Advanced Bash Shell Scripting – Solutions

Lab 1: Putting Case Statements into Practice

```
#!/bin/bash
# caseExample will accept a number between 1 and 12 as
# an argument to this script, then return the
# the name of the month that corresponds to that number.
# Demonstrates the use the of the case statement.

# Check to see if the user passed a parameter.

if [ $# -eq 0 ]
then
     echo "Error. Please pass an arguement that is a"
     echo "number between 1 and 12."
     exit 1
fi

# set numb equal to argument passed for use in the script
numb=$1

################################################
# The example of a case statement:

case $numb in

1)
echo "January"
;;
2)
echo "February"
;;
3)
echo "March"
;;
4)
```

```
echo "April"
5)
echo "May"
;;
6)
echo "June"
;;
7)
echo "July"
;;
8)
echo "August"
;;
9)
echo "September"
;;
10)
echo "October"
;;
11)
echo "November"
;;
12)
echo "December"
;;
*)
     echo "Error. No month matches that number"
     echo "Please pass an arguement that is a"
     echo "number between 1 and 12."
     exit 2
     ;;

esac

##########################################

exit 0
```

Lab 2: Script Arguments and Usage Information

```
#!/bin/bash
##
# usageExample will accept a directory name then return it
# back to the user. If no argument is given a usage message
# is returned
```

```
#
# check to see if they user put in the parameter.
if [ $# -eq 0 ]
then
     echo "Usage: usageLab word"
     exit 1
fi

echo $1

########################################

exit 0
```

Lab 3: Randomness

```
#!/bin/bash
##
# randomExample will accept a word then return it
# back to the user with with a random number as part
# of the word If no argument is given a usage message
# is returned
## Can be used, for example, to provide random files names
#

# check to see if they user put in the parameter.

if [ $# -eq 0 ]
then
     echo "Usage: randomExample word"
     exit 1
fi

randNumb=$RANDOM

echo "$1-$randNumb"

##############################################

exit 0
```

Lab 4: Strings

```
#!/bin/bash
##
## stringsLab
## demonstrates some strings operations
## does not test the input for blank lines
##########################################
##
## get two strings from the user
##
echo "Enter in a string: "
read mystring1
echo "Enter in a second string: "
read mystring2

#------------------------------------
## test command

echo "Is string 1 zero length? Value of 1 means FALSE"
test -z $mystring1
echo $?

echo "Is string 2 nonzero length? Value of 0 means TRUE;"
test -n $mystring2

echo $?

####################################
## demonstrates comparing the lengths of two string
##

myLen1=${#mystring1}
myLen2=${#mystring2}

if [ $myLen1 -gt $myLen2 ]
then
echo "String 1 is longer than string 2"
else
if [ $myLen2 -gt $myLen1 ]
then
echo "String 2 is longer than string 1"
else
echo "String 1 is the same length as string 2"
fi
```

```
fi

#############################################
## compare the two strings to see if they are the same
##
if [ $mystring1 == $mystring2 ]
then
echo "String 1 is the same as string 2"
else
     if [ $mystring1 != $mystring2 ]
          then
     echo "String 1 is not the same as string 2"
     fi
fi
```