

CS306 – Introduction to Perl
Summer 2005
Homework Assignment #3
Due: June 22nd, 2005, 5pm

The homework should be submitted in the form of a zip file which contains one program for each question. Your zip file should be named *firstname-lastname-hw3.zip* and should contain files like hw3q1, hw3q2, hw3q3, etc... Email this zip file to cs306@cis.uab.edu.

1. (10 points) Write a grocery store program. This program should present a menu of items which the user can buy. The user should indicate which item they want to add to their cart. The item should be added and the user allowed to make another selection. Allow the user to add as many items as they wish. When the user is done shopping, they will choose the “checkout” option from the menu. This should present the user with their itemized receipt and total cost.

Sample output:

```
Welcome to the General Store
1. Steak           6.99
2. Cereal          2.79
3. Eggs            0.99
4. Milk            3.09
.....
10. Checkout
Enter your choice (1-10):      (loop here until checkout)
Your Receipt
Steak    2@6.99      13.98
Cereal   1@2.79      2.79
Milk     2@3.09      6.18
TOTAL..... 22.95
Thank you for shopping with us!
```

Organize your code into subroutines, use formatted output for neatness (aligning the prices on the . point, etc...), avoid global variables, comment your code, and make good use of hashes where appropriate.

SEE NEXT PAGE FOR QUESTION #2

2. (10 points) Write a program called “wcount” (short for word count) that takes as input one or more text files, counts how many times each word appears, and produces a report of each word and its count. (This is essentially an extension of HW2 Q4). The text file(s) and/or STDIN will contain only words, spaces and newlines, no punctuation. Your program must meet the following requirements:

1. It should be able to take as input STDIN, one or more files, or a directory name.
 - a) if provided a directory name, it should open a directory handle and process each file in that directory, excluding the . and .. files.
2. It should be case-insensitive. To and to should be treated as the same word.
3. It should produce output on STDOUT which has one word per line, with its word count, highest first. The words in the report should be aligned (see sample below).
4. It should also produce a file called wcount.log that contains one line per each run of the program, with some basic summary information (see example below).
5. If you should encounter a blank line (see the length(\$string) function for lines of length 0 -after- chomp) it should print a warning on STDERR.

Sample STDOUT output:

```
172 the
 96 to
 82 a
...
 2 alabama
 1 widget
```

Sample wcount.log output:

```
Found 2657 total words, 1502 distinct words.
Found 722 total words, 523 distinct words.
...
```

The program will be called with zero or more arguments like this:

```
wcount
wcount filename/directoryname
wcount file1 file2 file3
```

If called with one argument, it could be either a filename or a directory name. If called with more than one, you may assume they are all file names.

Hint: Since you are dealing with multiple input options, it might make sense to gather all of the input first before attempting to analyze the contents.

Hint #2: `foreach $key (sort {$hash{$a} <=> $hash{$b}} keys %hash) { # sort by hash values`
`$a` and `$b` are special variables, don't change them. The `%hash` is whatever you name yours.

A sample input file and directory will be available for download on the class website.