CSCI 1321 (Principles of Algorithm Design II), Fall 2001

Homework 1[1]

Assigned: September 4, 2001.

Due: September 11, 2001, at 5pm.

Credit: 20 points.

> *Note:* The HTML version of this document may contain hyperlinks. In this version, hyperlinks are represented by showing both the link text, formatted like this, and the full URL as a footnote.

Contents

1 Overview **1**

2 Details **2**

3 What to turn in **3**

1 Overview

Write a program to compute the correct state sales tax, given a list of item prices. For example, a retailer selling two items each costing $3.45 and one item costing $123.45 could compute the sales tax by typing

```
salesTax salesTaxRates TX 3.45 3.45 123.45
```

into any shell. Here,

- `salesTax` is the name of the executable program.
- `salesTaxRates` is the name of a file with each line containing a two-letter state abbreviation[2], e.g., TX and CA, and the state sales tax rate written as a percentage, e.g., 6.25 for Texas and 7.25 for California. Sales tax rates are stored in a file because they continually change.
- `TX` indicates the state for which the sales tax should be computed.
- `3.45 3.45 123.45` indicates the item prices, written as decimal numbers with two digits after the decimal point.

The program should print 8.15, which is the sales tax due. That is,

```
(3.45 + 3.45 + 123.45) * 6.25% = 8.146875,
```

which is then rounded to the closest penny.

[1] © 2001 Jeffrey D. Oldham (`oldham@cs.stanford.edu`) and Berna L. Massingill (`bmassing@cs.trinity.edu`). All rights reserved. This document may not be redistributed in any form without the express permission of at least one of the authors.

[2] `http://www.usps.gov/ncsc/lookups/usps_abbreviations.htm`

2 Details

- Use command-line arguments. Command-line arguments are yet another way for programs to obtain input from the user. Rather than prompting the user for user for necessary input such as filenames, any input known when the program begins can be sent to the program using command-line arguments.

  The example given above has six command-line arguments, separated by whitespace. Just as for arrays, they are numbered starting with zero. For example, `argv[2]` is `TX`, and `argv[5]` is `123.45`.

  Here are the three rules for using command-line arguments:

  1. The `main` function should have two parameters: `int main(int argc, char *argv[])`
  2. To access the i-th argument, use `argv[i]`.
  3. `argv[argc]` equals 0.

  The first parameter `argc` indicates the number of command-line arguments. The second parameter can be interpreted as an array (`[]`) of C-style strings, i.e., strings ending with a null-termination character. For example, `argv[0]` is a C-style string listing the executable program's name — "salesTax" in the example given above.

  For an example of a program that uses command-line arguments, see sample program `addargs.cpp`[3].

  All command-line arguments are C-style strings, even for numeric arguments; e.g., `argv[3]` is the string "3.45". You can use the `atof()` library function to convert from a string to a `double`. Under Linux/Unix, you can get online help about this function by typing `man atof` or `info atof`. For example, use

  ```
  atof(argv[3])
  ```

  to convert the string to its double value 3.45. Be sure to include `stdlib.h`.

- Round fractions of pennies to the nearest penny. Round half-pennies up. For example, 10.5 cents is rounded to 11 cents, but 10.49999 cents is rounded down to 10 cents.

- Be sure your program works for any number of item prices and for a file of sales tax rates of any length. Be sure your program prints appropriate error messages in the following cases:

  – not enough command-line arguments
  – incorrect filename (i.e., no such file)
  – desired state not found in sales tax file

  Your program can assume that all the item prices are numeric. (Ideally the program would check for this, but doing so is somewhat beyond the scope of this assignment. If you want to try adding this functionality, try using the `strtod()` library function.)

- Remember to include comments in your program! At a minimum, include header comments giving your name and describing what the program does.

[3] `http://www.cs.trinity.edu/~bmassing/CS1321_2001fall/SamplePrograms/addargs.cpp`

3 What to turn in

Submit your source code as described in the Guidelines for Programming Assignments[4]. Use a subject heading of "cs1321 hw1", and submit a single file containing your C++ source code.

[4] `http://www.cs.trinity.edu/~bmassing/CS1321_2001fall/Notes/pgmguidelines/index.html`