

Bourne Shell Script Newbie Help

Source: <http://unix.derkeiler.com/Newsgroups/comp.unix.shell/2004-01/0438.html>

From: Daz (htpw16_at_hotmail.com)

Date: 01/14/04

Date: 14 Jan 2004 11:24:51 -0800

Hello Group, I need some assistance in starting an assignment. I have never programmed in Unix let alone shell scripting. I searched the internet to find similar problems to mine when I came accross this. So could you aid me in starting this problem. Thank you. Heres the problem:

Overview

In this assignment you will write a Bourne shell script to calculate averages and medians from an input file of numbers. This is the sort of calculation I might do when figuring out the grades for this course.

The input file will have whole number values separated by tabs, and each line

of this file will have the same number of values. (For example, each row

might be the scores of a student on assignments.) Your script should be able to calculate the average and median across the rows (like I might

do to calculate an individual student's course grade) or down the columns (like I might do to find the average score on an assignment).

You should read the paper "Introduction to the Unix Shell" by Bourne and the man pages on sh, read, expr, cut, head, tail, wc, and sort before you begin this assignment. These are the utilities that you will need to write this program.

Your script will be called stats. The general format of the stats command is

```
stats {-rows|-cols} [input_file]
```

Note that when things are in curly braces separated by a vertical bar it means you should choose one of the things; here for example, you must choose either `-rows` or `-cols`.

The option `-rows` calculates the average and median across the rows; the option `-cols` calculates the average and median down the columns. When things are in square braces it means they are

optional; you can include them or not, as you choose. If you specify an input_file the data is read from that file; otherwise, it is read from standard input.

Here is a sample run.

```
Script started on Wed Sep 30 12:07:50 1998
```

```
% cat test_file
1 1 1 1 1
9 3 4 5 5
6 7 8 9 7
3 6 8 9 1
3 4 2 1 4
6 4 4 7 7
% stats -rows test_file
Average Median
1 1
5 5
7 7
5 6
3 3
6 6
% cat test_file | stats -c
Averages:
5 4 5 5 4
Medians:
6 4 4 7 5
% echo $?
0
% stats
Usage: stats {-rows|-cols} [file]
% stats -r test_file nya-nya-nya
Usage: stats {-rows|-cols} [file]
% stats -both test_file
Usage: stats {-rows|-cols} [file]
% chmod -r test_file
% stats -columns test_file
stats: cannot read test_file
% stats -columns no_such_file
stats: cannot read no_such_file
% echo $?
1
% exit
Specifications
```

You must check for the right number and format of arguments to stats. You should allow users to abbreviate -rows and -cols; any word beginning with an r is taken to be rows and any word beginning with a c is taken to be cols. So, for example, you would get averages and medians across the rows with -r,

–rowwise and –rumpelstiltskin. If the command has too many or two few arguments or if the arguments of the wrong format you should output an error message to standard error. You should also output an error message to standard error if the input file is specified, but it is not readable.

You should output the statistics to standard output in the format shown above. Be sure all error messages are sent to standard error and the statistics are sent to standard output. If there is any error, the exit value should be 1; if the stats program runs successfully the exit value should be 0.

Your stats program should be able to handle files with any reasonable number of rows or columns. You can assume that each row will be less than 1000 bytes long (because Unix utilities assume that input lines will not be too long), but don't make any assumptions about the number of rows. Think about where in your program the size of the input file matters. You can assume that all rows will have the same number of values; you do not have to do any error checking on this.

You will probably need to use temporary files. For this assignment, the temporary files should be put in the current working directory. (A more standard place for temporary files is in /tmp but don't do that for this assignment; it makes grading easier if they are in the current directory.) Be sure the temporary file uses the process id as part of its name, so that there will not be conflicts if the stats program is running more than once. Be sure you remove any temporary files when your stats program is done. You should also use the trap command to catch interrupt, hangup, and terminate signals to remove the temporary files if the stats program is terminated unexpectedly.

All values and results are whole numbers. You will use the expr command to do your calculations; it only works with whole numbers. When you calculate the average you should round to nearest whole number. (How do you do that with only whole number operations?)

To calculate the median, sort the values and take the middle value. For example, the median of 97, 90, and 83 is 90. The median of 97, 90, 83, and 54 is still 90—when there are an even number of values, choose the larger of the two middle values.