

SHELL SCRIPTING

1. (output to terminal) Write a script to print:

a. "Welcome to Intelligrape"

`echo Welcome To Intelligrape` is command to print

```
fahad@fahad ~/shell <master*>
> cat >> print.sh
#!/bin/bash
echo Welcome To Intelligrape
^C
fahad@fahad ~/shell <master*>
> chmod +x print.sh
fahad@fahad ~/shell <master*>
> ./print.sh
Welcome To Intelligrape
fahad@fahad ~/shell <master*>
> █
```

b. <username>@<hostname>:<your present working directory>

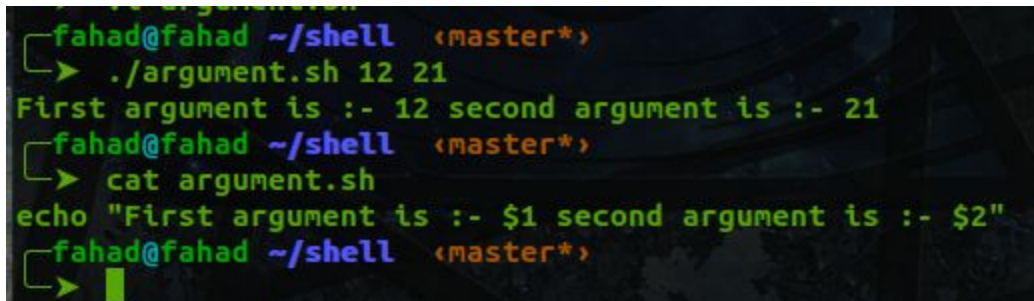
`echo `uname -n`@`hostname` : `pwd`` is a command to show this

```
fahad@fahad ~/shell <master*>
> cat >> print.sh
echo `uname -n`@`hostname` : `pwd`
^C
fahad@fahad ~/shell <master*>
> ./print.sh
Welcome To Intelligrape
fahad@fahad : /home/fahad/shell
fahad@fahad ~/shell <master*>
> █
```

2 (arguments) Write a script

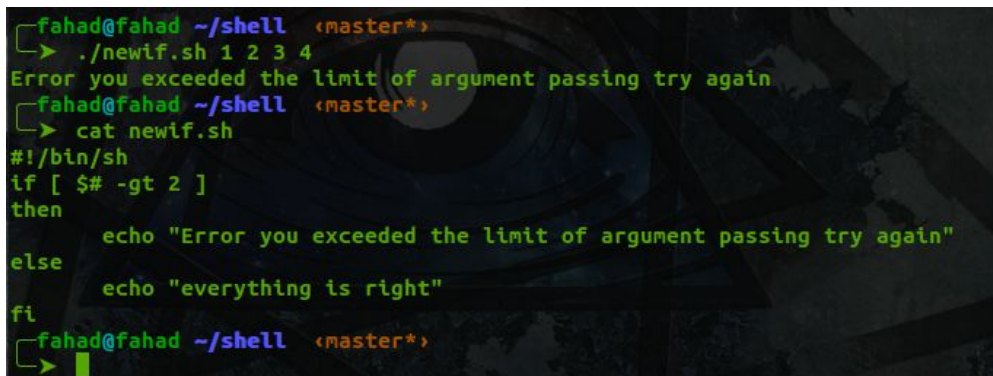
a. which takes in two arguments and print those arguments.

```
echo -n "Enter number : "  
read n  
echo -n "Enter another number"  
read m  
echo "First no. $n second no. is $m"
```

A terminal window showing the execution of a script named 'argument.sh'. The user runs './argument.sh 12 21' and the script outputs 'First argument is :- 12 second argument is :- 21'. Then, the user runs 'cat argument.sh' and the script content is displayed: 'echo "First argument is :- \$1 second argument is :- \$2"'.

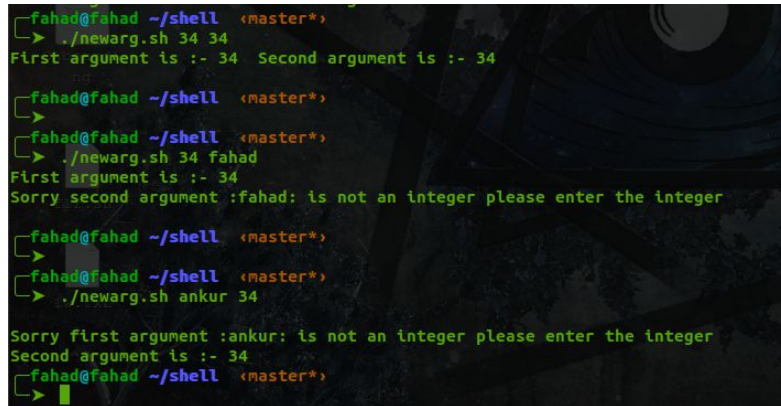
```
fahad@fahad ~/shell (master*)  
[> ./argument.sh 12 21  
First argument is :- 12 second argument is :- 21  
fahad@fahad ~/shell (master*)  
[> cat argument.sh  
echo "First argument is :- $1 second argument is :- $2"  
fahad@fahad ~/shell (master*)  
[> ]
```

b. which checks the number of arguments passed and if the number is greater than two print ERROR message along with printing the number of arguments.

A terminal window showing the execution of a script named 'newif.sh'. The user runs './newif.sh 1 2 3 4' and the script outputs 'Error you exceeded the limit of argument passing try again'. Then, the user runs 'cat newif.sh' and the script content is displayed: '#!/bin/sh', 'if [\$# -gt 2]', 'then', 'echo "Error you exceeded the limit of argument passing try again"', 'else', 'echo "everything is right"', 'fi'.

```
fahad@fahad ~/shell (master*)  
[> ./newif.sh 1 2 3 4  
Error you exceeded the limit of argument passing try again  
fahad@fahad ~/shell (master*)  
[> cat newif.sh  
#!/bin/sh  
if [ $# -gt 2 ]  
then  
    echo "Error you exceeded the limit of argument passing try again"  
else  
    echo "everything is right"  
fi  
fahad@fahad ~/shell (master*)  
[> ]
```

3. Continue with the above script
- check the two arguments are only integer values and if these are not integers print the proper error on terminal and also log it into a file.



```
fahad@fahad ~/shell <master*>
-> ./newarg.sh 34 34
First argument is :- 34 Second argument is :- 34

fahad@fahad ~/shell <master*>
->
fahad@fahad ~/shell <master*>
-> ./newarg.sh 34 fahad
First argument is :- 34
Sorry second argument :fahad: is not an integer please enter the integer

fahad@fahad ~/shell <master*>
->
fahad@fahad ~/shell <master*>
-> ./newarg.sh ankur 34
Sorry first argument :ankur: is not an integer please enter the integer
Second argument is :- 34

fahad@fahad ~/shell <master*>
->
```

```
#!/bin/bash
if ! [ "$1" -eq "$1" ] 2> /dev/null
then
    echo "\nSorry first argument :$1: is not an integer please
enter the integer" >> '/tmp/mylogs'
    echo -e "\nSorry first argument :$1: is not an integer please
enter the integer"
else
    echo -n "First argument is :- $1 "
fi
if ! [ "$2" -eq "$2" ] 2> /dev/null
then
    echo -e "\nSorry second argument :$2: is not an integer
please enter the integer" >> '/tmp/mylogs'
    echo -e "\nSorry second argument :$2: is not an integer
please enter the integer"
else
    echo "Second argument is :- $2"
```

```

fahad@fahad ~/shell (master*)
> cat newarg.sh
#!/bin/bash

if ! [ "$1" -eq "$1" ] 2> /dev/null
then
    echo "\nSorry first argument :$1: is not an integer please enter the integer" >> '/tmp/mylogs'
    echo -e "\nSorry first argument :$1: is not an integer please enter the integer"
else
    echo -n "First argument is :- $1 "
fi
if ! [ "$2" -eq "$2" ] 2> /dev/null
then
    echo -e "\nSorry second argument :$2: is not an integer please enter the integer" >> '/tmp/mylogs'
    echo -e "\nSorry second argument :$2: is not an integer please enter the integer"
else
    echo "Second argument is :- $2"
fi
fahad@fahad ~/shell (master*)
>

```

Logs are store in a [/tmp/mylogs](#)

```

fahad@fahad /tmp
> cat mylogs
Sorry :a: is not an integer
Sorry :bcd: is not an integer please enter the integer
Sorry:fahad: is not an integer please enter the integer
Sorry:ankur: is not an integer please enter the integer
Sorry :t34ak: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry :3: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry :av: is not an integer please enter the integer
Sorry first argument :av: is not an integer please enter the integer
Sorry second argument :av: is not an integer please enter the integer
Sorry first argument :av: is not an integer please enter the integer
Sorry second argument :av: is not an integer please enter the integer
Sorry second argument :av: is not an integer please enter the integer
Sorry first argument :av: is not an integer please enter the integer
Sorry first argument :av: is not an integer please enter the integer
\nSorry first argument :av: is not an integer please enter the integer
\nSorry first argument :av: is not an integer please enter the integer

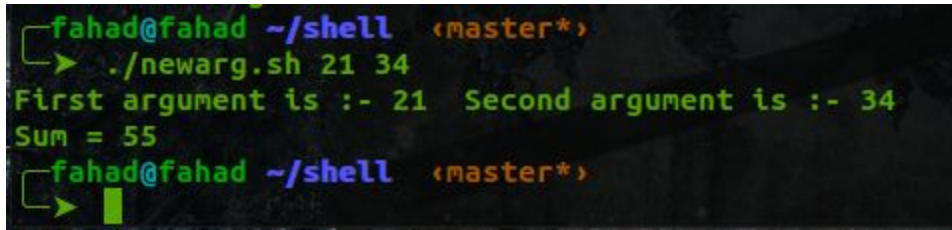
Sorry second argument :av: is not an integer please enter the integer

Sorry second argument :fahad: is not an integer please enter the integer
\nSorry first argument :ankur: is not an integer please enter the integer
fahad@fahad /tmp
>

```

b. perform addition on the two arguments and print result on screen. Use function for this.

```
function add()
{
    sum=$(( $1 + $2 ))
    echo "Sum = $sum"
}
add $1 $2
```

A terminal window with a dark background. The prompt is 'fahad@fahad ~/shell <master*>'. The user enters './newarg.sh 21 34'. The output is 'First argument is :- 21 Second argument is :- 34' followed by 'Sum = 55'. The prompt returns to 'fahad@fahad ~/shell <master*>' with a cursor on a new line.

```
fahad@fahad ~/shell <master*>
> ./newarg.sh 21 34
First argument is :- 21 Second argument is :- 34
Sum = 55
fahad@fahad ~/shell <master*>
> 
```

4. Create a calculator using the above script which would perform addition, subtraction, division and multiplication.

```
#!/bin/bash
echo "Enter First numbers : "
read a
echo "Enter Second number : "
read b

echo "Enter Choice : "
echo "1. Addition"
echo "2. Subtraction"
echo "3. Multiplication"
echo "4. Division"
read opr

if [ $opr = "1" ]
then
    echo $((a+b))
elif [ $opr = "2" ]
then
    echo $((a-b))
elif [ $opr = "3" ]
then
    echo $((a*b))
elif [ $opr = "4" ]
then
    echo $((a/b))
```



```
fahad@fahad ~/shell (master*)
└─> ./calculator.sh
Enter First numbers :
3
Enter Second number :
2
Enter Choice :
1. Addition
2. Subtraction
3. Multiplication
4. Division
2
Result : 1
fahad@fahad ~/shell (master*)
└─> ./calculator.sh
Enter First numbers :
4
Enter Second number :
5
Enter Choice :
1. Addition
2. Subtraction
3. Multiplication
4. Division
3
Result : 20
fahad@fahad ~/shell (master*)
└─> ./calculator.sh
Enter First numbers :
5
Enter Second number :
9
Enter Choice :
1. Addition
2. Subtraction
3. Multiplication
4. Division
4
Result : .55
fahad@fahad ~/shell (master*)
└─> █
```

a. the script should ask user which operation the user wants to perform:+,-,*,/

```
#!/bin/bash
echo "Enter First numbers : "
read a
echo "Enter Second number : "
read b
echo "Enter Choice : "
echo "+. Addition"
echo "-. Subtraction"
echo "*. Multiplication"
echo "/. Division"
read ch
case $ch in
  +)res=`echo $a + $b | bc`;;
  -)res=`echo $a - $b | bc`;;
  /)res=`echo "scale=2; $a / $b" | bc`;;
  *)res=`echo $a \* $b | bc`;;
esac
echo "Result : $res"
```

```
fahad@fahad ~/shell <master*>  
[> vi calculator.sh  
fahad@fahad ~/shell <master*>  
[> ./calculator.sh  
Enter First numbers :  
2  
Enter Second number :  
3  
Enter Choice :  
+. Addition  
-. Subtraction  
*. Multiplication  
/. Division  
+  
Result : 5  
fahad@fahad ~/shell <master*>  
[> ./calculator.sh  
Enter First numbers :  
3  
Enter Second number :  
2  
Enter Choice :  
+. Addition  
-. Subtraction  
*. Multiplication  
/. Division  
-  
Result : 1  
fahad@fahad ~/shell <master*>  
[> ./calculator.sh  
Enter First numbers :  
3  
Enter Second number :  
4  
Enter Choice :  
+. Addition  
-. Subtraction  
*. Multiplication  
/. Division  
*  
Result : 12
```

b. if user enters other than "+, -, *, /", print a proper message on the terminal and keep on asking for correct input (use while loop to accomplish this).

```
fahad@fahad ~/shell <master*>
> ./cal.sh
Enter First numbers :
1
Enter Second number :
2
Enter Choice :
+ . Addition
- . Subtraction
* . Multiplication
/ . Division
@
@ IS NOT THE CORRECT OPERATOR KINDLY CORRECT IT AND RETRY
Enter y to continue n to exit
y
Enter First numbers :
1
Enter Second number :
2
Enter Choice :
+ . Addition
- . Subtraction
* . Multiplication
/ . Division
#
# IS NOT THE CORRECT OPERATOR KINDLY CORRECT IT AND RETRY
Enter y to continue n to exit
n
fahad@fahad ~/shell <master*>
> █
```


c. Use case statement instead of if.

```
fahad@fahad ~/shell (master*)  
> ./cal.sh  
Enter First numbers :  
1  
Enter Second number :  
2  
Enter Choice :  
+. Addition  
-. Subtraction  
*. Multiplication  
/. Division  
@  
@ IS NOT THE CORRECT OPERATOR KINDLY CORRECT IT AND RETRY  
Enter y to continue n to exit  
y  
Enter First numbers :  
1  
Enter Second number :  
2  
Enter Choice :  
+. Addition  
-. Subtraction  
*. Multiplication  
/. Division  
#  
# IS NOT THE CORRECT OPERATOR KINDLY CORRECT IT AND RETRY  
Enter y to continue n to exit  
n  
fahad@fahad ~/shell (master*)  
> █
```

```
#!/bin/bash  
while true;  
do  
    echo "Enter First numbers : "  
    read a  
    echo "Enter Second number : "  
    read b  
    echo "Enter Choice :"  
    echo "+. Addition"  
    echo "-. Subtraction"  
    echo "*. Multiplication"  
    echo "/. Division"  
    read ch  
    case $ch in  
        +)res=`echo $a + $b | bc`  
        ;;  
        -)res=`echo $a - $b | bc`  
        ;;  
        /)res=`echo "scale=2; $a / $b" | bc`  
        ;;  
        \*)res=`echo $a \* $b | bc`  
        ;;  
        *)echo "$ch IS NOT THE CORRECT OPERATOR KINDLY CORRECT IT AND RETRY"  
        ;;  
    esac  
done
```

```

        esac
        echo "Enter y to continue n to exit"
        read f
        if [ "$f" = "n" ]
        then
            exit
        fi
    done

```

5. Write proper help documentation and print it with -h for above script.

```

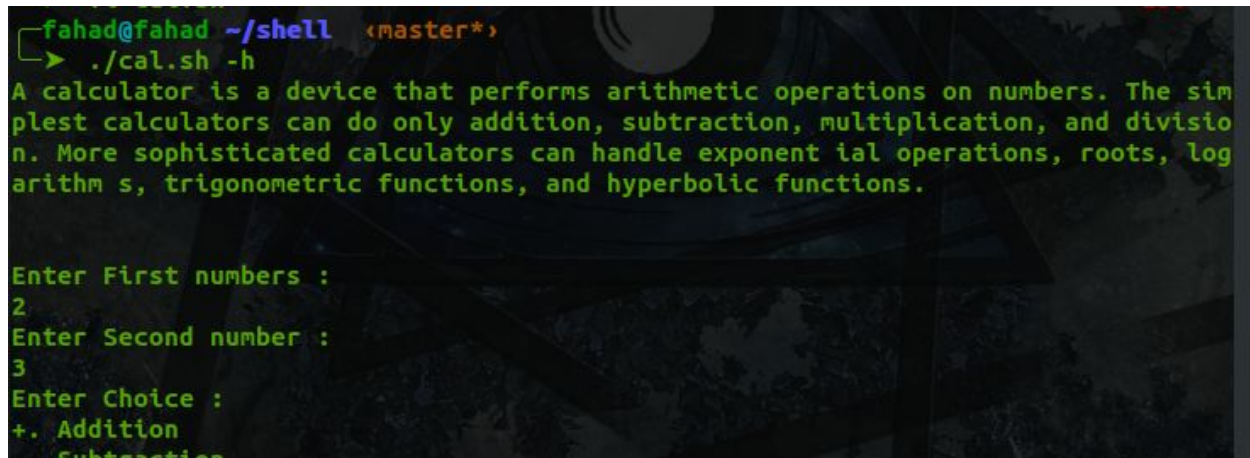
#!/bin/bash
if [ "$1" = "-h" ]
then
    echo "A calculator is a device that performs arithmetic operations on numbers.
The simplest calculators can do only addition, subtraction, multiplication, and division.
More sophisticated calculators can handle exponent ial operations, roots, logarithm s,
trigonometric functions, and hyperbolic functions.\n\n"
fi
while true;
do
    echo "Enter First numbers : "
    read a
    echo "Enter Second number : "
    read b
    echo "Enter Choice : "
    echo "+. Addition"
    echo "-. Subtraction"
    echo "*. Multiplication"
    echo "/. Division"
    read ch
    case $ch in
        +)res=`echo $a + $b | bc`
        ;;
        -)res=`echo $a - $b | bc`
        ;;
        /)res=`echo "scale=2; $a / $b" | bc`
        ;;
        \*)res=`echo $a \* $b | bc`
        ;;
        *)echo "$ch IS NOT THE CORRECT OPERATOR KINDLY CORRECT IT AND RETRY"
    esac
done

```

```

    ;;
    esac
    echo "Enter y to continue n to exit"
    read f
    if [ "$f" = "n" ]
    then
        exit
    fi
done

```



```

fahad@fahad ~/shell (master*)
> ./cal.sh -h
A calculator is a device that performs arithmetic operations on numbers. The simplest calculators can do only addition, subtraction, multiplication, and division. More sophisticated calculators can handle exponential operations, roots, logarithms, trigonometric functions, and hyperbolic functions.

Enter First numbers :
2
Enter Second number :
3
Enter Choice :
+. Addition
Subtraction

```

6. Create a script which takes input of "/etc/passwd" file and find out and print the sum of uids and gids. The script should tell which sum of greater.

```

#!/bin/bash
cat /etc/passwd|awk -F: '{print $3}' > uid_listing
uid=`awk '{ sum += $1 } END { print sum }' uid_listing`
cat /etc/passwd|awk -F: '{print $4}' > gid_listing
gid=`awk '{ sum += $1 } END { print sum }' gid_listing`
echo "SUM of UIDs :- $uid "
echo "SUM of GIDs :- $gid "
if [ "$uid" -gt "$gid" ]
then
    echo "SUM OF UIDs :- $uid is greater "
else
    echo "SUM OF GIDs :- $gid is greater "
fi

```

```

fahad@fahad ~/shell <master*>
> vi sum_u+g.sh
fahad@fahad ~/shell <master*>
> ./sum_u+g.sh
SUM of UIDs :- 72840
SUM of GIDs :- 465366
SUM OF GIDs :- 465366 is greater
fahad@fahad ~/shell <master*>
> █

```

7. A directory contains files and sub-directories. Move files to destination1 and directories to destination2

```

#!/bin/bash
for i in `ls`
do
    if [[ "$i" != "dst1" && "$i" != "dst2" && "$i" != "dectory.sh" ]]
    then
        if [ -f $i ]
        then
            mv $i dst1/$i
        fi
        if [ -d $i ]
        then
            mv $i dst2/$i
        fi
    fi
fi

```

Done

Before in shell directory

```

fahad@fahad ~/shell <master*>
> ls
'='  abc.sh      flr..      list.sh    sum_u+g.sh
1    argument.sh flr.sh     newarg.sh  t34
2    aws       flr..sh    newif.sh   t34ak
2]   aws-iam-authenticator for.sh     no_of_digit.sh uid_listing
3    calculator.sh  gid_listing print.sh
4    cal.sh        home_size.sh second_letter.sh
5    dectory.sh    if.sh      sed.sh
a    fahad         kubectl    sum.sh

```

After executing the **dectory.sh** script shell directory

```
dectory.sh dst1 dst2
fahad@fahad ~/shell <master*>
> ./dectory.sh
fahad@fahad ~/shell <master*>
> ls
dectory.sh dst1 dst2
fahad@fahad ~/shell <master*>
> ls dst1
'=' abc.sh flr.sh list.sh sum.sh
1 argument.sh flr..sh newarg.sh sum_u+g.sh
2 aws-iam-authenticator for.sh newif.sh t34
2] calculator.sh gid_listing no_of_digit.sh t34ak
3 cal.sh home_size.sh print.sh uid_listing
4 fahad if.sh second_letter.sh
5 flr.. kubectl sed.sh
```

8. Create a script which take three arguments, append first argument to every line in a file and second argument to the end of every line of the same file.

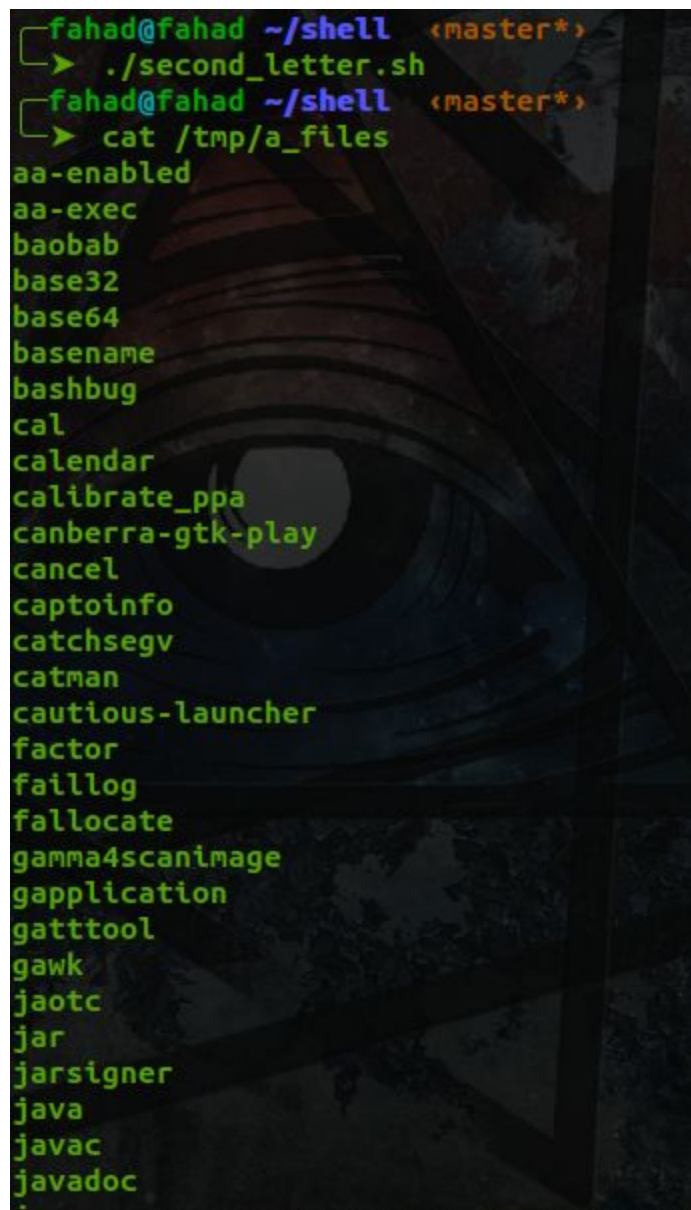
sed -i "s/^/\${1} / ; s\$/ \${2}/ " \$3

```
fahad@fahad ~/shell <master*>
> cat t34ak
WE
ARE
WATCHING
OUT
FOR
YOU
fahad@fahad ~/shell <master*>
> ./sed.sh fahad khan t34ak
fahad@fahad ~/shell <master*>
>
fahad@fahad ~/shell <master*>
> cat t34ak
fahad WE khan
fahad ARE khan
fahad WATCHING khan
fahad OUT khan
fahad FOR khan
fahad YOU khan
fahad@fahad ~/shell <master*>
> █
```


9. Make a list of files in /usr/bin that have the letter "a" as the second character. Put the result in a temporary file.

```
#!/bin/bash
```

```
ls /usr/bin | grep "^a" > /tmp/a_files
```



A terminal window screenshot showing the execution of a script. The prompt is 'fahad@fahad ~/shell <master*>'. The user runs './second_letter.sh'. The prompt remains the same. The user then runs 'cat /tmp/a_files'. The output is a list of files in /usr/bin where the second character is 'a':

```
aa-enabled
aa-exec
baobab
base32
base64
basename
bashbug
cal
calendar
calibrate_ppa
canberra-gtk-play
cancel
captaininfo
catchsegv
catman
cautious-launcher
factor
faillog
fallocate
gamma4scanimage
gappplication
gatttool
gawk
jaotc
jar
jarsigner
java
javac
javadoc
```

10. List all files in your home directory and print name and size in a table format.

```
#!/bin/bash
echo -e "NAME \t\t\t\t\t SIZE"
ls -l /home | awk '{printf "%-40s|%-18s\n", $9, $5}'
```



A terminal window showing the execution of a script. The prompt is 'fahad@fahad ~/shell <master*>'. The user enters './home_size.sh'. The output is a table with two columns: 'NAME' and 'SIZE'. The table lists four files: 'fahad' (4096), 'fahad1' (4096), 'lost+found' (16384), and 'test' (4096). The prompt returns to 'fahad@fahad ~/shell <master*>'.

NAME	SIZE
fahad	4096
fahad1	4096
lost+found	16384
test	4096