DPS918 Lab 3

Some important notes on this Lab:

(1) You do NOT have to do this whole Lab in one sitting! Any completed

sections will be tracked.

(2) You may abort the Lab at any time by pressing Ctrl-c but you may lose

any work done on an incomplete section.

(3) Instructions for each page will be displayed at the bottom of the screen.

(4) The answers expected in this Lab are derived from the Lab itself and the

course notes. The course notes should be used as a reference, if required. Using

Google or another search engine may provide answers that are not accepted by

the Lab.

(5) This Lab is maintained by Les Czegel. Please try to resolve any problems

with your instructor who will determine if Les should be contacted.

---------------------------------------------------------------------------------

Press <ENTER> to continue (or: 'm'-main menu).

DPS918 Lab 3: Selection Menu

----------------------------------------------------------------------------------------------------------

Available selections:

1 Introduction To Scripting (phone)

2 More Scripting (add)

3 Yet More Scripting (oldfiles)

4 Scripting (saferm)

5 Submit DPS918 Lab 3

The highlighted lines above indicate the incomplete parts of the Lab.

There is a total possible mark of 4, with 1 mark deducted for each

incomplete part. Late marks will be deducted at the rate of 0.4 marks (10%)

per day for Labs submitted after midnight of June 2 2017.

DPS918 Lab 3.1: Introduction To Scripting (phone)

----------------------------------------------------------------------------------------------------------

Available selections:

1 'phone' script

2 'phone2' script

3 'phone3' script

4 'phone4' script

Note that the highlighted lines indicate the incomplete sections of

this part of the Lab.

Section 4 can only be done when the other sections have been

completed.

DPS918 Lab 3.1: 'phone' Script Page 1/11

----------------------------------------------------------------------------------------------------------

You're going to write 4 different versions of a script, adding functionality at each stage. Let's start

by creating a directory to contain the scripts, and any other scripts you'd like to place into it.

Open up a new shell window, so you can read the instructions in this window while executing them in

the other window.

If you haven't already done so, create a directory called 'scripts' within your home directory.

Then hit 'Enter' on this window, it will check to ensure the directory was created:

DPS918 Lab 3.1: 'phone' Script Page 3/11

----------------------------------------------------------------------------------------------------------

In the other shell window, display the file:

~dps918/2017b/phonebook

This is the file that you will be using, within the 'phone' script, to display the record matching a requested name.

cat ~dps918/2017b/phonebook

Try entering the following (in the other window):

grep -i cheryl ~dps918/2017b/phonebook

hhuang66@matrix:~> grep -i cheryl ~dps918/2017b/phonebook

CREATORE CHERYL LE 6251 HEALTH SCIENCES

Now create a file called 'phone', within the ~/scripts directory. It should start with a line indicating that it needs to run in a bash shell, followed by the same line you just ran:

touch ~/scripts/phone

vi ~/scripts/phone

press Insert

type #!/bin/bash

type grep -i cheryl ~dps918/2017b/phonebook

#!/bin/bash

grep -i cheryl ~dps918/2017b/phonebook

Try using vi for this, just for the practice.

~/scripts/phone must be created to continue

Now give yourself execute permission for the 'phone' script, for example:

chmod u+x phone

Also, add your 'scripts' directory to your PATH:

PATH=$PATH:~/scripts

Now try running 'phone' and make sure it works.

|  |
| --- |
| hhuang66@matrix:~> grep -i cheryl ~dps918/2017b/phonebook  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> touch ~/scripts/phone  hhuang66@matrix:~> vi ~/scripts/phone  #!/bin/bash  grep -i cheryl ~dps918/2017b/phonebook  hhuang66@matrix:~> chmod u+x ~/scripts/phone  hhuang66@matrix:~> chmod u+x ~/scripts/phone  hhuang66@matrix:~> PATH=$PATH:~/scripts  hhuang66@matrix:~> ~/scripts/phone  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> |

Now change the 'phone' script so that it will allow the name to be specified as a

command line argument. For example, 'phone cheryl' should display the results for

'cheryl', and 'phone joel' should display the results for 'joel'. To do this,

change the second line in the script so that instead of searching for 'cheryl',

it searches for $1:

grep -i $1 ~dps918/2017b/phonebook

Now try running 'phone cheryl' and 'phone joel' and make sure your script works.

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/phone  #!/bin/bash  grep -i cheryl ~dps918/2017b/phonebook  #!/bin/bash  grep -i $1 ~dps918/2017b/phonebook  hhuang66@matrix:~> phone cheryl  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> phone joel  SHAPRIO JOEL SH 4802 BUSINESS STUDIES DIVISION  hhuang66@matrix:~> |

Good!

You may now proceed to the next version of the script.

DPS918 Lab 3.1: 'phone2' Script Page 1/4

----------------------------------------------------------------------------------------

Copy your script 'phone' to 'phone2'.

Now try running 'phone2 cheryl' and 'phone2 joel' (in another shell window) and make

sure your script works.

|  |
| --- |
| hhuang66@matrix:~> cp ~/scripts/phone ~/scripts/phone2  hhuang66@matrix:~> phone2 cheryl  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> phone2 joel  SHAPRIO JOEL SH 4802 BUSINESS STUDIES DIVISION |

Change 'phone2' to prompt the user for a name to search for in the phonebook.

In order to do that, add the following two lines before the grep:

echo -n "Enter a name to search for: "

read name

In order to use the name typed in by the user, the 'grep' needs to use the variable

value $name instead of $1. Now the user can simply enter 'phone2', then enter a

name to search for in response to the prompt.

Make the necessary changes, and try your 'phone2' script out, using both the names

'cheryl' and 'joel'.

'phone2' is not giving the correct output, make sure that it is correct,

with the correct permissions, and that 'scripts' is in your PATH

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/phone2  #!/bin/bash  echo -n "Enter a name to search for: "  read name  grep -i $name ~dps918/2017b/phonebook  hhuang66@matrix:~> phone2 cheryl  Enter a name to search for: cheryl  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> phone2 joel  Enter a name to search for: joel  SHAPRIO JOEL SH 4802 BUSINESS STUDIES DIVISION |

Good!

You may now proceed to the next version of the script.

Enter a menu selection or 'q' to quit: 3

DPS918 Lab 3.1: 'phone3' Script Page 1/4

---------------------------------------------------------------------------------

Copy your script 'phone2' to 'phone3'.

Now try running 'phone3' for both 'cheryl' and 'joel' (in another shell window)

and make sure your script works.

|  |
| --- |
| hhuang66@matrix:~> cp ~/scripts/phone2 ~/scripts/phone3  hhuang66@matrix:~> phone3 cheryl  Enter a name to search for: cheryl  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> phone3 joel  Enter a name to search for: joel  SHAPRIO JOEL SH 4802 BUSINESS STUDIES DIVISION |

Change 'phone3' to prompt the user for a name to search for in the phonebook, but

only if a name was not given as a positional parameter. In other words, 'phone3

cheryl' would search for 'cheryl', but 'phone3' would request a name to search for.

In order to do that, we'll assign the value of $1 to the variable 'name'. Then we'll

use an 'if' statement to determine if $name has a value. If not, then we'll use a

'read' statement to get a value from the user.

Use the following five lines to replace the 'echo' and 'read' statements:

name=$1

if [ "$name" = "" ]

then echo -n "Enter a name to search for: "

read name

fi

Make the necessary changes, and try your 'phone3' script out, using both the names

'cheryl' and 'joel' as positional parameters and as values read from the user.

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/phone3  #!/bin/bash  name=$1  if [ "$name" = "" ]  then echo -n "Enter a name to search for: "  read name  fi  grep -i $name ~dps918/2017b/phonebook  hhuang66@matrix:~> ~/scripts/phone3 cheryl  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> ~/scripts/phone3 joel  SHAPRIO JOEL SH 4802 BUSINESS STUDIES DIVISION |

Good!

You may now proceed to the next version of the script.

DPS918 Lab 3.1: 'phone4' Script Page 1/5

------------------------------------------------------------------------------------

Shell Scripting Exercise

The purpose of this section is to reinforce your skills with writing UNIX scripts

covered in the previous sections.

Copy your script 'phone3' to 'phone4'.

Test 'phone4' and make sure your script works.

|  |
| --- |
| hhuang66@matrix:~> cp ~/scripts/phone3 ~/scripts/phone4  hhuang66@matrix:~> ~/scripts/phone3 cheryl  CREATORE CHERYL LE 6251 HEALTH SCIENCES 250  hhuang66@matrix:~> ~/scripts/phone3 joel  SHAPRIO JOEL SH 4802 BUSINESS STUDIES DIVISION |

Try running 'phone4 xyz' and see what happens.

Modify your program so that if no matching name is found, an appropriate message

is displayed: "Name 'xyz' not in directory".

You could use an 'if' statement to check the value of $? to see if the grep command

was successful (remember, '0' indicates success). If the grep is NOT successful,

then echo the message (which includes the value of $name). Give it a try.

Make sure this works with both command line arguments and with a name read in

from the user, and make sure you use the message EXACTLY as shown.

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/phone4  #!/bin/bash  name=$1  if [ "$name" = "" ]  then echo -n "Enter a name to search for: "  read name  fi  grep -i $name ~dps918/2017b/phonebook    if [ $? != 0 ]  then  echo "Name $name' not in the directory"  fi  hhuang66@matrix:~> ~/scripts/phone4 xyz  Name xyz' is not in the directory |

Congratulations!

You have successfully completed 'Introduction to Shell Scripting'!

DPS918 Lab 3.1: Introduction To Scripting (phone)

------------------------------------------------------------------------------------

Available selections:

1 'phone' script - Completed

2 'phone2' script - Completed

3 'phone3' script - Completed

4 'phone4' script - Completed

Note that the highlighted lines indicate the incomplete sections of this part of the Lab.

Section 4 can only be done when the other sections have been

completed.

DPS918 Lab 3.2: More Scripting (add)

------------------------------------------------------------------------------------

Available selections:

1 'add' script

2 'add2' script

Note that the highlighted lines indicate the incomplete sections of

this part of the Lab.

Section 2 can only be done when the other section has been completed.

DPS918 Lab 3.2: 'add' Script Page 1/10

------------------------------------------------------------------------------------

You're going to write 2 versions of a script, adding functionality at each stage.

Open up a new shell window, so you can read the instructions in this window

while executing them in the other window.

If you haven't already done so, create a directory called 'scripts' within your

home directory.

If you haven't already done so, edit the file ".bashrc" in your home directory

and add the following two lines at the end of the file:

PATH=$PATH:~/scripts

umask 077

The first line will ensure that you can execute your scripts regardless of your

current directory, and the second line will ensure that other students can't

copy your files.

If you don't have a file called ".bashrc" in your home directory, create it and

add the above two lines.

|  |
| --- |
| hhuang66@matrix:~> vi .bashrc  # Sample .bashrc for SuSE Linux  # Copyright (c) SuSE GmbH Nuernberg  # There are 3 different types of shells in bash: the login shell, normal shell  # and interactive shell. Login shells read ~/.profile and interactive shells  # read ~/.bashrc; in our setup, /etc/profile sources ~/.bashrc - thus all  # settings made here will also take effect in a login shell.  #  # NOTE: It is recommended to make language settings in ~/.profile rather than  # here, since multilingual X sessions would not work properly if LANG is over-  # ridden in every subshell.  # Some applications read the EDITOR variable to determine your favourite text  # editor. So uncomment the line below and enter the editor of your choice :-)  #export EDITOR=/usr/bin/vim  #export EDITOR=/usr/bin/mcedit  # For some news readers it makes sense to specify the NEWSSERVER variable here  #export NEWSSERVER=your.news.server  # If you want to use a Palm device with Linux, uncomment the two lines below.  # For some (older) Palm Pilots, you might need to set a lower baud rate  # e.g. 57600 or 38400; lowest is 9600 (very slow!)  #  #export PILOTPORT=/dev/pilot  #export PILOTRATE=115200  test -s ~/.alias && . ~/.alias || true  PATH=$PATH:~/scripts  umask 077 |

If you have changed the ".bashrc" file during this session, enter the "bash"

command by itself in order to start an interactive subshell. This well allow

".bashrc" to execute. Pay close attention to make sure there are no errors.

|  |
| --- |
| hhuang66@matrix:~> bash |

You will write a bash shell script called 'add' that satisfies the following

requirements:

Usage: add number-list

'add' will add the numbers (integers) in the list and display the total. For

example:

add 4 -3 12 9

will produce the number "22" as output.

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/add  #!/bin/bash  sum=0  for number  do  sum=$ (( sum + number ))  done  echo $sum |

Start by creating a file called 'add', within the ~/scripts directory. It should

begin with a line indicating that it needs to run in a bash shell:

#!/bin/bash

Try using vi for this, just for the practice.

Within 'add', use the "for" control structure to loop through all the positional

parameters. Use a meaningful loop variable, for example "number" would make sense.

Within the loop, just echo the value of the "number" variable, to make sure

you're looping the right way.

Save the program, give yourself execute permission, and test it out.

For example:

add 4 -3 12 9

should produce the output:

4

-3

12

9

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/add  #!/bin/bash  sum=0  for number  do  echo $number | grep "[^0-9+-]" >/dev/null  sum=$ (( sum + number ))  done  echo $sum  ~  hhuang66@matrix:~> chmod ugo+x ~/scripts/add  hhuang66@matrix:~> ~/scripts/add 4 -3 12 9  /home/hhuang66/scripts/add: line 7: syntax error near unexpected token `('  /home/hhuang66/scripts/add: line 7: ` sum=$ (( sum + number ))'  hhuang66@matrix:~> vi ~/scripts/add  # |

Make sure the current version of your script works correctly before going on.

Next, initialize a variable called "sum" to 0, before the "for" loop. You will

use this variable to add each positional variable one-by-one, so starting it at

zero should make sense.

Don't worry about the adding for now, but echo out the value of the variable

"sum" after the loop.

Test your script again, make sure it displays the positional parameters and the

zero at the end.

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/add  #!/bin/bash  sum=0  for number  do  echo $number | grep "[^0-9+-]" >/dev/null  if [[ $? = 0 ]]  then  echo "Sorry, '$number' is not a number"  exit 1  fi  sum=$(( sum + number ))  done  echo $sum  hhuang66@matrix:~> ~/scripts/add 4 -3 12 9  22 |

Finally, within the loop, instead of echoing out the value of the loop variable

"number", add it to the variable "sum".

Use the x=$((x + y)) arithmetic format, as described in the lecture notes.

Again, check that it works. Make sure that

add 4 -3 12 9

will produce the number "22" as output. Try 'add' with other numbers as well.

Good!

You may now proceed to the next version of the script.

DPS918 Lab 3.2: 'add2' Script Page 1/9

------------------------------------------------------------------------------------

Shell Scripting Exercise

The purpose of this section is to reinforce your skills with writing UNIX scripts

covered in the previous sections.

Copy your script 'add' to 'add2'.

Try running:

add2 4 -3 twelve nine

You will change the script so that it will display the following message EXACTLY:

add2 4 -3 twelve nine

Sorry, 'twelve' is not a number

Notice that only the first problem was found, and then

the script terminated. Hit <Enter> to explore an appropriate

approach.

|  |
| --- |
| hhuang66@matrix:~> cp ~/scripts/add ~/scripts/add2  hhuang66@matrix:~> add2 4 -3 twelve nine  Sorry, 'twelve' is not a number |

The best approach is to check each positional parameter within the loop, before

the addition.

You will use an "echo" to display the value of the "number" variable, and pipe the

output into a "grep". The "grep" should search for a single character which is

NOT a digit or a plus or minus sign. A character class, something like "[^0-9+-]"

would be the easiest way to do this.

If the "grep" is successful, then the message should be displayed and the script

terminated.

Continue to the next step to see how to start.

Start this step by just adding the correct "echo" piped into the correct "grep",

without redirecting the output, and make sure the script acts appropriately,

for example:

add2 4 -3 12 9

22

add2 4 -3 twelve nine

twelve

nine

1

Now add an "if" structure right after the "echo ... | grep ...", testing the

exit status of the "grep". If the "grep" was successful, then the appropriate

message should be displayed, and the script terminated. Make sure the script

acts appropriately, for example:

add2 4 -3 12 9

22

add2 4 -3 twelve nine

twelve

Sorry, 'twelve' is not a number

Finally, redirect the output of the grep to "/dev/null". Make sure the script

acts correctly, for example:

add2 4 -3 12 9

22

add2 4 -3 twelve nine

Sorry, 'twelve' is not a number

Congratulations!

You have successfully completed 'More Scripting'!

Available selections:

1 'add' script - Completed

2 'add2' script - Completed

Note that the highlighted lines indicate the incomplete sections of

this part of the Lab.

Section 2 can only be done when the other section has been completed.

DPS918 Lab 3.3: Yet More Scripting (oldfiles)

------------------------------------------------------------------------------------

Available selections:

1 'oldfiles' script

2 'oldfiles2' script

Note that the highlighted lines indicate the incomplete sections of

this part of the Lab.

Section 2 can only be done when the other section has been completed.

DPS918 Lab 3.3: 'oldfiles' Script Page 1/10

------------------------------------------------------------------------------------

You're going to write 2 versions of a script, adding functionality at each stage.

Open up a new shell window, so you can read the instructions in this window

while executing them in the other window.

If you haven't already done so, create a directory called 'scripts' within your

home directory.

If you haven't already done so, edit the file ".bashrc" in your home directory

and add the following two lines at the end of the file:

PATH=$PATH:~/scripts

umask 077

The first line will ensure that you can execute your scripts regardless of your

current directory, and the second line will ensure that other students can't

copy your files.

If you don't have a file called ".bashrc" in your home directory, create it and

add the above two lines.

DPS918 Lab 3.3: 'oldfiles' Script Page 3/10

------------------------------------------------------------------------------------

If you have changed the ".bashrc" file during this session, enter the "bash"

command by itself in order to start an interactive subshell. This well allow

".bashrc" to execute. Pay close attention to make sure there are no errors.

DPS918 Lab 3.3: 'oldfiles' Script Page 4/10

------------------------------------------------------------------------------------

You will write a bash shell script called 'oldfiles' which takes one argument,

the name of a directory, and adds the extension ".old" to all visible files

in the directory that don't already have it. Treat subdirectories the same as

ordinary files. For example:

$ ls

file1 file2.old file3old file4.old

$ oldfiles .

$ ls

file1.old file2.old file3old.old file4.old

|  |
| --- |
| hhuang66@matrix:~> vi oldfiles  #!/bin/bash  for filename in $(ls $1)  do  echo $filename | grep "\.old$" > /dev/null  if [[ $? != 0 ]]  then  mv $1/$filename $1/$filename.old  fi  done |

Start by creating a file called 'oldfiles', within the ~/scripts directory.

It should begin with a line indicating that it needs to run in a bash shell:

#!/bin/bash

Try using vi for this, just for the practice.

Within 'oldfiles', use a "for" control structure to loop through all the non-hidden

filenames in the directory name in $1. Check the lecture notes for an example.

Use a meaningful loop variable, for example "filename" would make sense. Also,

use command substitution with "ls $1" instead of an ambiguous filename, or you'll

descend into subdirectories.

Within the loop, just echo the value of the "filename" variable, to make sure

you're looping the right way.

Save the program, give yourself execute permission, and test it out. To aid your

testing, create a test directory and use touch to create some filenames with and

without the ".old" extension. Then test your script and make sure that all the

filenames in the specified directory are displayed.

Make sure the current version of your script works correctly before going on.

Next, within the loop, continue to use an "echo" to display the value of the

"filename" variable, and pipe the output into a "grep". The grep should search

for the ".old" extension. A regular expression such as "\.old$" would be the

easiest way to do this.

Don't worry about changing the names yet, simply let "grep" display the filenames

that already have the extension.

Test your script again, and make sure it displays the correct filenames.

Within the loop, check the exit status of the "grep" with an "if" control

structure. If the "grep" is unsuccessful in finding ".old", then the file should

be renamed.

The renaming can be done with a simple "mv" command, renaming "$1/$filename" to

"$1/$filename.old".

Once this works, don't forget to redirect the "grep" output to "/dev/null".

Again, check that your script works correctly.

Make sure your script is in the correct directory

Please hit <Enter> to continue with the Lab

Make sure that 'oldfiles' is working correctly, with the correct permissions,

and that 'scripts' is in your PATH

|  |
| --- |
| hhuang66@matrix:~> chmod ugo+x oldfiles  hhuang66@matrix:~> oldfiles  mv: cannot stat `/2': No such file or directory  mv: cannot stat `/Acceptable-Use-Policy': No such file or directory  mv: cannot stat `/BTP305': No such file or directory  mv: cannot stat `/BTS530': No such file or directory  m |

Press <ENTER> to continue (or: 'm'-main menu).

Files created in 'testdir':

file1 file2.old file3old file4.old

Execution of 'oldfiles testdir':

Files in 'testdir' after 'oldfiles' was run:

file1.old file2.old file3old.old file4.old

Please hit <Enter> to continue with the Lab

|  |
| --- |
| hhuang66@matrix:~> ls ~/scripts/  add add2 oldfiles phone phone2 phone3 phone4 script1  hhuang66@matrix:~> mkdir ~/scripts/testdir  hhuang66@matrix:~> ls ~/scripts  add add2 oldfiles phone phone2 phone3 phone4 script1 testdir  hhuang66@matrix:~> cd ~/scripts/testdir  hhuang66@matrix:~/scripts/testdir> cd..  hhuang66@matrix:~/scripts> ~/scripts/oldfiles ~/scripts/testdir  hhuang66@matrix:~/scripts> ls ~/scripts/testdir  hhuang66@matrix:~/scripts> cd testdir  hhuang66@matrix:~/scripts/testdir> ls  hhuang66@matrix:~/scripts/testdir> touch file1  hhuang66@matrix:~/scripts/testdir> ls  file1  hhuang66@matrix:~/scripts/testdir> touch file2  hhuang66@matrix:~/scripts/testdir> touch file3old  hhuang66@matrix:~/scripts/testdir> touch file4  hhuang66@matrix:~/scripts/testdir> cd..  hhuang66@matrix:~/scripts> cd  hhuang66@matrix:~>  hhuang66@matrix:~> ~/scripts/oldfiles ~/scripts/testdir  hhuang66@matrix:~> ls ~/scripts/testdir  file1.old file2.old file3old.old file4.old  hhuang66@matrix:~> |

Good!

You may now proceed to the next version of the script.

DPS918 Lab 3.3: 'oldfiles2' Script Page 1/5

------------------------------------------------------------------------------------

Shell Scripting Exercise

The purpose of this section is to reinforce your skills with writing UNIX scripts

covered in the previous sections.

DPS918 Lab 3.3: 'oldfiles2' Script Page 1/5

------------------------------------------------------------------------------------

Shell Scripting Exercise

The purpose of this section is to reinforce your skills with writing UNIX scripts

covered in the previous sections.

DPS918 Lab 3.3: 'oldfiles2' Script Page 2/5

------------------------------------------------------------------------------------

Copy your script 'oldfiles' to 'oldfiles2'.

The way the 'oldfiles' script was written is inefficient from a computer

utilization point of view. We're looping through ALL the filenames, and then

using an "if" inside the loop to determine if each file should be renamed.

It would be more efficient to just loop through the filenames that need to

be changed.

|  |
| --- |
| hhuang66@matrix:~> cp ~/scripts/oldfiles ~/scripts/oldfiles2 |

DPS918 Lab 3.3: 'oldfiles2' Script Page 3/5

------------------------------------------------------------------------------------

Modify the command substitution that's being used to create the loop values that

will be placed into the "filename" variable.

Instead of just an "ls $1", pipe the output into a "grep". The "grep" will search

for all filenames that DO NOT end in ".old". This can easily be done with the

"grep -v" option.

With this approach, you can get rid of the "echo ... | grep ..." and the "if"

control structure inside the loop, and simply do the rename.

Again, check that your script works correctly.

|  |
| --- |
| hhuang66@matrix:~> cp ~/scripts/oldfiles ~/scripts/oldfiles2  hhuang66@matrix:~> vi ~/scripts/oldfiles2  !/bin/bash  for filename in $(ls $1 | grep -v "\.old")  do  mv $1/$filename $1/$filename.old  done |

Files created in 'testdir':

file1 file2.old file3old file4.old

Execution of 'oldfiles2 testdir':

'oldfiles2 testdir' incorrectly produced some output or errors:

|  |
| --- |
| hhuang66@matrix:~/scripts/testdir> ~/scripts/oldfiles2 ~/scripts/testdir  hhuang66@matrix:~/scripts/testdir> ls  file1.old file2.old file3old.old file4.old |

DPS918 Lab 3.3: 'oldfiles2' Script Page 5/5

------------------------------------------------------------------------------------

Congratulations!

You have successfully completed 'Yet More Scripting'!

Available selections:

1 'oldfiles' script - Completed

2 'oldfiles2' script - Completed

Note that the highlighted lines indicate the incomplete sections of

this part of the Lab.

Section 2 can only be done when the other section has been completed.

DPS918 Lab 3.4: Scripting (saferm)

---------------------------------------------------------------------------------

Available selections:

1 'saferm' script

2 'saferm2' script

Note that the highlighted lines indicate the incomplete sections of

this part of the Lab.

Section 2 can only be done when the other section has been completed.

DPS918 Lab 3.4: 'saferm' Script Page 1/11

---------------------------------------------------------------------------------

You're going to write 2 versions of a script, adding functionality at each stage.

Open up a new shell window, so you can read the instructions in this window while executing them in the other window.

If you haven't already done so, create a directory called 'scripts' within your home directory.

DPS918 Lab 3.4: 'saferm' Script Page 2/11

---------------------------------------------------------------------------------

If you haven't already done so, edit the file ".bashrc" in your home directory and add the following two lines at the end of the file:

PATH=$PATH:~/scripts

umask 077

The first line will ensure that you can execute your scripts regardless of your current directory, and the second line will ensure that other students can't copy your files.

If you don't have a file called ".bashrc" in your home directory, create it and add the above two lines.

DPS918 Lab 3.4: 'saferm' Script Page 3/11

---------------------------------------------------------------------------------

If you have changed the ".bashrc" file during this session, enter the "bash" command by itself in order to start an interactive subshell. This well allow ".bashrc" to execute. Pay close attention to make sure there are no errors.

DPS918 Lab 3.4: 'saferm' Script Page 4/11

---------------------------------------------------------------------------------

You will write a bash shell script called "saferm" that satisfies the following requirements:

Usage: saferm -l (or a filename)

saferm is a replacement for the rm utility. Rather than removing files, it moves them into a directory called ".saferm" in the user's home directory. If

"~/.saferm" doesn't exist, it will be created. The "-l" options lists the current contents of the "~/.saferm" directory.

Either the -l option or a valid existing filename must be specified, otherwise a suitable error message will be displayed and the script will terminate.

Hit "Enter" to see some examples of using saferm.

DPS918 Lab 3.4: 'saferm' Script Page 5/11

---------------------------------------------------------------------------------

Here's an example of saferm at work, including the exit status of each

execution:

$ ls -l testdir

total 0

-rw------- 1 joe.brown users 0 Apr 3 23:46 file1

-rw------- 1 joe.brown users 0 Apr 3 23:46 file2

$ saferm

Usage: saferm -l (or a filename)

$ echo $?

1

$ saferm -l testdir/file1

Usage: saferm -l (or a filename)

$ echo $?

1

$ saferm testdir/file2

$ echo $?

0

$ saferm -l

total 0

-rw------- 1 joe.brown users 0 Apr 3 23:46 file2

$ echo $?

0

$ ls -l testdir

total 0

-rw------- 1 joe.brown users 0 Apr 3 23:46 file1

$

DPS918 Lab 3.4: 'saferm' Script Page 6/11

---------------------------------------------------------------------------------

Start by creating a file called 'saferm', within the ~/scripts directory.

It should begin with a line indicating that it needs to run in a bash shell:

#!/bin/bash

Try using vi for this, just for the practice.

DPS918 Lab 3.4: 'saferm' Script Page 7/11

---------------------------------------------------------------------------------

Next, ensure that exactly one argument was passed using an "if" statement. If

anything other than one argument was passed, display the error message "Usage:

saferm -l (or a filename)" and terminate the script with an exit status of 1

(that's the number 1).

Save the program, give yourself execute permission, and test it out.

Try running your script with various numbers of arguments, and check that

the error message displays when appropriate, and that the exit status "$?" is

correct in each case.

DPS918 Lab 3.4: 'saferm' Script Page 8/11

---------------------------------------------------------------------------------

In the next part of the script, ensure that the "~/.saferm" directory exists

using an "if" statement. If the directory doesn't exist, create it.

Again, test your script, making sure that the directory is created when

appropriate.

DPS918 Lab 3.4: 'saferm' Script Page 9/11

---------------------------------------------------------------------------------

In the last part, check if "-l" was passed using an "if" statement. If it was,

then display the files in the "~/.saferm" directory. If "-l" was not passed,

then assume that the argument is a valid filename, and move the filename to

the "~/.saferm" directory.

Again, test your script, making sure that some sample files are moved to the

"~/.saferm" directory. Also make sure that the "-l" option displays the

contents of the "~/.saferm" directory, including any hidden files that were

safely removed.

\*\*\* ~/scripts/saferm does not have execute permission

Please hit <Enter> to continue with the Lab

|  |
| --- |
| hhuang66@matrix:~> vi ~/scripts/saferm  #!/bin/bash  if [[ $# != 1 ]]  then  echo "Usage: saferm -1 (or a filename)" >&2  exit 1  else  ls -a ~ | grep "\.saferm" > /dev/null  if [[ $? != 0 ]]  then  mkdir ~/.saferm  fi  if [[ $1 = "-1" ]]  then  ls -a1 ~/.saferm  else  mv $1 ~/.saferm  fi  fi |

DPS918 Lab 3.4: 'saferm' Script Page 10/11

---------------------------------------------------------------------------------

Make sure that 'saferm' is working correctly, with the correct permissions,

and that 'scripts' is in your PATH

|  |
| --- |
| hhuang66@matrix:~> chmod ugo+x ~/scripts/saferm |

Execution of 'ls -l testdir':

total 0

-rw------- 1 hhuang66 users 0 May 27 21:36 file1

-rw------- 1 hhuang66 users 0 May 27 21:36 file2

Execution of 'saferm':

Usage: saferm -1 (or a filename)

Execution of 'saferm -l testdir/file1':

Usage: saferm -1 (or a filename)

Execution of 'saferm testdir/file1 testdir/file2':

Usage: saferm -1 (or a filename)

Execution of 'saferm testdir/file2':

Execution of 'saferm -l':

\*\*\* 'saferm -l' incorrectly produced no output

\*\*\* 'saferm -l' incorrectly produced some errors:

mv: invalid option -- 'l'

Try `mv --help' for more information.

\*\*\* 'saferm -l' exit status is 1, should be 0

Execution of 'ls -l testdir':

total 0

-rw------- 1 hhuang66 users 0 May 27 21:36 file1

Please hit <Enter> to continue with the Lab

DPS918 Lab 3.4: 'saferm' Script Page 10/11

---------------------------------------------------------------------------------

Make sure that 'saferm' is working correctly, with the correct permissions,

and that 'scripts' is in your PATH

Execution of 'ls -l testdir':

total 0

-rw------- 1 hhuang66 users 0 May 27 21:54 file1

-rw------- 1 hhuang66 users 0 May 27 21:54 file2

Execution of 'saferm':

Usage: saferm -1 (or a filename)

Execution of 'saferm -l testdir/file1':

Usage: saferm -1 (or a filename)

Execution of 'saferm testdir/file1 testdir/file2':

Usage: saferm -1 (or a filename)

Execution of 'saferm testdir/file2':

Execution of 'saferm -l':

total 4

drwx------ 2 hhuang66 users 18 May 27 21:54 .

drwx--x--x 47 hhuang66 users 4096 May 27 21:39 ..

-rw------- 1 hhuang66 users 0 May 27 21:54 file2

Execution of 'ls -l testdir':

total 0

-rw------- 1 hhuang66 users 0 May 27 21:54 file1

Please hit <Enter> to continue with the Lab

DPS918 Lab 3.4: 'saferm' Script Page 11/11

---------------------------------------------------------------------------------

Good!

You may now proceed to the next version of the script.

DPS918 Lab 3.4: 'saferm2' Script Page 1/7

---------------------------------------------------------------------------------

Shell Scripting Exercise

The purpose of this section is to reinforce your skills with writing UNIX

scripts covered in the previous sections.

DPS918 Lab 3.4: 'saferm2' Script Page 2/7

---------------------------------------------------------------------------------

Copy your script 'saferm' to 'saferm2'.

The way the 'saferm' script was written does not properly handle trying

to safely remove a filename that doesn't exist. Try running saferm with a

missing filename, and see what happens.

DPS918 Lab 3.4: 'saferm2' Script Page 3/7

---------------------------------------------------------------------------------

Modify the 'saferm2' script so that if the specified filename doesn't exist,

then display the error message "Filename 'xxx' does not exist", where 'xxx'

will be replaced by the specified filename within single quotes. The script

should then terminate with an exit status of 2.

Hit "Enter" to see some examples of using saferm2.

DPS918 Lab 3.4: 'saferm2' Script Page 4/7

---------------------------------------------------------------------------------

Here's an example of saferm2 at work, including some of the resulting exit

statuses:

$ ls -l testdir

total 0

-rw------- 1 joe.brown users 0 Apr 4 00:48 file1

-rw------- 1 joe.brown users 0 Apr 4 00:48 file2

$ saferm2 testdir/file3

Filename 'testdir/file3' does not exist

$ echo $?

2

$ saferm2

Usage: saferm2 -l (or a filename)

$ echo $?

1

$ saferm2 testdir/file2

$ echo $?

0

$ saferm2 testdir

$ echo $?

0

$ saferm2 -l

total 4

-rw------- 1 joe.brown users 0 Apr 4 00:48 file2

drwx------ 2 joe.brown users 4096 Apr 4 00:51 testdir

$ ls -l testdir

ls: cannot access testdir: No such file or directory

$

DPS918 Lab 3.4: 'saferm2' Script Page 5/7

---------------------------------------------------------------------------------

In order to do this, check if the specified filename exists using an "if"

statement. If the filename doesn't exist, display the error message "Filename

'xxx' does not exist", where 'xxx' is replaced by the specified filename within

single quotes, and terminate the script with an exit status of 2. It makes

sense to do this after the correct number of arguments has been verified.

After making the change, try running your script with various invalid numbers

of arguments, an invalid filename,and a valid filename, and check that the

error messages display when appropriate, and that the exit status "$?" is

correct in each case.

Press <ENTER> to continue (or: 'm'-main menu).

Execution of 'ls -l testdir':

total 0

-rw------- 1 hhuang66 users 0 May 27 22:06 file1

-rw------- 1 hhuang66 users 0 May 27 22:06 file2

Execution of 'saferm2 testdir/file3':

Filename '/var/tmp/dps918.hhuang66.lab3.25135.testdir/file3' does not exist

Execution of 'saferm2':

Usage: saferm -1 (or a filename)

Execution of 'saferm2 -l testdir/file1':

Usage: saferm -1 (or a filename)

Execution of 'saferm2 testdir/file1 testdir/file2':

Usage: saferm -1 (or a filename)

Execution of 'saferm2 testdir/file2':

Execution of 'saferm2 testdir':

Execution of 'saferm2 -l':

total 4

drwx------ 3 hhuang66 users 59 May 27 22:06 .

drwx--x--x 47 hhuang66 users 4096 May 27 22:06 ..

drwxr-xr-x 2 hhuang66 users 18 May 27 22:06 dps918.hhuang66.lab3.25135.testdir

-rw------- 1 hhuang66 users 0 May 27 22:06 file2

Execution of 'ls -l testdir':

ls: cannot access /var/tmp/dps918.hhuang66.lab3.25135.testdir: No such file or directory

Please hit <Enter> to continue with the Lab

DPS918 Lab 3.4: 'saferm2' Script Page 7/7

---------------------------------------------------------------------------------

Congratulations!

You have successfully completed 'Scripting (saferm)'!

DPS918 Lab 3.4: Scripting (saferm)

---------------------------------------------------------------------------------

Available selections:

1 'saferm' script - Completed

2 'saferm2' script - Completed

Note that the highlighted lines indicate the incomplete sections of

this part of the Lab.

Section 2 can only be done when the other section has been completed.

DPS918 Lab 3.5: Submit Lab 3 Page 1/2

---------------------------------------------------------------------------------

Total marks obtained is 4 out of 4.

Lab 3 Section 1 was completed

Lab 3 Section 2 was completed

Lab 3 Section 3 was completed

Lab 3 Section 4 was completed

Submission Confirmation

You may now submit Lab 3 by typing 'Yes'. If you do NOT want to submit at

this time, type 'm' or 'M'.

DPS918 Lab 3.5: Submit Lab 3 Page 2/2

---------------------------------------------------------------------------------

Congratulations!

You have successfully submitted Lab 3!