sh

if [ $# -eq 1 ]
then
	path=`pwd`
	string=
	cantidad=
	cd /home
	for user in `ls -l | tr -s " " | cut -d " " -f 3`
	do
		cd /home
		cd $user
		string=$user
		cantidad=`ls -l | grep $1 | wc -l`
		echo $string"    "$cantidad >>
$path"/reporte.txt"
	done
else
	echo "No pasaste ningun parametro"
fi

18)
#!/bin/bash
#verificar cada 10 seg si un usuario se ha logueado en
el sistema, el nombre viene por parametro
#cuando se loguea debe mostrar  "Usuario $1 logueado
en el sistema"

control()
{
	for user in `who | cut -d " " -f1`
	do
		if [ $user == $1 ]
		then
			echo "Usuario $1 logueado
en el sistema"
			exit
		fi
	done

}

if [ $# -eq 1 ]
then
	control $1
	while [ 0 -lt 1 ]
	do
		sleep 10; control $1
	done
else
	echo "Debe pasar un nombre de usuario"

fi

19)
#!/bin/bash
#muestra un menu con la selecciona de cada uno de los
scripts creados.
#el menu debe permanecer activo hasta que se
precione salir

while [ 0 -ne 1 ]
do
	echo "MENU DE COMANDOS"
	echo "01. Script 1"
	echo "00. Salir"
	echo -ne "Ingrese numero de opcion a
ejecutar: "
	read opcion
	case $opcion in
		01)
			echo "eligio la opcion 01"
		;;

		00)
			exit
		;;

		*)
		;;
	esac
done

20)
#!/bin/bash
#el comportamiento de una pila
#21) Dentro del mismo script y utilizando las funciones
implementadas agregue 10
```

```
#elementos a la pila, saque 3 de ellos, imprima la
longitud de la cola y luego la totalidad
#de los elementos que en ella se encuentran.
vector=()
index=0

print()
{
        for elem in ${vector[*]}
        do
                echo $elem
        done
}

function length
{
        return `expr $index + 1`
}

pop()
{
        unset vector[index]
        let index--
}

push()
{
        vector[index]=$1
        let index++
}

push 1
push 2
push 3
push 4
push 5
push 6
push 7
push 8
push 9
push 0
#el primer pop no hace nada
pop
pop
pop
pop
print
#imprimir valor de funcion
```

22)

```
#!/bin/bash
#dado el array multiplicarlo atravez de una funcuion
num=(11 3 5 7 9 3 5 4)
productoria()
{
        resultado=1
        for x in ${num[*]}
        do
                #echo $x
                resultado=`expr ${resultado} '*' ${x}`
        done
        echo "El resultado de la multiplicacion es:
$resultado"

}

productoria ${num[*]}
23)
#!/bin/bash
#recorre un arreglo e imprime los numeros pares, y
cuenta los numeros impares

vector=(1 2 3 4 5 6 7 8 9 10)

impares=

for elem in ${vector[@]}
do
        mod=`expr $elem % 2`
        if [ $mod -eq 0 ]
        then
                echo $elem
        else
                let impares++
        fi
done
echo "La cantidad de numeros impares es $impares"

#!/bin/bash
#dados dos vectores de longitud igual, pero no se
conocen, sumar elemento a elemento e imprimir

```

```
vector1=(1 15 9 20)
vector2=(3 1 10 40)

for((i=0; i<${#vector1[@]}; i++))
do
        echo "La suma de los elementos de la posicion
$i es : `expr ${vector1[$i]} + ${vector2[$i]}`"
done
```

Calculadora)

```
#!/bin/bash
#calculadora, recibe la operacion y los numeros por
parametro
#ejemplo: 2 + 2

if [ $# -eq 3 ]
then
        case $2 in
        +)
                echo "La suma entre $1 y $3 es `expr
$1 + $3`"
        ;;
        -)
                echo "La resta entre $1 y $3 es `expr
$1 - $3`"
        ;;
        '*')
                echo "La multiplicacion entre $1 y $3
es `expr $1 '*' $3`"
        ;;
        %)
                echo "La division entre $1 y $3 es
`expr $1 / $3`"
        ;;
        *)
        ;;
        esac
else
        echo "Numero de parametros incorrectos.
Ejemplo: 2 + 2"
fi
```

ASCII)

```
#!/bin/bash
#identificar archivos ASCII dentro de los home de
usuario

virus=0
cd /home
for user in `ls`
do
        cd /home
        cd $user
        for archivo in `ls | grep ".txt"`
        do
                if [ `file $archivo | grep ASCII` == ]
                then
                        echo "Es un virus $archivo"
                        let virus++
                fi
        done
done
echo "La cantidad de archivos infectados eliminados es:
$virus"
```

funcion)

```
#!/bin/bash
# imprimir lo que devuelve una funcion

function funcion
{
        return 1
}

funcion
echo $?
```

mover)

```
#!/bin/bash
#implemente el comando mover. Recibe dos
parametros, el primero el origen y el segundo el
destino
#debera mantener un archivo de log en /var/log/mover
donde por cada invocacion del script
#debera escribir una linea que contenga: "El archivo
<nombre> ha sido movido a <destino>


if [ $# -eq 2 ]
then
        if [ -e $1 ]
        then
```

```
459                     mv $1 $2
460                     echo "El archivo $1 ha sido movido a
461 $2" >> /home/frank/Escritorio/mover
462                     exit 0
463             else
464                     echo No existe el archivo/directorio
465 origen
466                     exit 1
467             fi
468
469 else
470             echo Numero de parametros incorrectos
471             exit 1
472 fi
473
474 buscar)
475 #!/bin/bash
476 #dar dos opciones, ingrese el patron a buscar y el
477 nombre del archivo destino
478 #busca en todo /etc los archivos que concuerden con
479 $1 y los empaquete y comprime en un unico archivo
480 #con nombre de $2
481
482 echo -n "Ingrese patron a buscar: "
483 read patron
484 echo -n "Ingrese el nombre del archivo destino: "
485 read archivoDestino
486 cd /etc
487 tar -cvzf $archivoDestino `ls | grep $patron`
```