Lab 8

Isaiah Hoffer

PART A:

LAB Q A1 RESULT:



LAB Q A1 SCRIPT:

#! /bin/bash

# lab8\_arrays.sh

#LAB-Q A1

#-------------------------------------------------

#Creating an Array

declare -a array\_1

#Filling Array {10..15}

for ((i = 0; i <= 5; i++))

do

array\_1[$i]=$((10 + i))

done

#LAB-Q A1.1

#Printing Length Of Array

echo ${#array\_1[\*]}

#LAB-Q A1.2

#Printing Array Values

echo "Values In Array\_1: "

for i in ${array\_1[@]}

do

echo $i

done

LAB Q A2 RESULT:



LAB Q A2 SCRIPT:

#LAB-Q A2

#--------------------------------------------------

#Creating New Array

array\_2=("bash" "is the" "coolest" "scripting language")

#LAB-Q A2.1

#Printing Array

echo "Original Array: ${array\_2[\*]}"

#LAB-Q A2.2

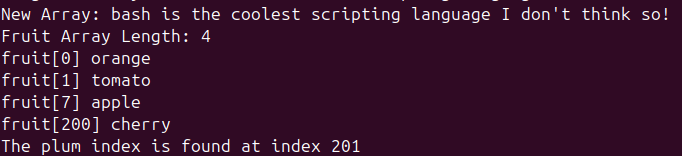
#Adding String

array\_2+=( "I don't think so!")

#Displaying New Array

echo "New Array: ${array\_2[\*]}"

LAB Q A3 RESULT:



LAB Q A3 SCRIPT:

#LAB-Q A3

#--------------------------------------------------

#Creating Another Array

fruit[0]=orange fruit[1]=tomato fruit[7]=apple fruit[200]=cherry

#LAB-Q A3.1

#Printing Fruit Array Size

echo "Fruit Array Length: ${#fruit[\*]}"

#LAB-Q A3.2

#Printing Array Content

for i in ${!fruit[@]}

do

echo "fruit[$i] ${fruit[$i]}"

done

#LAB-Q A3.3

#Adding String

fruit+=("plum")

for i in ${!fruit[@]}

do

if [[ ${fruit[$i]} == "plum" ]]

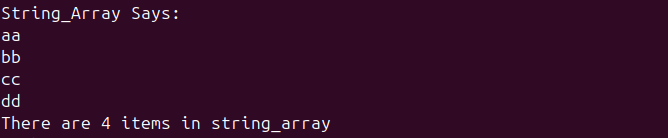
then

echo "The plum index is found at index $i"

fi

done

LAB Q A4 RESULT:



LAB Q A4 SCRIPT:

#LAB-Q A4

#---------------------------------------------------

#Another Array

string1="aa,bb,cc,dd"

#Putting string1 into new array, string\_array

IFS=, read -a string\_array <<< $string1

#Displaing Array

echo "String\_Array Says: "

for i in ${string\_array[@]}

do

echo $i

done

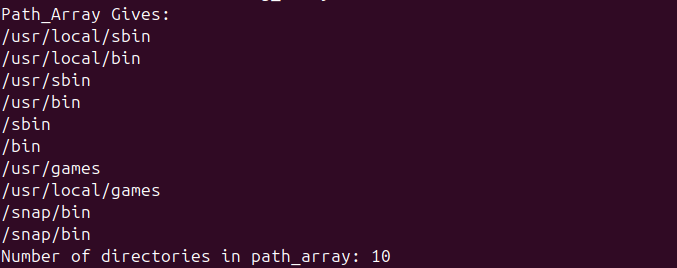
#Words in Array

num\_items=${#string\_array[@]}

#Displaying Results

echo "There are $num\_items items in string\_array"

LAB Q A5 RESULT:



LAB Q A5 SCRIPT:

#LAB-Q A5

#--------------------------------------------------

#New Array

IFS=: path\_array=($PATH)

#LAB-Q A5.1

#Printing Array

echo "Path\_Array Gives: "

for i in ${path\_array[@]}

do

echo $i

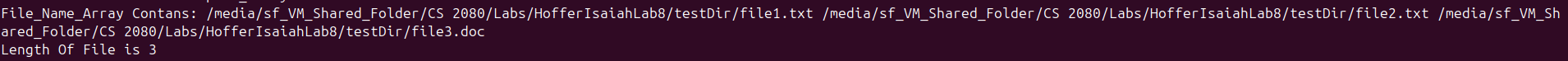
done

#LAB-Q A5.2

#Number of Directories

echo "Number of directories in path\_array: ${#path\_array[@]}"

LAB Q A6 RESULT:



LAB Q A6 SCRIPT:

#LAB-Q A6

#---------------------------------------------------

#Another Array

dir=$(pwd)

file\_name\_array=($dir/testDir/\*.txt $dir/testDir/\*.doc)

#LAB-Q A6.1

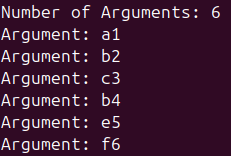
#Pring Length And Contents

echo "File\_Name\_Array Contans: ${file\_name\_array[@]}

Length Of File is ${#file\_name\_array[@]}"

PART B:

LAB Q B1 RESULT:



LAB Q B1 SCRIPT:

#! /bin/bash

# lab8\_functions.sh

#LAB-Q B1

#-----------------------------------------------------------------------

#Function to take any # of arguments, display arguments, # of arguments

arg\_counter() {

echo "Number of Arguments: $#"

for i in $@

do

echo "Argument: $i"

done

} #arg\_counter Function

#Calling arg\_counter Function

arg\_counter a1 b2 c3 b4 e5 f6

LAB Q B2 RESULTS:



LAB Q B2 SCRIPT:

#LAB-Q B2.1

#------------------------------------------------------

#Calculates Average

average() {

#Variables

sum=0

totalArgs=$#

#Need At least one Integer Argument

if [ totalArgs == 0 ]; then

echo "Need Integer Argument!"

return 1

fi

#Finding Sum of Arguments

for i in $@; do

sum=$((sum + i))

done

#Getting Average, 2 decimal places

average=$(echo "scale=2; $sum / $totalArgs" | bc)

#Printing Average

echo "Average: $average"

return $?

} #average Function

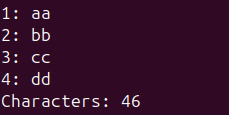
#LAB-Q B2.2

#Calling average Function

average\_result=$(average 3 66 92 8 14 4)

echo "Average $average\_result"

LAB Q B3 RESULT:



LAB Q B3 SCRIPT:

LAB-Q B3.1

#------------------------------------------------------------

#

line\_nums() {

line\_nums=1

while read line; do

echo "$line\_nums: $line"

((line\_nums++))

done

} #line\_nums Filter Function

#Calling Filter Function line\_nums

echo -e "aa \n bb \n cc \n dd" | line\_nums

#LAB-Q B3.2

#Count Amount Of Characters

chars() {

count=0

while read line; do

count=$((count + ${#line}))

done

echo "Characters: $count"

} # chars Filter Function

#Calling Filter Function chars

echo The quick brown fox quietly strummed the banjo | chars

PART C:

LAB Q C1 SCRIPT:

! /bin/bash

# lab8.libary

#LAB-Q C1

#--------------------------------------

#Max function, returns greater value

max() {

if [ $1 >= $2 ]; then

echo $1

else

echo $2

fi

}

#Min Function, returns smaller value

min() {

if [ $1 <= $2 ]; then

echo $1

else

echo $2

}

LAB Q C2 RESULT:



LAB Q C2 SCRIPT:

#LAB-Q C2

#--------------------------------------------

#Getting Libary Functions

source ./lab8.library

#Initalizing Variables

var\_a=24

var\_b=1

var\_c=19

#LAB-Q C2.1

#Calling Max Function

echo "Max: $(max $var\_a $var\_b)"

#Calling Min Function

echo "Min: $(min $var\_b $var\_c)"

LAB Q C3 PUZZLE:

First It declares an array, colors, and fills specific indexes with the names of colors. Next a new array is created named slots that are filled with only the used indexes from colors, not strings! Next the code gets the amount of index strings there are (length) and then goes through a for loop starting at the length of the slots array. The for loop starts at the end of the array and uses the slots array as an index for the original array to get the strings starting from the indexes farthest from 0, so it displays: teal green yellow blue