Firas Aboushamalah
250-920-750
CS2211B
02/13/2018
Assignment 2

Question 1 (10 Marks):

a)

1. just the login to the shell at the top of the script. It doesn’t output anything.
2. A comment following the login command.
4. assigning the variable “x” a command to the first argument which isn’t given, so it’s just empty
5. you’re echoing the 0 argument which invokes bash as login shell
6. Echoing argument 1
7. echoing the number of arguments in the script which is 0
8. Echo all the arguments in a single string (which there are none)
9. printing the contents of “x” to echo, which are none.
11. shifts all arguments to the left by one.
13. again, invoking bash server
14. echo the first positional parameter (argument)
15. echo number of arguments in the script (still 0)
16. echo all the arguments in one line
17. echo the contents of variable x, still outputting blank because no arguments are given to it in $1.

b)

```
[fabousha@cs2211b ~]$ test1.sh a b c
./test1.sh
a
3
a b c
a
./test1.sh
b
2
b c
a
```

Question 2: (10 Marks)

a)

1. just the login to the bash at the top of the script to make sure it runs. It doesn’t output anything.
3. Taking standard input from user to run the echo commands
4. reads the input to place it inside the variable x.
6. does not process the arithmetic, it just reads the standard input and processes the literal statement “expr __ + 7” word for word.
7. Same as line 6.
8. It does the same as line 6, except it reads the $x as the literal word and does it process it as a variable.
9. the back quote makes shell execute command in the quotes and returns the result.

b)

```
[fabousha@cs2211b ~]$ test2.sh
Input Number: 10
expr 10 + 7
expr 10 + 7
expr $x + 7
17
```

Question 3: (15 Marks)

a)

1. The script reads the current hour from the “date” command, and if it is before 12PM noon, then echo “Good Morning”, or if it is before 6PM (18:00), then echo “Good Afternoon”, else print “Good Evening” followed by a “fi” to end the script. The echo’s at the beginning and end are just empty lines.

b)

The good2.sh script is attached as a plain text file.

Question 4: (18 Marks)

```
#log in bash server
#!/bin/bash

#Firas Aboushamalah
#250920750
#Assignment 2 Part 4: This script takes standard input for a directory
and creates the backup of it by calling scp.

#taking standard input for directory name
echo -n 'Input a directory name: "
read direc #reading the input and putting it in a variable "direc"


#if the directory exists
if [ -d $direc ]; then

        #If the directory backup exists
        if [ -d 'backup' ]; then
                #remove it
                rm -r backup
        fi #end if statement

        #copy the directory to current direc and name it "backup"
        cp -r $direc backup
        #making an archive of backup directory named "backup.tar"
        tar -cf "backup.tar" $direc
        #renaming backup.tar file so its in correct format
        my backup.tar backup`date +-%d-%m-%Y`.tar
        #calling scp to send copy of tar file to home directory as
"mybackups"
        scp -r fabousha@cs2211b.gaul.csd.uwo.ca:~/backup`date +-%d-%m-
%Y`.tar $USER$cs2211b.gaul.csd.uwo.ca:~/mybackups

#if directory doesn't exist
else
     #print error saying it doesn't exist and give exit 1 error
        echo "ERROR: Directory does not exist"
        exit 1

#ending first if statement
fi
```

Question 5: (20 Marks)

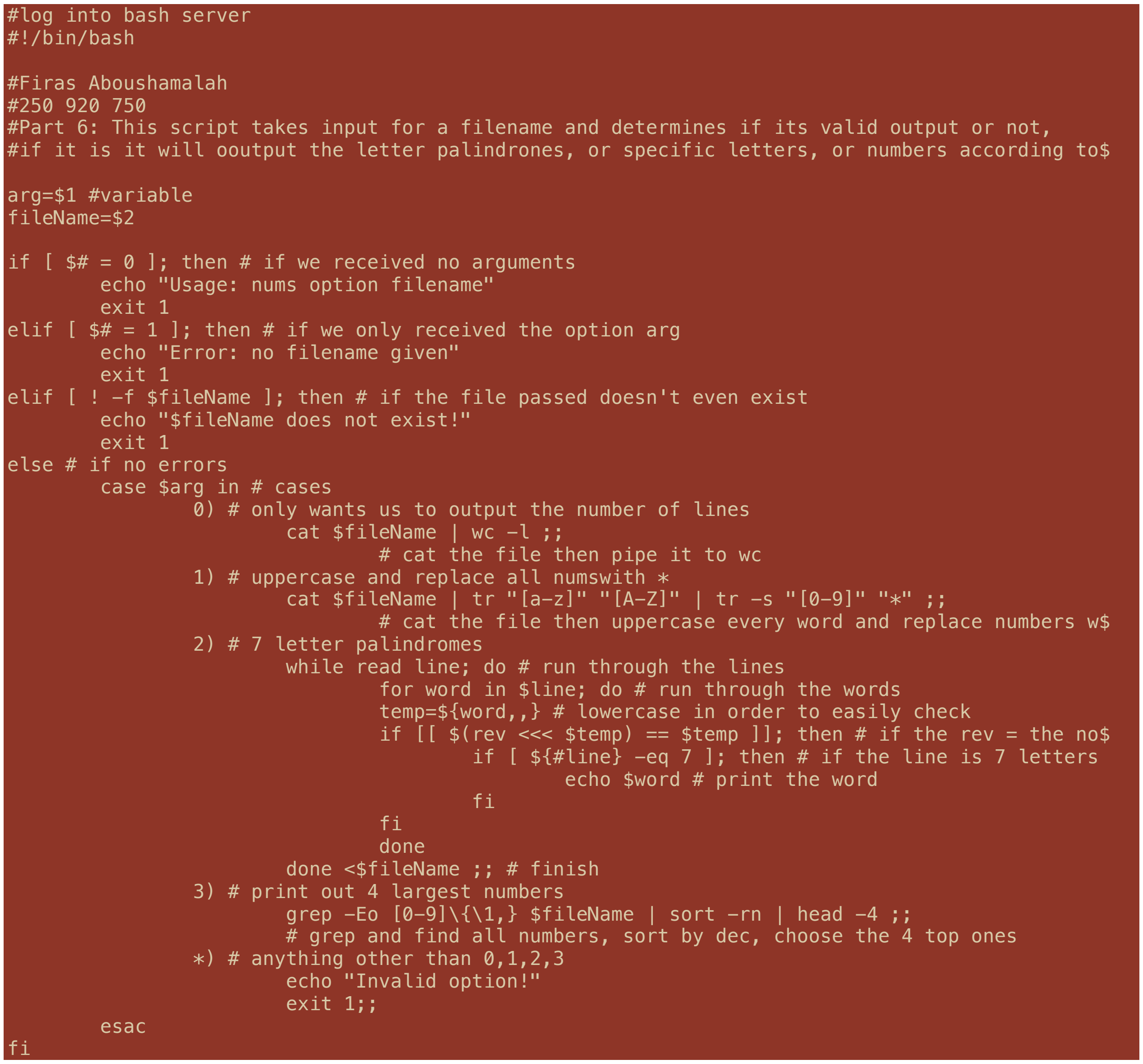
SCRIPT:

```
#!/bin/bash
# Firas Aboushamalah
# 250 920 750
# This script takes any number of arguments as valid integers by command line and outputs
thei$
echo

total=0
avg=0
number_of_args=$#

if [ $# -gt 0 ]; then

        for var in $*
        do
              total=`expr $total + $var`
        done

       echo "scale=2; $total / $number_of_args" | bc
else
       echo "Need at least one argument!"
        exit 1

fi
echo
```

OUTPUT:

```
[fabousha@cs2211b ~]$ avg.sh

Need at least one argument!

[fabousha@cs2211b ~]$ avg.sh 42

42.00

[fabousha@cs2211b ~]$ avg.sh -50 0 50 100

25.00

[fabousha@cs2211b ~]$ avg.sh 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
18 19 20

10.50
```

Question 6: (25 Marks)



```
#log into bash server
#!/bin/bash

#Firas Aboushamalah
#250 920 750
#Part 6: This script takes input for a filename and determines if its valid output or not,
#if it is it will ooutput the letter palindrones, or specific letters, or numbers according to$

arg=$1 #variable
fileName=$2

if [ $# = 0 ]; then # if we received no arguments
        echo "Usage: nums option filename"
        exit 1
elif [ $# = 1 ]; then # if we only received the option arg
        echo "Error: no filename given"
        exit 1
elif [ ! -f $fileName ]; then # if the file passed doesn't even exist
        echo "$fileName does not exist!"
        exit 1
else # if no errors
        case $arg in # cases
                0) # only wants us to output the number of lines
                        cat $fileName | wc -l ;;
                                # cat the file then pipe it to wc
                1) # uppercase and replace all numswith *
                        cat $fileName | tr "[a-z]" "[A-Z]" | tr -s "[0-9]" "*" ;;
                                # cat the file then uppercase every word and replace numbers w$
                2) # 7 letter palindromes
                        while read line; do # run through the lines
                                for word in $line; do # run through the words
                                temp=${word,,} # lowercase in order to easily check
                                if [[ $(rev <<< $temp) == $temp ]]; then # if the rev = the no$
                                        if [ ${#line} -eq 7 ]; then # if the line is 7 letters
                                                echo $word # print the word
                                        fi
                                fi
                                done
                        done <$fileName ;; # finish
                3) # print out 4 largest numbers
                        grep -Eo [0-9]\{\1,} $fileName | sort -rn | head -4 ;;
                        # grep and find all numbers, sort by dec, choose the 4 top ones
                *) # anything other than 0,1,2,3
                        echo "Invalid option!"
                        exit 1;;
        esac
fi
```