Assignment No.04

Name:- Omprakash Khawshi

****

Q.1 Write a program in the following steps

a. Generates 10 Random 3 Digit number.

b. Store this random numbers into a array.

c. Then find the 2nd largest and the 2nd smallest element without sorting the array.

Code: -

```
#!/bin/bash
for((i=0 ;i<11;i++))
do
random=$((100 + RANDOM%20))
num[$i]=$random
done
echo "Array Elements :-" ${num[@]}
temp=0
for (( i=0; i<11; i++))
do
for ((j=i+1; j<11 ; j++))
do
if [ ${num[i]} -gt ${num[$((j))]} ]
then
temp=${num[i]}
num[$i]=${num[$((j))]}
num[$((j))]=$temp
fi
done
done
##echo "Array After Sorting :- { ${num[@]} }"
echo "Second Smallest :- ${num[1]}"
echo "Second Largest :- ${num[9]}"
```

MINGW64:/d/Assignments/Assignment No.4 Array/Q.1 Write a pr

```
  GNU nano 5.4
#!/bin/bash
for((i=0 ;i<11;i++))
do
random=$((100 + RANDOM%20))
num[$i]=$random
done
echo "Array Elements :-" ${num[@]}
temp=0
for (( i=0; i<11; i++))
do
for ((j=i+1; j<11 ; j++))
do
if [ ${num[i]} -gt ${num[$((j))]} ]
then
temp=${num[i]}
num[$i]=${num[$((j))]}
num[$((j))]=$temp
fi
done
done
##echo "Array After Sorting :- { ${num[@]} }"
echo "Second Smallest :- ${num[1]}"
echo "Second Largest :- ${num[9]}"
```

Output: -

```
Om@DESKTOP-D8GLB66 MINGW64 /d/Assignments/Assignment No.4 Array/Q.1 Wr
$ ./Q1.sh
Array Elements :- 109 113 102 110 109 104 115 111 112 110 110
Second Smallest :- 104
Second Largest :- 113
```

Q.2 Extend the above program to sort the array and then find the 2nd largest and the 2nd smallest element.

Code: -

```
#!/bin/bash
for((i=0 ;i<11;i++))
do
random=$((100 + RANDOM%20))
num[$i]=$random
done
echo "Array Before Sorting :- { ${num[@]} }"
temp=0
for (( i=0; i<11; i++))
do
for ((j=i+1; j<11 ; j++))
do
if [ ${num[i]} -gt ${num[$((j))]} ]
then
temp=${num[i]}
num[$i]=${num[$((j))]}
num[$((j))]=$temp
fi
done
done
echo "Array After Sorting :- { ${num[@]} }"
echo "Second Smallest :- ${num[1]}"
echo "Second Largest :- ${num[9]}"
```

MINGW64:/d/Assignments/Assignment No.4 Array/Q.2 Extend th

```
  GNU nano 5.4
#!/bin/bash
for((i=0 ;i<11;i++))
do
random=$((100 + RANDOM%20))
num[$i]=$random
done
echo "Array Before Sorting :- { ${num[@]} }"
temp=0
for (( i=0; i<11; i++))
do
for ((j=i+1; j<11 ; j++))
do
if [ ${num[i]} -gt ${num[$((j))]} ]
then
temp=${num[i]}
num[$i]=${num[$((j))]}
num[$((j))]=$temp
fi
done
done
echo "Array After Sorting :- { ${num[@]} }"
echo "Second Smallest :- ${num[1]}"
echo "Second Largest :- ${num[9]}"
```

Output: -

```
Om@DESKTOP-D8GLB66 MINGW64 /d/Assignments/Assignment No.4 Array/Q.2 Extend the above program to sor
ray and then find the 2nd largest and the 2nd smallest element
$ ./Q2.sh
Array Before Sorting :- { 116 116 109 113 105 110 102 117 115 109 119 }
Array After Sorting :- { 102 105 109 109 110 113 115 116 116 117 119 }
Second Smallest :- 105
Second Largest :- 117

Om@DESKTOP-D8GLB66 MINGW64 /d/Assignments/Assignment No.4 Array/Q.2 Extend the above program to sor
```

Q.3 Extend the Prime Factorization Program to store all the Prime Factors of a number n into an array and finally display the output.

Code: -

```
echo "enter an integer:"
read input
i=2
count=0
flag=0
for ((i;i<$input;));do
if [ `expr $input % $i` -eq 0 ];then
factor=$i
for ((j=2;j<=`expr $factor / 2`;));do
flag=0
if [ `expr $factor % $j` -eq 0 ];then
flag=1
break
fi
j=`expr $j + 1`
done
if [ $flag -eq 0 ];then
nos=$factor
array[$i]=$nos
count=1
fi
fi
i=`expr $i + 1`
done
if [ $count -eq 0 ];then
echo "no prime factors found except 1 and $input"
```

fi

echo ${array[@]}

MINGW64:/e/Assignments/Assignment No.4 Array/Q.3 Extend the Prime Factoriz

```
  GNU nano 5.4
echo "enter an integer:"
read input
i=2
count=0
flag=0
for ((i;i<$input;));do
      if [ `expr $input % $i` -eq 0 ];then
      factor=$i
for ((j=2;j<=`expr $factor / 2`;));do
      flag=0
      if [ `expr $factor % $j` -eq 0 ];then
      flag=1
      break
      fi
      j=`expr $j + 1`
  done
  if [ $flag -eq 0 ];then
   nos=$factor
   array[$i]=$nos
  count=1
  fi
  fi
  i=`expr $i + 1`
  done
  if [ $count -eq 0 ];then
  echo "no prime factors found except 1 and $input"
  fi
echo ${array[@]}
```

Output:-

MINGW64:/e/Assignments/Assignment No.4 Array/Q.3 Extend the Prime Factorization Progra

```
Om@DESKTOP-D8GLB66 MINGW64 /e/Assignments/Assignment No.4 Array/Q.3 Ext
to store all the Prime Factors of a number n into an array and finally
$ ./Q3.sh
enter an integer:
315
3 5 7

Om@DESKTOP-D8GLB66 MINGW64 /e/Assignments/Assignment No.4 Array/Q.3 Ext
$ |
```

Q.4 Write a Program to show Sum of three Integer adds to ZERO

Code: -

```
function tsfz()
{
echo " ☃☃☃ Array Elements which have Addition is Zero ☃☃☃"
for (( i=0 ; i<$(($n-2)) ; i++ ))
do
for (( j=$(($i+1)) ; j<$(($n-1)) ; j++ ))
do
for (( k=$(($i+2)) ; k<$n ; k++ ))
do
a=$(( ${arr[$i]} + ${arr[$j]} + ${arr[$k]} ))
if(($a==0))
then
echo "(${arr[$i]},${arr[$j]},${arr[$k]})"
found=1
fi
done
done
done
if(( $found==0 ))
then
echo "☹☹☹ No Elements find Addition is Zero ☹☹☹ "
fi
}
arr=(0 -1 2 -3 1 1 -2 1 0 5 )
n=${#arr[@]}
tsfz $arr $n
```

MINGW64:/d/Assignments/Assignment No.4 Array/Q.4 Write a Program to sho

```
  GNU nano 5.4
function tsfz()
{
echo " @@@ Array Elements which have Addition is Zero @@@"
for (( i=0 ; i<$(($n-2)) ; i++ ))
do
for (( j=$(($i+1)) ; j<$(($n-1)) ; j++ ))
do
for (( k=$(($i+2)) ; k<$n ; k++ ))
do
a=$(( ${arr[$i]} + ${arr[$j]} + ${arr[$k]} ))
if(($a==0))
then
echo "(${arr[$i]},${arr[$j]},${arr[$k]})"
found=1
fi
done
done
done
if(( $found==0 ))
then
echo "&&& No Elements find Addition is Zero &&& "
fi
}
arr=(0 -1 2 -3 1 1 -2 1 0 5 )
n=${#arr[@]}
tsfz $arr $n
```

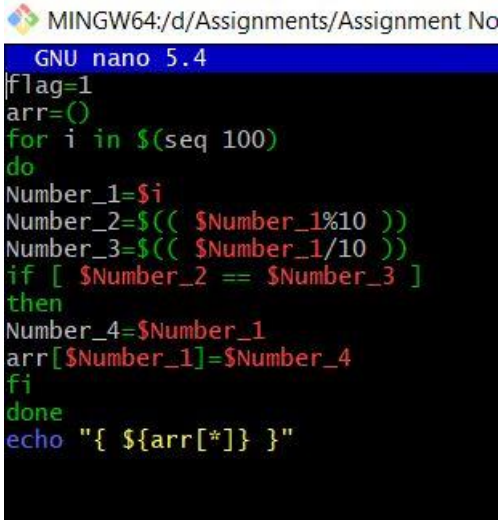
Output: -

```
Om@DESKTOP-D8GLB66 MINGW64 /d/Assignments/Assignment No.4 Array/Q.4 Write a
teger adds to ZERO
$ nano Q4.sh

Om@DESKTOP-D8GLB66 MINGW64 /d/Assignments/Assignment No.4 Array/Q.4 Write a
teger adds to ZERO
$ ./Q4.sh
 @@@ Array Elements which have Addition is Zero @@@
(0,-1,1)
(0,-1,1)
(0,-1,1)
(0,2,-2)
(0,-2,2)
(0,0,0)
(-1,1,0)
(-1,1,0)
(-1,1,0)
(-1,0,1)
(-1,0,1)
(-1,0,1)
(2,-3,1)
(2,-3,1)
(2,-3,1)
(2,-2,0)
(2,0,-2)
(-3,-2,5)
(1,1,-2)
(1,-2,1)
(1,1,-2)
(1,-2,1)

Om@DESKTOP-D8GLB66 MINGW64 /d/Assignments/Assignment No.4 Array/Q.4 Write a
teger adds to ZERO
```

Q.5 Take a range from 0 – 100, find the digits that are repeated twice like 33, 77, etc and store them in an array

Code: -

```
flag=1
arr=()
for i in $(seq 100)
do
Number_1=$i
Number_2=$(( $Number_1%10 ))
Number_3=$(( $Number_1/10 ))
if [ $Number_2 == $Number_3 ]
then
Number_4=$Number_1
arr[$Number_1]=$Number_4
fi
done
echo "{ ${arr[*]} }"
```



Output: -