Damon Britt Lab 1

1.

#!/bin/bash

DATE=`date '+%d%b%Y@%H:%M:%S'`

HOSTNAME=`hostname`

IP=`hostname -I | awk {print\ $1}`

MEM=`grep -i MemFree /proc/meminfo | awk '{print $2 " " $3}'`

CORES=`grep -i cpu\ core /proc/cpuinfo | awk '{print $4}'`

echo -e Hostname ' \t \t \t ' IP Address ' \t \t ' Free Memory ' \t ' CPU Cores >> report\_"$DATE".log

echo -e $HOSTNAME ' \t ' $IP ' \t ' $MEM ' \t ' $CORES >> report\_"$DATE".log

2.

ps aux | wc -l

3.

#!/bin/bash

echo "Please enter a name to verify as a User: "

read User

if [ -z "$User" ]

then

echo "You didn't enter a name. Aborting script."

exit

fi

if grep User: /etc/passwd | grep -q "^$User:"

then

echo "$User is a User."

exit 0

else

echo "No user found."

exit 1

fi

Lab 2

1./2.

#!/bin/bash

DATE=`date '+\_%d%b%Y'`

echo -n "Name of file to backup: "

read name

if [ -f "$name" ];

then

if [ -f "$name".bak"$DATE" ]

then

echo -n "This file already exists. Do you want to overright it?(Y or N): "

read YorN

if [[ "${YorN}" == y || "${YorN}" == Y ]]

then cp "$name" "$name".bak"$DATE"

echo "You chose to overright your file. Hopefully you didn't lose anything"

else

echo "You chose not to overright your file. So why you here?"

fi

else

cp "$name" "$name".bak"$DATE"

echo "$name" was backed up.

fi

else

echo "No file with that name found. Aborting script."

fi

3.

#!/bin/bash

echo "Please enter a number: "

read num

if [[ $num -gt 10 && $num%2 -ne 0 ]];

then

echo You win!

else

echo You fail!

fi

4.

#!/bin/bash

echo "Please enter a filename: "

read name

if [[ -f $name ]]

then

ls -l $name

else

echo The filename you entered was not found.

fi

5.

#!/bin/bash

echo "Enter your first number: "

read first

echo "Enter your second number: "

read second

if [[ $first =~ ^-?[0-9]+$ ]] || [[ $second =~ ^-?[0-9]+$ ]];

then

echo "Enter the number associated to the function to be performed: "

echo "1. Addition"

echo "2. Subtraction"

echo "3. Multiplication"

echo "4. Division"

echo "5. Modulus"

read num

case $num in

1)

echo "$first + $second = $(($first+$second))";;

2)

echo "$first - $second = $(($first-$second))";;

3)

echo "$first \* $second = $(($first\*$second))";;

4)

echo "$first / $second = $(($first/$second))";;

5)

echo "$first % $second = $(($first%$second))";;

\*)

echo "You didn't select an appropriate function.";;

esac

else

echo "One of your numbers was invalid. Aborting Script."

exit

fi

Lab 3

1.

#!/bin/bash

while :

do

echo "Enter a number: "

read first

echo "Enter a second number: "

read second

if [[ $first =~ ^-?[0-9]+$ ]] || [[ $second =~ ^-?[0-9]+$ ]];

then

if [[ $first -gt 9 ]] || [[ $second -gt 9 ]];

then

echo "$first + $second = $(($first+$second))"

break

else

echo "Your numbers are $first and $second. Try raising both numbers 10 or higher."

fi

else

echo "You entered an invalid number. Try again."

fi

done

2.

#!/bin/bash

echo "Enter classmate IP address for port look up."

read ip

sudo nmap -sT -p- $ip

3.

#!/bin/bash

IFS=$'\n'

mounted=`mount | awk '{print "Drive: " $1 ", Filetype: " $5 ", Mounted on: " $3}'`

for line in $mounted;

do

echo "$line"

done

Lab 4.

1.

\* \* \* \* \* DiskSpace/diskOver10.sh

#1/bin/usage

used=`df -h | awk '{print $5}' | sort -nr | head -n 1 | sed 's/%//'`

if [[ $used -gt 10 ]];

then

echo Disk Space Utilization at $used%. $(date) >> DiskSpace/usageReport.log

fi

2.

#!/bin/bash

DATE=`date '+%d%b%Y@%H:%M:%S'`

if [[ -d $1 ]]

then

sudo tar -cvf ~/$1\_"$DATE".tar $1

else

echo Invalid Directory

fi

3.

#!/bin/bash

if grep -q "^$1:" /etc/passwd

then

echo $1 is already a User.

echo $1 is already a User. >> users.log

else

sudo useradd -c “User” $1

echo Added $1 as a user.

fi

Lab 5.

1.

#1/bin/bash

echo Enter 10 string values.

read -a array

for value in ${array[@]}

do

echo $value

done

2.

#!/bin/bash

num=5

while [ $num -le 100 ]

do

echo $num

num=$(($num + 5))

done

3.

#!/bin/bash

echo "Please enter the size of your matrix: "

read size

element=0

array=()

while [ $element -lt $size ]

do

array+=($element)

element=$(($element + 1))

if [ $element -eq $size ]

then

echo ${array[\*]}

fi

done

4.

#!/bin/bash

States=("New Mexico" "New York" "North Carolina" "North Dakota" "Ohio" "Oklahoma" "Oregon" "Pennsylvania" "Rhode Island" "South Carolina" "South Dakota" "Tennessee" "Texas" "Utah" "Vermont" "Virginia" "Washington" "West Virginia" "Wisconsin" "Wyoming" "Alabama" "Alaska" "Arizona" "Arkansas" "California" "Colorado" "Connecticut" "Delaware" "Florida" "Georgia" "Hawaii" "Idaho" "Illinois" "Indiana" "Iowa" "Kansas" "Kentucky" "Louisiana" "Maine" "Maryland" "Massachusetts" "Michigan" "Minnesota" "Mississippi" "Missouri" "Montana" "Nebraska" "Nevada" "New Hampshire" "New Jersey")

IFS=$'\n'

sorted=($(sort <<<"${States[\*]}"))

unset IFS

printf "%s\n" "${sorted[@]}"

5.

#!/bin/bash

var1=$HOSTNAME

var2=$USER

var3=$PWD

echo Hostname: $var1 User: $var2 PWD: $var3

6.

#!/bin/bash

echo "Please enter three numbers you wish to have bubble sorted: "

read -a num

for (( i = 0; i < ${#num[@]}; i++ ))

do

for (( j = $i; j < ${#num[@]}; j++ ))

do

if [ ${num[$i]} -gt ${num[$j]} ];

then

t=${num[$i]}

num[$i]=${num[$j]}

num[$j]=$t

fi

done

done

echo ${num[\*]}

7.

#!/bin/bash

menu=y

echo "Enter the number corresponding to the day of the week to receive your workout."

echo "1 - Sunday"

echo "2 - Monday"

echo "3 - Tuesday"

echo "4 - Wednesday"

echo "5 - Thursday"

echo "6 - Friday"

echo "7 - Saturday"

echo "8 - Quit"

while [ "$menu" = y ]

do

read day

case $day in

1) echo Jumping Jacks

menu=z ;;

2) echo Wall Sit

menu=z ;;

3) echo Push-Ups

menu=z ;;

4) echo Abdominal Crunch

menu=z ;;

5) echo Squats

menu=z ;;

6) echo Run

menu=z ;;

7) echo Planks

menu=z ;;

8) echo THERE IS NO QUITING! SELECT YOUR WORKOUT! ;;

\*) echo Invalid choice. Try again. ;;

esac

done

echo I hope you enjoyed your workout.

Lab 6.

1.