Q1 Write a script that will read a set of positional variables (max 9) and display the resultant expression and the sum using the shift command

```
#!/bin/bash
npospar=$#   #pick up the number of pos parameters typed
i=0
((len = $npospar - 1 ))
total=0
while test $i -le $len   # [ $i -le $len ]
do
 #echo "$i"
 (( total = $total + $1 ))
 if test $i -eq $len
 then
   echo -e "$1 \c"
 else
   echo -e "$1 + \c"
 fi
 shift
((i = $i + 1 ))
done
echo "= $total"
```

Q2 **Write a script that will read a set of positional variables (max 9) and display the resultant expression and the sum using an array**

```
#!/bin/bash
echo "The pos paratmeters are : $@" #print all the positional parameters
darray=($@) #generating an array asigining all the pos parameters
npar=$#  #picking up the number of pos parameters
((nitem = $npar - 1)) # decrease by 1 start our count from 0
i=0   #initilise index to 0
total=0
while test $i -le $nitem  #loop from 0 to nitem
do
 if test $i -eq  $nitem
 then
    echo -e "${darray[$i]}  \c"
 else
    echo -e "${darray[$i]} + \c"   #print out index and array element
 fi
 ((total = total + ${darray[$i]} ))
 (( i = $i + 1 ))  # increment i by
done
echo " = $total"
```

Q3 Write a script that will have a single parameter a filename. The script will then display if the filename is a regular file; or if its a directory.

```
#!/bin/bash
if test $# -ne 1
then
  echo "Should provide a single parameter "
  exit 1
fi
if test -f $1
then
   echo "$1 is a regular file"
elif test -d $1
then
   echo "$1 is a directory "
fi
exit 0
```

Q4 Write a script that will read n positional parameters and then display them in order(smallest first), using the bubble sort algorithm

```
#!/bin/bash
Nnumb=$#  #pick up number of positional parameters
data=($@) #store all positional prarmeters into an array starting from $1 ..$n
(( Dlimt = $Nnumb - 1 ))
i=0
#echo "limt = $Dlimt i = $i"
while test  $i -lt $Dlimt
do
  j=0
  while test $j -lt $Dlimt
  do
    ((k =  $j + 1)) # k index for adjacent element (j + 1)
    #echo "$j ${data[$j]} $k ${data[$k]}"
    if  [ ${data[$j]} -gt  ${data[$k]} ]
    then
        #swap elements as element j larger than element k
        (( temp = ${data[$j]} ))
        (( data[$j] = data[$k] ))
        (( data[$k] = $temp ))
    fi
    ((j = $j + 1 )) # incrementing j
  done
   ((i = $i + 1)) #increementing counter
done  # outer loop
echo "ordered elements ${data[*]}" #print out all elements in array
exit 0
```

Q5 **Write a script that will have two parameters two files; your script will then sort the contents of the 1st file in alphanumeric order and store this into a second file. Note that it's to be assumed that the 2nd file does not exist**

```
#!/bin/bash
if test $# -ne 2
then
  echo "Need two positional paramters filename and new filename"
  exit 1
fi
if test -f $1
then
  sort $1 > $2
  echo "$2:"
  cat $2
  exit 0
else
 echo "$1 needs to be a file "
 exit 2
fi
```

Q6 **Write a script that will have a single parameter a filename and then displays a menu of different editors e.g. nano and vi, and prompts the user for their choice. Once a choice is made it will open the file with the editor of their choice, using the case statement**

```
#!/bin/bash
if test $# -ne 1
then
  echo "Need a single poistional parameter only"
fi

echo "*****************************"
echo "1 ... nano"
echo "2 ... vi"
echo "*****************************"
read -p ": " option
case $option in
 1)
  nano $1
  ;;
 2)
  vi $1
  ;;
 *)
  echo "Wrong Option"
  ;;
 esac
exit 0
```

Q7 **Write a script that will have a single parameter a filename. The script prompts the user for a string and it appends it at the end of the file**

```
#!/bin/bash
if test $# -ne 1
then
  echo "You need 1 parameter a filename"
  exit 1
fi
read -p "Type string to be appeneded to file $1 : " strinput
echo $strinput >> $1
exit 0
```