DSCC-401
Homework Assignment #3

Afnan Mostafa
Date: 09/28/2023

Sample I/O:

Input commands and outputs are shown in maroon and blue, respectively. Also, the reason behind the colored prompt is that I changed my .bashrc profile to make the prompt look like this.

Example:

```
{amostafa@bluehive:/scratch/amostafa}$ du -h legal_cases.tar.gz ← Input
9.7M            legal_cases.tar.gz     ← Output
```

#1.
Downloaded legal_cases.tar.gz and copied it to /scratch/amostafa/ directory using 'scp' command.

Input:

```
scp legal_cases.tar.gz amostafa@bluehive.circ.rochester.edu:/gpfs/fs2/scratch/amostafa/
```

Output:

```
You are attempting to connect to a computer system that is private property. Any and all login attempts are recorded. Unauthorized or improper use of this system may result in civil and criminal penalties.

(amostafa@bluehive.circ.rochester.edu) Password:
legal_cases.tar.gz                                    100%  9906KB  293.6KB/s   00:33
```

Disk Space:

```
{amostafa@bluehive:/scratch/amostafa}$ du -h legal_cases.tar.gz
9.7M		legal_cases.tar.gz
```

Here, the output shows that 'legal_cases.tar.gz' has a disk space of ~9.7 MB.

Size of the file:

```
{amostafa@bluehive:/scratch/amostafa}$ ls -lth
total 9.7M
drwxrws---+ 8 amostafa amostafa 4.0K Sep 26 22:14 tmp
-rw-r-----  1 amostafa amostafa 9.7M Sep 26 12:07 legal_cases.tar.gz
drwxrws---+ 2 amostafa amostafa 4.0K Sep 25 12:12 New Folder
drwxrws---+ 2 amostafa amostafa 4.0K Sep 25 09:31 abcd
-rw-rw----  1 amostafa amostafa    0 Sep  8 16:04 NOT_BACKED_UP
drwxrws---+ 6 amostafa amostafa 4.0K Sep  4 09:42 test-run
```

Here, we can see that 'legal_cases.tar.gz' has a size of ~9.7 MB allocated.

#2.

Uncompress using gunzip:

```
{amostafa@bluehive:/scratch/amostafa}$ gunzip -v legal_cases.tar.gz
legal_cases.tar.gz:	 77.4% -- replaced with legal_cases.tar
```

Unarchive using tar:

```
{amostafa@bluehive:/scratch/amostafa}$ tar -xvf legal_cases.tar
```

A folder titled 'legal cases' with sub-directories of group 1 and group 2 was unarchived which has a total of 1751 files (output is just a long list of these 1751 files, hence intentionally not included here).

#3.

```
{amostafa@bluehive:/scratch/amostafa}$ du -h legal_cases
51M	legal_cases/group2
52M	legal_cases/group1
103M	legal_cases
```

Here, we can see that 'legal_cases' has a disk space of ~103 MB.

```
{amostafa@bluehive:/scratch/amostafa}$ ls -lht
total 43M
drwxrws---+ 8 amostafa amostafa 4.0K Sep 26 22:59 tmp
-rw-rw----  1 amostafa amostafa  43M Sep 26 01:32 legal_cases.tar
drwxrws---+ 2 amostafa amostafa 4.0K Sep 25 12:12 New Folder
drwxrws---+ 2 amostafa amostafa 4.0K Sep 25 09:31 abcd
drwx--S---+ 4 amostafa amostafa 4.0K Sep 24 17:47 legal_cases
-rw-rw----  1 amostafa amostafa    0 Sep  8 16:04 NOT_BACKED_UP
drwxrws---+ 6 amostafa amostafa 4.0K Sep  4 09:42 test-run
```

Here, we can see that the 'legal_cases' directory has a size of ~4 KB only.

#4.

No. of files:

```
{amostafa@bluehive:~/amostafa/legal_cases/group1}$ find *.* -type f | wc -l
677
```

There is a total of 677 files in /group1/ directory.

#5.

Largest file in group1 directory:

A. Using 'find':

```
{amostafa@bluehive:~/amostafa/legal_cases/group1}$ find . -type f -printf
"%p\t\t%\s\n" | sort -nr -k 2 | head -1
./06_1234.txt                    6477876
```

We can see that the file '06_1234.txt' is the largest in the group1/ directory, and it has a disk usage of 6477876 bytes or ~6.1778 megabytes (~6.2MB).

B. Using 'du':

```
{amostafa@bluehive:~/amostafa/legal_cases/group1}$ du -ah | sort -rh -k 1 | sed -n 2p
~6.2M          ./06_1234.txt
```

We can see that the file ‘06_1234.txt’ is the largest in the group1/ directory, and it has a disk usage of ~6.2 MB. And, we can get human-readable data from this command.

#6.

Characters in the smallest size in group2/ directory:

```
{amostafa@bluehive:~/amostafa/legal_cases/group2}$ find . -type f -printf
"%p\t%s\n" | sort -n -k 2 | head -1
./06_98.txt   267
```

```
{amostafa@bluehive:~/amostafa/legal_cases/group2}$ wc -m 06_98.txt
267              06_98.txt
```

So, the smallest file in group2/ directory, ‘06_98.txt’, has 267 characters.

#7.

No. of instances found of the string “United”:

```
{amostafa@bluehive:~/amostafa/legal_cases/}$ grep -r "United" ./ | wc -l
779
```

So, the “United” string is found 779 times inside all the subdirectories of legal_cases/.

No. of instances found of the case-insensitive string “united”:

```
{amostafa@bluehive:~/amostafa/legal_cases/}$ grep -ir "united" ./ | wc -l
888
```

So, the case-insensitive “united” or “United” or “UnItEd” string is found 888 times inside all the subdirectories of legal_cases/.

Caveat: A line can have two instances of “United” and if we consider a regular text file with Windows line endings where the line endings are marked as period (.), question mark (?), exclamation (!) [ASCII], we need to subtract multiple instances of “United”.

```
{amostafa@bluehive:~/amostafa/legal_cases/group1}$ grep -rn "United[^\.\?\!]*United"
./ | wc -l
126
```

So, if we subtract these 126 instances where two ‘United’ are present in the same line (for Windows-based systems), then only **779 - 126 = 653** lines have a “United” string.

For case-insensitive “united”, it becomes 129 double instances, hence there are **888 - 129 = 759** lines with case-insensitive “united” strings.

#8.

Bash script for changing file extension from .txt to .xml and vice versa:

```
#!/bin/bash/

#################################################################
##
## Afnan Mostafa
## Homework Assignment 3, problem 8, DSCC-401
## Last modified on: 09/28/2023 15:10 ET
## type in the terminal: bash chng_ext.sh [initial-filetype: either txt or xml (no dot necessary here)]
## ex: (for .txt to .xml) --> bash chng_ext.sh txt
## ex: (for .xml to .txt) --> bash chng_ext.sh xml
## future improvements: can be made more efficient by eliminating the two for loops inside the if
statement by introducing a new intermediate variable.
##
#################################################################

##============= directories =============##

cur_dir=$(pwd)
target_dir="/home/amostafa/amostafa/legal_cases"

##============= variables for file types =============##

var1="txt"
var2="xml"

##============= sanity check for right directory =============##

if [ "$cur_dir" != "$target_dir" ]; then
```

```
echo "this is not the legal_cases/ directory, changing pwd to legal_cases/"

cd "$target_dir"

echo "New directory: $(pwd)"

fi

##============= change extension from one to another =============##

echo "Task: changing extension";

#####-------- case 1: .txt to .xml --------#####

if  [ "$1" == "$var1" ]; then

echo “changing file extension from .txt to .xml, brace yourself”

for all_files in ./*/*.txt

do
      mv ${all_files} ${all_files%.txt}.xml
done

#####-------- case 2: .xml to .txt --------#####

elif [ "$1" == "$var2" ]; then

echo “changing file extension from .xml to .txt, brace yourself”

for all_files in ./*/*.xml

do
      mv ${all_files} ${all_files%.xml}.txt
done

fi
echo “Done...”
```

#9.

```
{amostafa@bluehive:~/amostafa/legal_cases/}$ . sample.sh
: No such file or directory
-bash: pi: command not found
-bash: =: command not found
two_pi
```

#10.

Error in line 1: No such file or directory

Reason: Bash is invoked incorrectly so it is not initialized, shebang is #!, and in the given script, it was written the other way (!#).

Corrected line 1: #!/bin/bash

Error in line 2: -bash: pi: command not found

Reason: White spaces—variable declaration cannot have any white space, i.e., there must not be any white space between the variable name and the assigned value.

Corrected line 2: pi=3.14159

#11.

The output will not be equal to the numerical value of 2*pi as bash can only do integer calculations by itself. To get an actual numerical floating point value, we need to use other commands or tricks to remove and put back the decimal point in its proper place.

#12.

Corrected script:

```
#!/bin/bash                        #← shebang fixed
pi=3.14159                         #← white spaces eliminated
two_pi=$(echo "2*$pi" | bc)        #← used bc command for float calc.
echo "$two_pi"                     #← fixed missing $ and “”
```

Output:

```
6.28318
```

#13.

A sample bash script titled ‘backupdata-AM.sh’ for backing up data with specific features that can help one identify and differentiate between different versions of backed-up files.

```
#!/bin/bash

####################################################################################
## Afnan Mostafa
## Homework Assignment 3, problem 13, DSCC-401
## Last modified on: 09/28/2023 18:58 ET
## type in the terminal: "bash backupdata-AM.sh" or ". backupdata-AM.sh"
## future improvements: can be made more efficient by using rsync or other combinations.
## function: backup files using cp and just ignoring the backup directory
## seed is to distinguish among multiple backup files in a single day
####################################################################################

########============ create backup directory ============#########

today=$(date +%d-%b-%Y)

backup="backup"

rand_seed=${RANDOM:0:2}

backup_dir="${backup}-${today}_seed${rand_seed}"

mkdir $backup_dir

########============ loop all items in pwd ============#########

for contents in *
do
   if [[ "$contents" != "$backup_dir" ]]; then
      cp -r "$contents" $backup_dir/
   fi
done

########============ archive and compress ============#########

tar -czf ${backup_dir}.tar.gz ${backup_dir}/

echo "Data backup completed at $(date +%H:%M:%S--%b-%d-%Y) with a random seed identifier of
${rand_seed}"
```

Note: I used Linux and a regular SSH client terminal for this homework assignment. Specifically, for problems no. 8 and 13, I modified the bash script using a simple Windows text editor through the SSH client. Hence, when I just copy the above bash script and paste it to a file using SSH client SFTP window, I see Windows carriage return (^M$) instead of Linux's ($) upon doing "cat -e backupdata-AM.sh". So, it results in an error when I do "bash backupdata-AM.sh". However, with a simple trick, I can remove the unwanted ^M symbols— sed -i "s/\r//" backupdata-AM.sh
So, if you see this script throwing errors, that is probably due to the carriage return mix-up.

Thank you.