//1.sum of two no

```
#!/bin/sh
echo "enter the no"
read a
echo "enter the second no"
read b
sum=` expr $a + $b `
echo $sum
```

///2.no is positive or negative

```
#!/bin/sh
read a
if [ $a -gt 0 ]
 then
        echo "its positive"
 elif [ $a -lt 0 ]
 then
        echo "its negative"
 else
        echo "its 0"
  fi
```

```
///3.sum of 5 no
#!/bin/sh
for (( i=1; i<5; i++ ))

 do
        sum=$(( $sum + $i ))

 done
 echo "sum is"$sum



 ///4.odd or even no
 echo "enter the no"
read a
if [ `expr $a % 2` == 0 ]
then
       echo "the no is odd"
else
       echo "the no is even"
fi
/////////HR q.....


////5.trangale side equal or not if equal EQUILATERAL if not equal "SCALENE";
read x
read y
read z
if [ $x == $y ] && [ $y == $z ]
```

```
then
echo "EQUILATERAL";
elif [ $x != $y ] && [ $x != $z ] && [ $y != $z ]
then
echo "SCALENE";
else
echo "ISOSCELES";
fi
```

///6.print 1 to 100 odd no

```
#!/bin/sh
for (( i=1; i <=100; i++ ))
do
if [ `expr $i % 2` != 0 ]
then
echo  "$i"
fi
done
```

///7 print no 1 to 50

```
#!/bin/sh
for (( i=1; i<=50; i++ ))

 do
        echo $i

 done
```

```
///8.add sum mul div of 2 no
read x
read y
sum=` expr $x + $y `
echo $sum
difference=` expr $x - $y `
echo $difference
s=$(( $x * $y ))
echo $s
u=` expr $x / $y `
echo $u

////9.compa of two no
read x
read y
if [ $x > $y ]
then
echo "X is greater than Y"
elif [ $x < $y ]
then
echo "X is less than Y"
else
echo "X is equal to Y"
fi


/////10.Y-print yes y-print yes
read ch;
if [ $ch == 'Y' ] || [ $ch == 'y' ]
```

```
then
echo "YES";
else
echo "NO";
fi
```

```
//9 A mathematical expression containing +,-,*,^, / and parenthesis
//will be provided. Read in the expression, then evaluate it.

// Display the result rounded to  decimal places.

read a
printf "%.3f" $(echo "scale = 4; $a" | bc);
```

```
//10 Given  integers, compute their average, rounded to three decimal places.
read t
sum=0;
for((i=0;i<t;i++))
do
read num;
sum=$((sum+num))
done
printf "%.3f" $(echo "scale=4; $sum / $t " | bc )
```

```
////////comand Linux
//1.....
```

head -n 20 /////for display first 20 line

//2...

head -n 22 | tail -n 11 /////display the line from 12 to 22

//3 In this challenge, we practice using the tail

// command to display the last lines of a text file.

///Display the last lines of an input file.

tail -n 20

//4 In a given fragment of text, replace all sequences of multiple

// spaces with just one space.

tr -s ' '

///5 You are given a file of text, where each line contains a number

sort -n -r

//66 large size colum and row sort

sort -n -k2 -r -t $'\t'

//7 Given a text file, display only those lines which are not

//followed or preceded by identical replications.

uniq -u

//8 Given a list of countries, each on a new line, your task is to

// read them into an array and then display the entire array,

Solution

```
i=1;
while read line
do
   a[i]=$line;
   i=$((i+1));
done
echo "${a[@]}";
```

9//Objective

We now transition to some basic examples of bash scripting for the purpose of text processing and data munging.

In this challenge, we practice reading and filtering an array.

```
readarray array
declare -a output=( ${array[@]/*[a,A]*/} )

echo ${output[@]}
```

10// Given a list of countries, each on a new line, your task is to read them into a

//n array and then display the count of elements in that array.

```
arr=( $(cat) )

echo ${#arr[@]}
```

11//In this challenge, we practice reading and transforming arrays.

```
arr=( $(cat) )

for elem in ${arr[@]} ; do
   echo -ne ".${elem:1} "
done
```

13// Sed is a popular utility which enables quick parsing and transformation of text.

//Here are some very simple examples of sed in action.

//Substitute the first occurrence of 'editor' with 'tool'.

```
sed -E 's/([[:digit:]]{4}) ([[:digit:]]{4}) ([[:digit:]]{4}) ([[:digit:]]{4})/\4 \3 \2 \1/'
```

//14 There are integers in an array . All but one integer occur

// in pairs. Your task is to find the number that occurs only once.

```
read
arr=($(cat))
echo "${arr[@]}" | tr ' ' '\n' |sort | uniq -u | tr '\n' ' '
```

//15

///mam solution

OSC HAckerrak Challenege 2

---

1. Head of a Text File #1

```
head -20
```

---

2. Middle of a Text File

```
head -22 | tail -11
```

---

3. Tail of a Text File #1

```
tail -c 20
```

---

4. 'Tr' Command #3

tr -s ' '

---

5. Sort Command #4

sort -n -r

---

6. Sort Command #5

sort -k2 -n -r -t$'\t'

---

7. 'Uniq' command #4

uniq -u

---

8. Read in an Array

```
while read line
do
    arr=(${arr[@]} $line)
```

done

echo ${arr[@]}

---

9. Filter an Array with Patterns

arr=($(cat))

echo ${arr[@]/[aA]/}

---

10. Count the number of elements in an Array

arr=($(cat))

echo ${arr[@]/[aA]/}

---

11. Remove the First Capital Letter from Each Element

arr=($(cat))

echo ${arr[@]/[A-Z]/.}

---

12. 'Grep' - A

#We retain only those lines which have at least one of the following words:

the

that

then

```
# those
grep -iwe "the\|that\|then\|those"
```

---

13. 'Sed' command #5

```
sed -E 's/([0-9]{4}) ([0-9]{4}) ([0-9]{4}) ([0-9]{4})/\4 \3 \2 \1 /g'
```

---

14. Lonely Integer - Bash!

```
read
arr=($(cat))
echo "${arr[@]}" | tr ' ' '\n' |sort | uniq -u | tr '\n' ' '
```

---

15. Lonely Integer

```
        read
numArr=($(cat))

numArr=${numArr[*]}
echo $((${numArr// /^}))
```