Dikshitha Vani V
205001032

Exercise 1

1.If the cost price and selling price of an item is input through the keyboard, write a program to determine whether the seller has made profit or incurred loss. Also determine how much profit was made or loss incurred.

```
read -p "Enter cost price: " cost
read -p "Enter selling price: " sell
if (( sell>cost ))
then
profit=$(($sell-$cost))
echo "Profit of $profit"
elif (( sell<cost ))
then loss=$(($cost-$sell))
echo "Loss of $loss"
else
echo "No profit/loss"
fi
```

```
~/ex8$ bash ex8_1.sh
Enter cost price: 1000
Enter selling price: 200
Loss of 800
```

2.Write a shell script to validate password strength. Here are a few assumptions for the password string.

- •Length – minimum of 8 characters.
- •Contains both alphabet and numbers.
- •Include both the small and capital case letters.

if the password doesn't satisfy with any of the above conditions, then the script should print it as a "Weak Password"

```
read -p "Enter password: " pass
if [[ ${#pass} -ge 8 ]]
then
  echo "$pass" | grep -q "[A-Z]"
  if [[ $? -eq 0 ]]
  then
    echo "$pass" | grep -q "[a-z]"
  if [[ $? -eq 0 ]]
  then
    echo "$pass" | grep -q "[0-9]"
    if [[ $? -eq 0 ]]
    then
       echo "Strong password"
    else
       echo "Must contain numbers"
    fi
 else
    echo "Must contain lowercase characters"
 fi
 else
   echo "Must contain uppercase characters"
fi
else
 echo "Must be 8 characters"
fi
```

```
Enter password: ABCDEDFj
Must contain numbers

~/ex8$ bash ex8_1.sh
Enter password: 1234Abcd
Strong password
```

3.Write a script that prints essentially the same information as ls -l a but in a more user- friendly way. (a) file exists or not (b) regular file?
(c) directory?
(d) readable?
(e) writable?
(f) executable?

(g) owner
Print suitable messages.Rewrite the above script as a shell function finfo and call the function with a filename.

```
read -p "Enter file name:" filename
if [ -e $filename ]; then
 echo "File exists"
 if [ -f $filename ]; then
       echo "regular file"
 else
       echo "special file"
 fi

 if [ -r $filename ]; then
       echo "readable"
 else
       echo "Not readable"
 fi

 if [ -w $filename ]; then
       echo "writeable"
 else
       echo "Not writeable"
 fi

 if [ -x $filename ]; then
       echo "executable"
 else
      echo "Not executable"
 fi
else
 echo "file does not exist"
 fi
```

```
finfo(){
if [ -e $filename ]; then
echo "File exists"
if [ -f $filename ]; then
      echo "regular file"
else
      echo "special file"
fi
if [ -r $filename ]; then
      echo "readable"
else
      echo "Not readable"
fi
if [ -w $filename ]; then
      echo "writeable"
else
      echo "Not writeable"
fi
if [ -x $filename ]; then
      echo "executable"
else
     echo "Not executable"
fi
else
echo "file does not exist"
fi
}
read -p "Enter file name:" filename
finfo $filename
```

```
~/ex8$ bash ex8_1.sh
Enter file name: nonExisting.txt
Path does not exist

~/ex8$ bash ex8_1.sh
Enter file name: ex8_1.sh
runner
Path exists
Regular File
Readable file
```

```
Enter file name:db
file does not exist

~/ex8$ bash ex8_1.sh
Enter file name:ex8_1.sh
File exists
regular file
readable
writeable
Not executable
```

Excerise 2 (loops)

1. Write a program to generate all combinations of digits 1, 2 and 3 to form different numbers using for loops.

```
for i in {1..3}
do
for j in {1..3}
do
for k in {1..3}
do echo "$i $j $k"
done
done
done
```

```
~/ex8$ bash ex8_1.sh
1 1 1
1 1 2
1 1 3
1 2 1
1 2 2
1 2 3
1 3 1
1 3 2
1 3 3
2 1 1
2 1 2
2 1 3
2 2 1
```

2 2 2
2 2 3
2 3 1
2 3 2
2 3 3
3 1 1
3 1 2
3 1 3
3 2 1
3 2 2
3 2 3
3 3 1
3 3 2
3 3 3

2. Use seq with for statement to print the multiplication table.

```
read -p "Enter number: " num
end=$(($num*10))
echo "Multiplication Table"
seq $num $num $end
```

```
~/ex8$ bash ex8_1.sh
Enter number: 5
Multiplication Table
5
10
15
20
25
30
35
40
45
50
```

3.Write a shell script to check whether a given string is a palindrome or not

```
read -p "Enter string to check for palindrome: " data
```

```
revdata=`echo "$data"|rev`
if [ "$data" == "$revdata" ]
then
echo "$data is a palindrome"
else
echo "$data is not a palindrome"
fi
```

```
~/ex8$ bash ex8_1.sh
Enter string to check for palindrome: madam
madam is a palindrome
```

```
~/ex8$ bash ex8_1.sh
Enter string to check for palindrome: sad
sad is not a palindrome
```

4. Write a shell script to compute ‘m’ to the power of a positive integer ‘n’, i.e. m^n (while loop)

```
read -p "Enter m: " m
read -p "Enter n: " n
result=1
while (( n>0 ))
do result=$((result*m))
(( n-- ))
done
echo "Result: $result"
```

```
~/ex8$ bash ex8_1.sh
Enter m: 5
Enter n: 2
Result: 25
```

5. Write a script that attempts to copy a file to a directory and, if it fails, waits 5 seconds and then tries again continuing until it succeeds. (Use Until statement).

```
read -p "Enter file to copy: " file
read -p "Enter directory to copy to: " directory
until cp $file $directory/
do
echo "Trying to copy file to directory"
```

```
sleep 5
read -p "Enter file to copy: " file
read -p "Enter directory to copy to: " directory
done
```

```
~/ex8$ bash ex8_1.sh
Enter file to copy: ex8_1.sh
Enter directory to copy to: newdir
cp: cannot create regular file 'newdir/': Not a directory
Trying to copy file to directory
Enter file to copy: ex8_1.sh
Enter directory to copy to: new_directory
```

6. Write a menu based program to copy a given file, to remove the specified file and to move a file.

```
echo "Menu"
echo "1. Copy a File"
echo "2. Remove a file "
echo "3. Move a file"
echo "4. Quit"
read -p "Enter your choice: " choice
case "$choice" in
          1) read -p "Enter file name to copy: " f1
             read -p "Enter file to copy to: " f2
             if [ -f $f1 ]
             then
                 cp $f1 $f2
                 echo "Original file: "
                 cat $f1
                 echo "Copied file: "
                 cat $f2
             else
                    echo "$f1 does not exist"
             fi
             ;;
         2) read -p "Enter file to be removed: " r
            if [ -f $r ]
            then
```

```
               rm -i $r
         else
              echo "file $r does not exist"
         fi
         ;;
     3) read -p "Enter source file: " f1
        read -p "Enter destination directory: " f2
        if [ -f $f1 ]
        then
             if [ -d $f2 ]
             then
                  mv $f1 $f2
             else
                 echo "$f2 is not a directory"
             fi
         else
             echo "$f1 does not exist"
        fi
        ;;
     4)  echo "Exiting the program"
         exit;;
    esac
```

```
~/ex8$ bash ex8_1.sh
Menu
1. Copy a File
2. Remove a file
3. Move a file
4. Quit
Enter your choice: 1
Enter file name to copy: text.txt
Enter file to copy to: new_text.txt
Original file:
Some content to fill the space
next line
now the next lineCopied file:
Some content to fill the space
next line
now the next line

~/ex8$ bash ex8_1.sh
```

```
Menu
1. Copy a File
2. Remove a file
3. Move a file
4. Quit
Enter your choice: 2
Enter file to be removed: new_text.txt
rm: remove regular file 'new_text.txt'? y
```

```
~/ex8$ bash ex8_1.sh
Menu
1. Copy a File
2. Remove a file
3. Move a file
4. Quit
Enter your choice: 3
Enter source file: text.txt
Enter destination directory: new_directory
```

Excerise 3 (function)

1.Write shell script to read a text file name and count the number of lines using function. Pass the file name as an argument to the function. Return the number of lines and print it

```
lines(){
   if [ -f "$1" ]
   then
       num=`wc -l $1 | cut -d" " -f1`
       echo "File $1 has $num lines"
   else
       echo "$1 is not a file"
   fi
}

lines $1
```

```
~/ex8$ bash ex8_1.sh random.txt
File random.txt has 3 lines
```

Contents of random.txt:

Some content to fill the space

next line
now the next line

2. Write a shell script to count the number of occurrences of given word in the file. (Note: File name and word to be passed as an argument to the script).

```
occurence(){
   file=$1
   word=$2
   if [ -f "$file" ]
   then
       num=`grep -c "$word" "$file" | cut -d" " -f1`
       echo "There are $num occurences of the word $word in $file"
   else
       echo "$1 is not a file"
   fi
}

occurence $1 $2
```

Contents of random.txt
Some content to fill the space
next line
now the next line
i am repeating the word
word wantedly

```
~/ex8$ bash ex8_1.sh random.txt line
There are 2 occurences of the word line in random.txt
```

3. Anna University converts the marks in an exam to letter grades according to the following table. Write a shell script to translate the marks of a student in a semester into letter grades.

```
fn (){
for i
do
case $i in
9[1-9]|100)l=S g=10
;;
8[1-9]|90)l=A g=9
;;
7[1-9]|80)l=B g=8
;;
6[1-9]|70)l=C g=7
;;
5[7-9]|60)l=D g=6
;;
5[1-6])l=E g=5
;;
*)l=U g=0
;;
esac
echo -e "Grade: $g Lettergrade: $l"
done
}
fn $@
```

```
~/ex8$ bash ex8_1.sh
~/ex8$ bash ex8_1.sh 99 87 45
Grade: 10 Lettergrade: S
Grade: 9 Lettergrade: A
Grade: 0 Lettergrade: U
```

Exercise 4

1. Write a shell script that prints 5 command line arguments. What happens if we pass fewer than 5 arguments?

```
j=1
for i in "$@"
do
echo $i
j=$(( $j+1 ))
if [[ $j > 5 ]]; then
break
fi
```

```
done
```

```
~/ex8$ bash ex8_1.sh 1 2 6 7
1
2
6
7
```

"$@" is used to print the command line arguments, if less number of arguments are provided nothing is printed at that place

2. Change the value of a positional parameter. Did you succeed

```
echo the parameters are:
for i in "$@"
do
echo $i
done
echo changing argument 2
$1="goingToChange"
echo the parameters after change are:
for i in "$@"
do
echo $i
done
```

```
~/ex8$ bash ex8_1.sh 1 2 3 4 5
the parameters are:
1
2
3
4
5
changing argument 1
ex8_1.sh: line 7: 1=goingToChange: command not found
the parameters after change are:
1
2
3
4
5
```

“$@” is used to print the command line arguments, changing one of the parameters is not possible and thus error is shown

Exercise 5

1. Develop an interactive script to maintain a database of employees.
The database is in the format
The script should allow users to 1. List the records 2. Search for an employee 3. Modify the hours_worked of an employee whose existing hours_worked is equal to 0. 4. Delete an employee 5. Quit

```
while [ 1 ]
do
echo -e "1)list 2)search 3)modify 4)delete 5)quit"
echo enter choice
read i
case $i in
1)echo -e "list record:"
awk '{print $0}' f2.txt
;;
2)echo -e "search record:"
echo enter record to search:
read name
sed -n "/$name/p" f2.txt
;;
3)echo -e "modify record:"
read -p "Enter modified hours:" hours
sed -ie "s/ 0$/${hours}$/" f2.txt
;;
4)echo -e "delete record"
echo enter record to delete:
read name
sed "/$name/d" f2.txt
;;
5)break
;;
*)echo wrong choice $i
esac
done
```

```
~/ex8$ bash ex8_1.sh
1)list 2)search 3)modify 4)delete 5)quit
enter choice
1
list record:
Beth 4.00 0
Dan 3.75 0
Kathy 4.00 10
Mark 5.00 20
Mary 5.50 22
Susie 4.25 18
1)list 2)search 3)modify 4)delete 5)quit
enter choice
2
search record:
enter record to search: name
Mark
Mark 5.00 20
1)list 2)search 3)modify 4)delete 5)quit
enter choice
3
modify record:
Enter modified hours:10
1)list 2)search 3)modify 4)delete 5)quit
enter choice
1
list record:
Beth 4.00 10$
Dan 3.75 10$
Kathy 4.00 10
Mark 5.00 20
Mary 5.50 22
Susie 4.25 18
1)list 2)search 3)modify 4)delete 5)quit
enter choice
^C
~/ex8$ bash ex8_1.sh
1)list 2)search 3)modify 4)delete 5)quit
enter choice
1
list record:
Beth 4.00 0
```

```
Dan 3.75 0
Kathy 4.00 10
Mark 5.00 20
Mary 5.50 22
Susie 4.25 18
1)list 2)search 3)modify 4)delete 5)quit
enter choice
2
search record:
enter record to search:
Kathy
Kathy 4.00 10
1)list 2)search 3)modify 4)delete 5)quit
enter choice
3
modify record:
Enter modified hours:10
1)list 2)search 3)modify 4)delete 5)quit
enter choice
1
list record:
Beth 4.00 10
Dan 3.75 10
Kathy 4.00 10
Mark 5.00 20
Mary 5.50 22
Susie 4.25 18
1)list 2)search 3)modify 4)delete 5)quit
enter choice
4
delete record
enter record to delete:
susie
Beth 4.00 10
Dan 3.75 10
Kathy 4.00 10
Mark 5.00 20
Mary 5.50 22
Susie 4.25 181)list 2)search 3)modify 4)delete 5)quit
enter choice
4
delete record
enter record to delete:
Susie
```

Beth 4.00 10
Dan 3.75 10
Kathy 4.00 10
Mark 5.00 20
Mary 5.50 22
1)list 2)search 3)modify 4)delete 5)quit
enter choice
5

2. Create an array by assignment of prices for five different fruits with fruit name as key and price as value. a. Display the all the key. b. Display the values. c. Display the key value pair. d. Remove the third fruit. e. Add one new fruit. f. Calculate the total cost of all fruits and display the amount. g. Delete the all items and display

```
declare -A fruits
for i in {1..5}
do
read -p "Enter fruit name: " name
read -p "Enter price: " price
fruits[$name]=$price
done
echo "Keys: "
for key in "${!fruits[@]}"; do echo $key; done
echo
echo "Values: "
for val in "${fruits[@]}"; do echo $val; done
echo
echo "Key and Values: "
for key in "${!fruits[@]}"
do
echo "Key: $key Value: ${fruits[$key]}"
done
echo
echo "Deleting the third element: "
count=0
for key in "${!fruits[@]}"
do
(( count++ ))
if (( count == 3 ))
then
```

```
unset fruits[$key]
fi
done
echo
echo "Key and Values: "
for key in "${!fruits[@]}"
do
echo "Key: $key Value: ${fruits[$key]}"
done
echo
echo "Adding a new element: "
read -p "Enter fruit name: " name
read -p "Enter price: " price
fruits[$name]=$price
echo
echo "Key and Values: "
for key in "${!fruits[@]}"
do
echo "Key: $key Value: ${fruits[$key]}"
done
echo
echo "Total cost: "
total=0
for value in "${fruits[@]}"
do
total=$(( $total+$value ))
done
echo "Total price: " $total
echo
echo "Deleting associative array: "
unset fruits
echo "${fruits[@]}"
```

```
~/ex8$ bash ex8_1.sh
Enter fruit name: Mango
Enter price: 100
Enter fruit name: watermelon
Enter price: 120
Enter fruit name: grapes
Enter price: 60
Enter fruit name: apple
Enter price: 50
Enter fruit name: orange
Enter price: 30
Keys:
apple
orange
Mango
grapes
watermelon

Values:
50
30
100
60
120

Key and Values:
Key: apple Value: 50
Key: orange Value: 30
Key: Mango Value: 100
Key: grapes Value: 60
Key: watermelon Value: 120

Deleting the third element:

Key and Values:
Key: apple Value: 50
Key: orange Value: 30
Key: grapes Value: 60
Key: watermelon Value: 120

Adding a new element:
Enter fruit name: banana
Enter price: 20

Key and Values:
Key: apple Value: 50
Key: banana Value: 20
Key: orange Value: 30
Key: grapes Value: 60
Key: watermelon Value: 120
```

Total cost:
Total price: 280

Deleting associative array:
(no o/p)

using unset we delete the array and after that printing array gives no output

Exercise 6.
 1. Write a function that allows the user to select a directory from the list of directories. Move the selected directory to the first position of the list. (Using select statement).

```
declare -a list
list=`ls`
echo $list
select dir in $list
do
echo "Selected: $dir"
echo $dir
for i in $list
do
if [ $dir != $i ]
then
echo $i
fi
done
done
```

```
~/ex8$ bash ex8_1.sh
a b dfhhb.txt ex8_1.sh f2.txt f2.txte main.sh new_directory new_text.tx random.txt
1) a            5) f2.txt       9) new_text.txt
2) b            6) f2.txte     10) random.txt
3) dfhhb.txt        7) main.sh
4) ex8_1.sh     8) new_directory
#? 1
Selected: a
a
b
dfhhb.txt
ex8_1.sh
f2.txt
```

```
f2.txte
main.sh
new_directory
new_text.txt
random.txt
#? 2
Selected: b
b
a
dfhhb.txt
ex8_1.sh
f2.txt
f2.txte
main.sh
new_directory
new_text.txt
random.txt
#? 3
Selected: dfhhb.txt
dfhhb.txt
a
b
ex8_1.sh
f2.txt
f2.txte
main.sh
new_directory
new_text.txt
random.txt
```

2. Write a shell script to translate the contents of a file into Upper case, Lower case, title case and print not valid case when invalid argument passed where file name is entered through command line.(use select case)

```
if [ -e $1 ]
then
select choice in UpperCase LowerCase TitleCase
do
case $choice in
UpperCase) echo "File in Upper case: "
cat $1|tr "[a-z]" "[A-Z]"
;;
```

```
LowerCase) echo "File in Lower case: "
cat $1|tr "[A-Z]" "[a-z]"
;;
TitleCase) echo "Title case: "c
sed "s/\<./\U&/g" $1
;;
*) echo "Wrong choice"
esac
done
else
echo "Invalid file"
fi
```

```
~/ex8$ bash ex8_1.sh
1) UpperCase
2) LowerCase
3) TitleCase
#? 1
File in Upper case:
file in upper case
FILE IN UPPER CASE
#? 2
File in Lower case:
file in lowercASE
file in lowercase
#? 3
Title case: c
title in progress
Title In Progress
#? ^C
```