Simple Shell Scripts

1. Write a shell script to accept two file names and check if both exists. If the second filename exists, then the contents of the first filename should be appended to it. If the second file name does not exist then create a new file with the contents of the first file.

```
echo "Enter First file name" read file1
echo " Enter Second file name" read file2
if [ -e $file1 ] then
   cat $file1 >> $file2
else
   echo " $file1 does not exist"
fi
echo " $file1 content:" cat $file1
echo " $file2 content:" cat $file2
```

2. Write a shell script that accepts a file name, starting and ending line numbers as arguments and displays all the lines between the given line numbers.

```
echo " Enter the file name :"
read file
if [ -f $file ] then
   echo "Enter the Starting line number:"
   read snum
   echo "Enter the Ending line number:"
   read enum
   if [ $snum -lt $enum ]
   then
       echo "The selected lines from $snum line to $enum line in $file :"
       sed -n ' '$snum','$enum"p" $file
  else
      echo "Enter proper starting & ending line numbers."
  fi
else
   echo "The file ' $file ' doesn't exists. "
fi
```

3. Write an interactive file-handling shell program. Let it offer the user the choice of copying, removing, renaming, or linking files. Once the user has made a choice, have the program ask the user for the necessary information, such as the file name and so on.

```
echo "1.COPY"
echo "2.RENAME"
echo "3.REMOVE"
echo "4.LINK"
echo "5.EXIT"
echo "Enter your choice"
read ch
```

```
case $ch in
1) echo "Enter the sources file"
    read s
    echo "Enter the destination file " read d
    cp $s $d
    ;;
2) echo "Enter the old file name"
    read of
    echo "enter the new file name"
    read nf
    mv $of $nf
    ;;
3) echo "Enter file name to delete"
    read $df
    rm $df
    ;;
4) echo "Enter the file1"
    read f1
    echo "enter the file2"
    read f2
    ln $f1 $f2
    ;;
5) exit
    ;;
esac
```

4. Write a shell script that accepts one or more file name as arguments and converts all of them to uppercase, provided they exist in the current directory.

```
for i in $*
do
   if [ ! -f $i ]
   then
    echo "Filename $fileName does not exists"
    exit
  fi
  # convert uppercase to lowercase using tr command
  tr '[A-Z]' '[a-z]' < $i
done
```

5. Write a shell script that takes a valid directory name as an argument and recursively descend all the subdirectories, finds the largest file in the directory

```
clear
if [ $# -ne 1 ]
then
     echo -e “\n\nInvalid Number of arguments passed\n”
```

```
    exit
fi
cd $1
echo The directory name is  $1
set -- ` ls -lR| grep -v "^d" |sort +4 -5 -rn`
echo "size of the largest file is $5 blocks"
```

6. Write a shell script that accepts two file names as arguments, checks if the permissions for these files are identical and if the permissions are identical, output common permissions and otherwise output each file name followed by its permissions.

```
if [ $# -ne 2 ] then
       echo "pass 2 argument"
       exit
fi
p1=`ls -l $1 | cut -c 2-10`
p2=`ls -l $2 | cut -c 2-10`
if [ $p1 = $p2  ]
then
      echo permissions are same
      echo $p1
else
      echo permissions are different
      echo permission of file $1 is $p1
      echo permission of file $2 is $p2
fi
```

7. Write a non-recursive shell script which accepts any number of arguments and prints them in the reverse order (for example, if the script is named rags, then executing rags ABC should produce CBA on the standard output)

```
a=$#
echo "Number of arguments are" $a
x=$*
c=$a
res=''
while [ 1 -le $c ]
do
     c=`expr $c - 1`
    shift $c
    res=$res' '$1
    set $x
done
echo Arguments in reverse order  $res
```