Lab 2: Search and Report

Introduction

Lab 2 is an opportunity to use a select number of UNIX/Linux tools to search subdirectories, report the nature of files found and produce a summary report.

> There is a bit less hand-holding in this lab than in the previous lab. Refer back to the Lab 1 writeup if you have forgotten anything important.

Requirements

Your script shall be named `srpt` and be marked executable. Here is how you execute your script:

```
./srpt PATH
```

Example

`./srpt ./testfolder`

Your script shall inspect the file names of every file and directory in the folder represented by PATH and accumulate a count of each of the following file system objects:

- Every file
- Every subdirectory (excluding "PATH" itself)
- Every symbolic link
- Every file that is 365 days old or older (old files)
- Every file that is larger than 500,000 bytes (large files)
- Every graphics file (any file with suffix of .jpg, .gif or .bmp)
- Every temporary file (any file with a suffix of ".o")
- Every executable file (any file executable by the user)
- Total byte count of each file

Your `srpt` script shall produce a report in this format (note the exact spelling of the labels, the spacing and the commas in the numbers):

```
SearchReport host path date
Execution time nnn
Directories n,nnn,nnn
Files n,nnn,nnn
Sym links n,nnn,nnn
Old files n,nnn,nnn
Large files n,nnn,nnn
Graphics files n,nnn,nnn
Temporary files n,nnn,nnn
Executable files n,nnn,nnn
TotalFileSize n,nnn,nnn
```

Your `srpt` script should exit with return code 0 (zero) after producing its report unless the path was not specified on the commandline. If your script is executed without a PATH, issue this usage statement and exit with return code 1:

```
Usage: srpt PATH
```

Additional Requirements

You are free to use only the following Linux commandline programs: `find` (only once), `date`, `mkdir`, `rm`, `rmdir`, `wc`, `cut`, `awk`, `printf`, `echo` and `exit`. Using any other Linux programs may not be found on the Jenkins server. Please note the following:

- You may only make one pass over the files per execution of your script (meaning no multiple file system searches). The `find` command is the best utility to obtains all of the required information in a single execution of `find`. Search the Internet for examples on how to use `find` to search and count files of a given name or type and save that information in a temporary file.
- You may not modify any file or directory in the requested path. If you create any temporary files, do so **only in a unique folder you create**, in `/tmp`. Create a folder in `/tmp` with your icarus userID IN YOUR SCRIPT so that its name is unique.

  Example

  > `mkdir /tmp/$USER` at the top of your script will create a folder named `/tmp/dj45678` if your icarus userID happened to be `dj45678`. The `$USER` variable is always set to the current userID. Create **all** of your temporary files in this folder.

- At the end of your script, delete your `/tmp` folder with all of your temporary files. Be a good computer citizen, and do not expect that any files or folders in `/tmp` will still be there the next day. This is why it is called `/tmp`.
- Do not "prompt" the user to enter any information. Supply only the name of the folder on the commandline.
- Use self-documenting variables to hold the counts.

- Format counts with commas for readability. Look up the `printf` command, which can output numbers with commas.
- Output the date in the report header as produced by the `date` utility. Use the BASH variable `$HOSTNAME` to print the hostname in the report header. Here is an example output from date:

  ```
  Thu Dec 25 07:00:00 MST 2014
  ```

- Execution time is calculated from invocation and ends just before printing the summary report.
- Use whitespace and indenting to make your script readable.
- Add comments to your script to document your logic.

Clone your private repo on github.com

In the assignment module in Canvas, find the link to the Canvas page entitled "Clone your Lab 2 Repo", click on it and follow the instructions.

Use `git` to clone (download) a copy of your private repo to your local machine

To obtain a copy of your private repo to icarus, log into icarus and enter these commands:

```bash
git clone https://github.com/cowancs3030<SEMESTER>/lab2-YOURGITHUBUSERNAME

cd lab2-YOURGITHUBUSERNAME
```

Write and test `srpt`

Fire up your favorite text editor, and begin with your header:

```text
#!/bin/bash
# (Your name)
# Lab 2 - Search and Report
# CS 3030 - Scripting Languages

(Add your great code here)
```

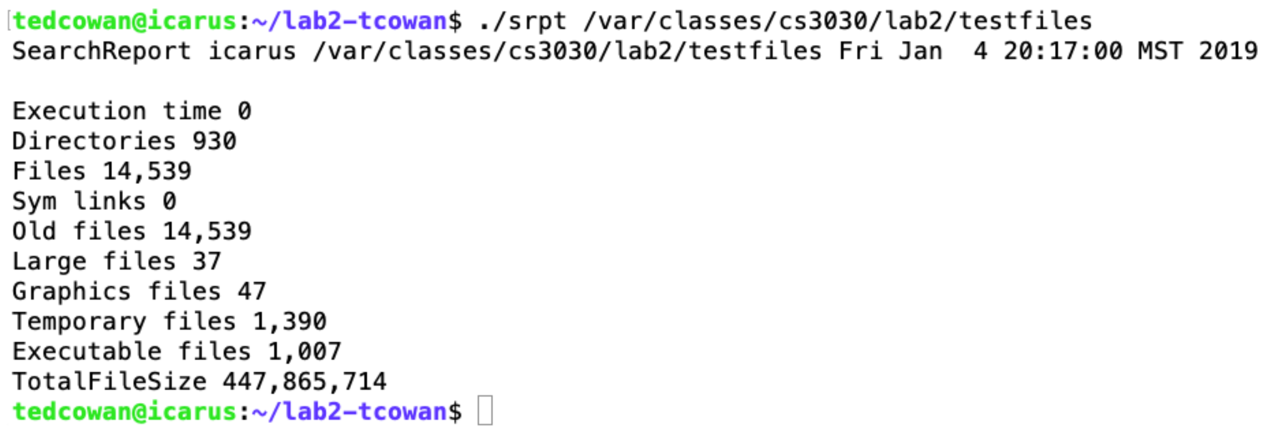
Be sure to save your script as `srpt`.

- As you add features, test your script with a folder of file system objects, provided on icarus for your convenience:

  ```
  ./srpt /var/classes/cs3030/lab2/testfiles
  ```

Your script must successfully complete scanning the `testfiles` folder in 5 seconds or less.

Here is an EXAMPLE of what your script might produce (the date of course will be different).



```
tedcowan@icarus:~/lab2-tcowan$ ./srpt /var/classes/cs3030/lab2/testfiles
SearchReport icarus /var/classes/cs3030/lab2/testfiles Fri Jan  4 20:17:00 MST 2019

Execution time 0
Directories 930
Files 14,539
Sym links 0
Old files 14,539
Large files 37
Graphics files 47
Temporary files 1,390
Executable files 1,007
TotalFileSize 447,865,714
tedcowan@icarus:~/lab2-tcowan$ 
```

Test your program using Cucumber

```
./cucumber -s
```

Cucumber for this lab generates a random set of file system objects to fully test your program.

Submit your assignment code for grading

Remember, you must push your script in your private repo to github.com to receive any points, for all assignments in this course.

```
git add srpt
git commit -m"COMMIT MESSAGE"
git push origin master
```

Files created for this lab: `srpt`

Grading

Here is how you earn points for this assignment:

FEATURES	POINTS
Must-Have Features	
Script is named correctly and found in its proper place in your repo	10
Script is executable	10
Required Features	
Report header is in the correct format	10
Execution time is reported in seconds	10
Execution time is < 5 seconds for `./testfiles`	40
Directory count is correct	10
File count is correct	10
Sym link count is correct	10
Old file count is correct	10
Large file count is correct	10
Graphics file count is correct	10
Temporary file count is correct	10
Executable file count is correct	10
Total file size is correct	10
Exit code is zero for normal execution	10
Exit code is 1 if the PATH is not specified on the commandline	10
A “Usage” statement is printed if the PATH is not specified on the commandine	10
Grand Total	200