

Functions:**Billing for VT Long Distance**

This project has the same basic input and output as Project 3. The main difference is to change the code to conform to the requirements that are specified below. The Input, Calculations, and Output sections are here for your convenience. There has been no change to these three sections since Project 3. Be sure to read the Evaluation section carefully to see what the specific requirements of your program are.

Sample input data:

Here is a sample input file, named `CallData.txt`, for the program. The first line specifies the name of the customer, and the second specifies the customer's phone number, which is also used as the account number. The third line specifies the billing date. Each of the next three lines specifies the number of minutes of calls that were made this month in each of three categories. The significance of the categories is explained in the next section.

Each data value is preceded by a descriptive label that ends with a colon character ': '.

Name:	John McBryde
Phone:	540.231.2346
Date:	02/24/2003
Peak time:	7:50
Off peak time:	0:40
Calling card:	8:16

The input values are guaranteed to be syntactically and logically correct. The name, phone number and date are just string data. For formatting purposes, you may assume that the name will be no more than 40 characters long, and the phone number and date will be no longer than the ones shown above. The time values will be expressed in HH:MM form (as shown above).

Calculations:

Under the calling plan selected by all the customers this project will deal with, VT Long Distance has three categories for long-distance calls:

Category	Charge per minute
Peak time	0.10
Off-peak time	0.05
Calling card	0.25

The rates are given in dollars. In addition to these per-minute charges, VT Long Distance charges customers on this plan a flat monthly fee of \$4.95 unless they make no peak time or off-peak time calls, in which case the flat fee is reduced to \$1.00. On top of that, if the customer made any calling card calls, there is a flat fee of \$1.00.

In addition to these charges, VT Long Distance also must collect various taxes and statutory user fees. Each customer, no matter how many or how few minutes they have used, must pay a Universal Subscriber Fee of \$5.31. Each customer must pay a local tax of \$1.25.

Each customer must also pay a state tax of 0.4% on the total amount of VT Long Distance charges (but not including the USF or the local tax). Mathematically the state tax will usually not be an exact number of cents; the result must be truncated to an integer number of cents, so a tax of 0.147 would be truncated to 0.14. (If you follow the advice on handling monetary amounts given later, this will happen automatically.)

Given the call data in the input file, you must calculate an itemized bill for the customer. The required format is shown in the sample output file given next.

Sample output:

Here is a sample output file, named `PhoneBill.txt`, which corresponds to the input data given above:

```

Programmer: Rich Wheaton
CS 1044 Project 5 Fall 2003

Customer:           John McBryde
Account:            540.231.2346
Billing date:      02/24/2003

=====
                Minutes      Charge
=====
Peak:              7:50       47.00
Offpeak:           0:40        2.00
Card:              8:16       124.00
=====
Subtotal                          173.00
Calling plan fee                    4.95
Calling card fee                     1.00
Universal connectivity charge        5.31
Local tax                            1.25
State tax                            0.71
=====
Total                              186.22

```

As usual the first three lines just identify the programmer and project. The next three lines specify the customer, account and billing date. Next there is a blank line, and then the headers for the table with the bill amounts. The minutes and total charge for each call category are reported; there will be a line for each category even if there were no call minutes.

Next there is a subtotal for the per-minute charges, then the VT Long Distance fees that were assessed. There will always be a calling plan fee. If there were no calling card calls then the line for the calling card fee is omitted.

Next the USF and local tax are reported, and then the state tax amount. (The state tax cannot be zero, so of course there will always be an entry for it.) Finally the grand total is reported.

If you have read the description of how the Curator scores your program in the *Student Guide*, you know that is important that you use the same spelling and capitalization for all the labels shown above. The horizontal spacing does not effect scoring unless you combine things that should be separate or separate things that should be combined. You are free to experiment with the horizontal layout but you should try to align the columns neatly.

Note that you must insert blank lines and divider lines as shown.

Evaluation:

Everything that was said in the specification for Project 3 about monetary values and the Curator still applies here.

Read the *Programming Standards* page on the CS 1044 website for general guidelines.

Specifically, your program must contain at least 4 user-defined functions.

- At least one of these functions must be an input function.
- At least one of these functions must be an output function.
- At least two functions must be neither input nor output functions. Of these two, at least one function must have a return code. This must be a return code of meaning (e.g. the function cannot always return the same value).

Your implementation must also meet the following requirements:

- You should have a header comment identifying yourself, and describing what the program does.

- Don't forgot the Honor Code pledge.
- Every constant and variable you declare should have a comment explaining its logical significance in the program.
- Every major block of code should have a comment describing its purpose.
- Each function in the program must also have a header comment explaining its purpose, commenting the parameters and explaining the return code (as described in lecture)
- Adopt a consistent indentation style and stick to it for readability purposes.
- No global variables!
- Do not use any advanced data structure, such as arrays or structures.
- Choose descriptive identifiers when you declare a variable or constant. Avoid choosing identifiers that are entirely lower-case.
- Use named constants instead of literal constants when the constant has logical significance.
- Use C++ streams for input and output, not C-style constructs.
- Use C++ string variables to hold character data, not C-style character pointers or arrays.

Understand that the list of requirements here is not a complete repetition of the *Programming Standards* page on the course website. It is possible that requirements listed there will be applied, even if they are not listed here.

Submitting your program:

You will submit this assignment to the Curator System (read the *Student Guide*), and it will be graded automatically. Instructions for submitting, and a description of how the grading is done, are contained in the *Student Guide*.

You will be allowed up to five submissions for this assignment. Use them wisely. Test your program thoroughly before submitting it. Make sure that your program produces correct results for every sample input file posted on the course website. If you do not get a perfect score, analyze the problem carefully and test your fix with the input file returned as part of the Curator e-mail message, before submitting again. The highest score you achieve will be counted.

The *Student Guide* and submission link can be found at:

<http://www.cs.vt.edu/curator/>

Pledge:

Each of your program submissions must be pledged to conform to the Honor Code requirements for this course. Specifically, you **must** include the following pledge statement in the header comment for your program:

```
// On my honor:
//
// - I have not discussed the C++ language code in my program with
// anyone other than my instructor or the teaching assistants
// assigned to this course.
//
// - I have not used C++ language code obtained from another student,
// or any other unauthorized source, either modified or unmodified.
//
// - If any C++ language code or documentation used in my program
// was obtained from another source, such as a text book or course
// notes, that has been clearly noted with a proper citation in
// the comments of my program.
//
// - I have not designed this program in such a way as to defeat or
// interfere with the normal operation of the Curator System.
//
// <Student Name>
```

Failure to include this pledge in a submission is a violation of the Honor Code.