Assignment 3: awk

CS3423 - Systems Programming

Sam Silvestro, Rocky Slavin, Steven O'Hara

Document version v1.2.1, April 10, 2020 11:35am

Introduction

For this assignment you will use **`awk`** to create a program for summarizing and printing information based on the directory listing data of files and information.

You are not to use any other programs, utilities, or scripting languages not covered in class, unless otherwise specifically and explicitly stated in this document.

Your program should take the output from the modified `ls` command line seen below, and process the data in order to output the aggregate information:

```
ls -la --time-style='+%Y-%m-%d %H:%M:%S'
```

In fact, to avoid human error and ensure you are always using the correct command line, I suggest creating and adding a new alias to your `bash` resource configuration file:

```
alias lsa="\ls -la --time-style='+%Y-%m-%d %H:%M:%S'"
```

Note that the inclusion of the leading backslash ensures no other previously-defined/existing `ls` aliases are used; certain other options such as `-h` could cause your script to fail, for example.

Aggregated information requirements

The aggregated information processed from the directory listing data should consist of the following (see example later for proper output formatting):

- Per-user grouping of file-related counts found in specified directories
  - Username of the entity owning these files
  - Total number of files found owned by this user, printing two values: all files versus hidden files
  - Total number of directories found that are owned by this user
  - Total number of "other" files found that are owned by this user
    (these items include, but are not limited to, symbolic links, FIFO's, character or block devices, etc. Basically, anything that is not a regular file nor a directory will fall under this category)
  - Total file storage (in bytes) occupied by the user's regular files.
- Itemization of the oldest and newest **regular files** found *(if no regular files exist in the listing, simply report "None" for these items. If only one regular file exists, it is reasonable to report this file as both the oldest and newest.)*

Also note, if multiple files share the same oldest or newest timestamps, you can break the tie however you wish; there are no guidelines you must adhere to while doing so.

- Total file-related counts found in the specified directories
  - Total users owning files within these paths
  - Total number of files found, printing two values: all files versus hidden files
  - Total number of directories found
  - Total number of "other" files found
    (these items include, but are not limited to, symbolic links, FIFO's, character or block devices, etc. Basically, anything that is not a regular file nor a directory will fall under this category)
  - Total file storage (in bytes) occupied by all regular files listed.

Note: again, **do not** use `sed` , Python, or any other languages or utilities not explicitly allowed by this assignment.

Note 2: ensure to test the processing of `ls` listings for multiple directories, rather than just one. Such listings can be generated by passing more than one directory to `ls` and/or by the simple addition of the `-r` recursive option to the custom `ls` command shown previously. Two examples of such command lines can be seen here:

```
ls -la --time-style='+%Y-%m-%d %H:%M:%S' dir1 dir2 dir3
ls -lar --time-style='+%Y-%m-%d %H:%M:%S' dir1
```

or if you have defined the aforementioned alias, equivalently:

```
lsa dir1 dir2 dir3 file1 dir4
lsa -r dir1 file1 dir2
```

Note that these commands can also include filenames alongside the directory names on the command line as well; this is perfectly permissible and should be accounted for, hence why it was shown in the example above.

Example

The example below is an excerpt from the following command, executed upon my home directory:

```
ls -la -time-style='+%Y-%m-%d %H:%M:%S'  ~
```

Input

```
ssilvestro@fox05:~/courses/cs/3423/Spring20/assign3$ head -n 30 data/input.txt
total 17160
drwxrwxrwt 98 root   root      528384 2020-04-07 13:38:14 .
drwxr-xr-x 26 root   root        4096 2018-09-04 10:50:29 ..
drwx------  2 pmp099 students    4096 2020-03-03 20:57:31 appInsights-nodeAIF-444c3af9↩
    -8e69-4462-ab49-4191e6ad1916
```

```
-rw-------  1 mce237 students     199 2020-03-01 18:41:59 .build1276786824731864129.log
-rw-------  1 mce237 students     199 2020-03-01 20:18:42 .build291177188595028335.log
-rw-------  1 mce237 students     199 2020-03-01 20:10:44 .build4195866878600813549.log
-rw-------  1 mce237 students     199 2020-03-01 20:08:55 .build4503681510908034369.log
-rw-------  1 mce237 students     199 2020-03-01 18:18:44 .build4964061885086964943.log
-rw-------  1 mce237 students     199 2020-03-01 20:17:13 .build5474334865226720725.log
-rw-------  1 mce237 students     199 2020-03-01 19:08:39 .build6322670020019345604.log
-rw-------  1 mce237 students     420 2020-03-01 20:08:08 .build8057453026527719771.log
-rw-------  1 mce237 students     199 2020-03-01 20:08:32 .build8316126450060215695.log
-rw-------  1 mce237 students     732 2020-03-01 20:13:35 .build8317708361921336382.log
-rw-------  1 mce237 students     420 2020-03-01 20:07:57 .build8983757940366444429.log
drwxr-xr-x  3 bfn715 students    4096 2020-03-03 23:07:12 dlight_bfn715
drwx------  3 dad980 students    4096 2020-03-05 15:44:15 dlight_dad980
drwx------  3 hrb980 students    4096 2020-04-06 09:54:44 dlight_hrb980
drwx------  3 hrm102 students    4096 2020-04-06 18:43:17 dlight_hrm102
drwx------  3 kaq447 students    4096 2020-02-26 17:58:46 dlight_kaq447
drwx------  3 mce237 students    4096 2020-03-30 00:04:57 dlight_mce237
drwx------  3 mjy610 students    4096 2020-02-27 15:33:54 dlight_mjy610
drwx------  3 pdq039 students    4096 2020-04-06 18:43:48 dlight_pdq039
drwx------  3 xie192 students    4096 2020-03-23 17:47:37 dlight_xie192
drwx------  3 ynb963 students    4096 2020-04-07 13:26:46 dlight_ynb963
-rw-------  1 hrb980 students      95 2020-03-09 16:25:53 exec1108000877022604592.log
-rw-------  1 hrb980 students      74 2020-04-03 13:39:09 exec1218509371493740144.log
-rw-------  1 hrb980 students    1470 2020-03-09 13:28:36 exec1334040267987479302.log
-rw-------  1 hrb980 students    1134 2020-04-06 10:16:23 exec1413924165655873346.log
-rw-------  1 mce237 students    1538 2020-03-01 18:17:50 exec1520228248140431728.log
...
...
```

Output

```
user: mjy610
   dirs: 3

user: hrb980
   files:
     all/hidden: ( 195 / 2 )
   dirs: 3
   file storage: 76235 B

user: pdq039
   dirs: 3

user: zqu051
   files:
     all/hidden: ( 452 / 0 )
   file storage: 652583 B

user: mce237
   files:
     all/hidden: ( 52 / 12 )
   dirs: 4
   file storage: 2729344 B
```

```
user: dad980
   files:
     all/hidden: ( 4 / 1 )
   dirs: 3
   file storage: 6614 B

user: pmp099
   dirs: 2
   other: 10

user: ynb963
   files:
     all/hidden: ( 2 / 0 )
   dirs: 3
   file storage: 4202 B

user: xie192
   dirs: 3

user: kaq447
   files:
     all/hidden: ( 2 / 0 )
   dirs: 3
   file storage: 3092 B

user: bfn715
   dirs: 3

user: root
   files:
     all/hidden: ( 1 / 1 )
   dirs: 5
   other: 1
   file storage: 11 B

user: hrm102
   dirs: 3

oldest file:
    -r--r--r--  1 root   root          11 2020-02-25 15:30:11 .↩
       X0-lock
newest file:
    -rw-------  1 ynb963 students    1308 2020-04-06 19:40:46 ↩
       output1586220046526
```

```
total users:    13
total files
    all/hidden: ( 708 / 16 )
total dirs:     38
total other:    11
file storage:   3472081 B
```

Extra Credit (15%)

A 15% bonus will be awarded for those whose script correctly and properly sorts the username-grouped portion of the output. Such sorted output for the above example can be seen here:

Extra Credit Output

```
user: bfn715
   dirs: 3

user: dad980
   files:
     all/hidden: ( 4 / 1 )
   dirs: 3
   file storage: 6614 B

user: hrb980
   files:
     all/hidden: ( 195 / 2 )
   dirs: 3
   file storage: 76235 B

user: hrm102
   dirs: 3

user: kaq447
   files:
     all/hidden: ( 2 / 0 )
   dirs: 3
   file storage: 3092 B

user: mce237
   files:
     all/hidden: ( 52 / 12 )
   dirs: 4
   file storage: 2729344 B
```

```
user: mjy610
   dirs: 3

user: pdq039
   dirs: 3

user: pmp099
   dirs: 2
   other: 10

user: root
   files:
     all/hidden: ( 1 / 1 )
   dirs: 5
   other: 1
   file storage: 11 B

user: xie192
   dirs: 3

user: ynb963
   files:
     all/hidden: ( 2 / 0 )
   dirs: 3
   file storage: 4202 B

user: zqu051
   files:
     all/hidden: ( 452 / 0 )
   file storage: 652583 B

oldest file:
    -r--r--r--  1 root   root          11 2020-02-25 15:30:11 .↩
       X0-lock
newest file:
    -rw-------  1 ynb963 students    1308 2020-04-06 19:40:46 ↩
       output1586220046526

total users:    13
total files
    all/hidden: ( 708 / 16 )
total dirs:     38
total other:    11
file storage:   3472081 B
```

Hint: research awk's `asort` function for help.

Script Execution

Your program should each be invoked through a single bash file (see below) with input taken from `stdin`. The resulting output should be printed directly to `stdout`.

Assignment Data

A few sample input files can be found at the following location on the `fox` servers, however it is imperative that you fabricate many of your own examples to ensure that your script functions *according to the specifications outlined above*:
`/usr/local/courses/ssilvestro/cs3423/Spring20/assign3`.

Script Files

Your submission should consist of exactly two files:

- **`assign3.sh`** - a bash script used as the driver program for your `awk` script
- **`assign3.awk`** - the `awk` program used in `assign3.sh`

Verifying Your Program

In addition to the above Assignment Data, your program should also work with arbitrary input from the `ls -la --time-style='+%Y-%m-%d %H:%M:%S'` command defined on page 1. This include both reading from one or more input files, as well as accepted piped input directly from standard input, as in these examples:

```
ls -la --time-style='+%Y-%m-%d %H:%M:%S' ~ | ./assign3.sh
```

– or –

```
./assign3.sh listing.txt [listing2.txt [...]]
```

Submission

Turn your assignment in via Blackboard. Your zip file, named **`abc123.zip`** with your personal abc123 should contain only your two `bash` and `awk` files.

If you attempt the extra credit, name your file **`abc123_EC.zip`**. Without the **`_EC`**, your submission will be graded as normal.