Assignment 1: First C Programs

Weight: 5%

Due: Friday, February 6, 2009

Assignment Goals

The lectures so far have focused on the C programming language. We have covered conditional statements, while loops, data types, arrays and strings. These are the basic components of imperative programming that you learned in CPSC 217 using the python language. The goal of this assignment is to help you to become proficient in using these concepts in the C programming language.

Part 1 (50%): Computing Basic Statistics for a List of Numbers

Problem

We need a program that will tell us some basic statistics about a list of non-zero floating-point numbers. Specifically, we want to know the mean (or average), the minimum value, the maximum value, and the standard deviation.

The standard deviation should be computed using the following formula:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1}(x_i - \mu)^2}$$

Where,

$\sigma =$ standard deviation
$\mu =$ mean
$x_0, x_2, \ldots, x_{N-1}$ are the *N* numbers entered on the command line

The purpose of this assignment is to learn to program in C, not to understand mathematical equations. Feel free to ask the instructor or the TA for help in understanding the meaning of the above equation.

Specification

Input

Your program must take a list of non-zero floating-point numbers as arguments on the command line. If no arguments are provided, your program must print out a message that states how to use the program and then terminate. If any of the numbers are zero or not numbers at all, an error message must be printed and the program must terminate. Here are some examples,

```
> ./statistics
Usage: ./statistics <list-of-numbers>
```

```
> ./statistics 0 1 4
‘0’ is not a valid number
> ./statistics three 5 hello
‘three’ is not a valid number
```

Output

Your program must print out the correct mean, minimum, maximum, and standard deviation values for the list of numbers. For example,

```
> ./statistics 3.4 1.9 10.3 1.1
Mean: 4.18
Min: 1.10
Max: 10.30
Standard Deviation: 3.63
> ./statistics 4 5
Mean: 4.50
Min: 4.00
Max: 5.00
Standard Deviation: 0.50
> ./statistics -3.5 100.345 5
Mean: 33.95
Min: -3.50
Max: 100.35
Standard Deviation: 47.08
```

Demonstration of Part 1

You will be required to demonstrate to your TA that your program accurately performs the computations outlined above. You will not be given the list of test numbers until demo time.

Hints

In order to complete this part of the assignment, you will need to know how to get input from the command line arguments of a program. This is accomplished by adding arguments to your main function as follows:

```
int main(int argc, char **argv)
{
…
}
```

`argc` will store the number of arguments and `argv` is the array containing the arguments (the first of which will be the name of the program that is executed). The second parameter could be rewritten as `char *argv[]`.

You will also need to be able to translate strings into floating-point numbers. I recommend that you use the function `atoff` or `atof`. These functions convert a character array into a `float` or `double`, respectively.

Part 2 (50%): Big Mac Carbon Footprint

Problem

We need a program that will provide information about the carbon footprint of Big Macs. The dataset to be used is the following:

Data collected from ManyEyes (http://manyeyes.alphaworks.ibm.com/manyeyes/).

Production Stage	Energy Cost ($)	CO2/Methane emissions (pounds)
Crop/feed production	0.27	1.5
Cow burping/flatulence	0.0	0.07
Transport	0.02	0.13
Milling	0.01	0.15
Baking	0.03	0.37
Milking/making cheese	0.01	0.12
Slaughtering/cutting	0.04	0.52
Grinding/freezing	0.005	0.06
Freeze-drying	0.002	0.03
Pickling	0.001	0.01
Frying	0.03	0.37
Storage	0.12	1.5

Specification

Your program must accept search criteria via a menu system. You should be able to search for all production stages which match:

a) a string of characters within the name of the production stage;
b) a range of energy costs (from a specified minimum value to a specified maximum value); or
c) a range of carbon emissions (from a specified minimum value to a specified maximum value).

For example (user input is highlighted in bold):

```
> ./bigmac
Search on:
1. Production Stage
2. Energy Cost
3. Emissions
4. Exit

Enter your choice: 1
Stage contains: reez

Production Stage         Energy Cost    Emissions
Grinding/freezing        0.00           0.06
Freeze-drying            0.00           0.03
```

```
Search on:
1. Production Stage
2. Energy Cost
3. Emissions
4. Exit

Enter your choice: 2
Enter minimum value: .1
Enter maximum value: .5

Production Stage         Energy Cost    Emissions
Crop/feed production     0.27           1.50
Storage                  0.12           1.50

Search on:
1. Production Stage
2. Energy Cost
3. Emissions
4. Exit

Enter your choice: 3
Enter minimum value: .2
Enter maximum value: .9

Production Stage         Energy Cost    Emissions
Baking                   0.03           0.37
Slaughtering/cutting     0.04           0.52
Frying                   0.03           0.37

Search on:
1. Production Stage
2. Energy Cost
3. Emissions
4. Exit

Enter your choice: 4
>
```

Criteria

- The comparison performed on the production stage must be case sensitive. This constraint is being added to simplify the assignment.
- A title line must be printed before displaying the matching rows (as seen in the examples). All of the columns within the output must line up appropriately.
- In the menu, if a number other than 1-4 is entered, an error message must be printed.
- No other error-checking is required.
- Once a search has been completed, the menu must be reprinted. The only way to end the program is to enter the number ‘4’ as your choice in the main menu.

Hint

A string function which would help the completion of this assignment is called `strstr`. This function searches for the occurrence of one string within another string.

Demonstration of Part 2

Your TA will execute your program several times to ensure that it is behaving as specified above.

Python Solution to a Similar Assignment

A code listing for a similar assignment in a previous 217 class is posted on the website. This code searches through a feline dataset and does not match the specification for this assignment. It is provided as a source of hints as to how you might solve part 2 of this assignment.

Documentation

For your program, you should minimally document:

- your name;
- the purpose of the program;
- the date you started writing the program;
- any global variable use; and
- any use of structures.

For each function, you should minimally document:

- the purpose of the function;
- the input parameters;
- the output/return values; and
- the algorithm used if it is not obvious from the code.

Evaluation

Your mark for each part will be calculated as follows:

	Excellent	Satisfactory	Unsatisfactory
Documentation	(15 marks) Your documentation is effective, concise and includes all of the components listed above.	(10-14 marks) Your documentation is missing one or two of the components above, you are overly verbose in a few places, or it is difficult to understand what you have written for one or two descriptions.	(0-9 marks) Most of your documentation is missing the above components; your comments are extremely long and usually difficult to understand.
Programs output correct values	(15 marks) Your program produces correct output for every possible input (according to the specifications).	(10-14 marks) Your program produces mostly correct output, with the exception of up to four types of input.	(0-9 marks) Your program produces mostly incorrect output. The range of 0-9 will depend on how close the output is to being correct.
Program structure	(15-20 marks) It is very easy to follow the flow of your program and it is clear why each step is performed. You use several functions to avoid repeating code. Each function is a small number of lines of code and represents a reusable bit of code.	(10-14 marks) The TA has some difficulty understanding the flow of your program, but is able to eventually figure it out. Some of your functions are long and could be broken down into two or more functions.	(0-9 marks) The TA has a very difficult time understanding the flow of your program (or cannot at all understand it). Most of your functions are too long and could be broken down into two or more.
Variable names	(15 marks) Every variable has a name that makes your code clear and easy to read.	(10-14 marks) Up to six variables have names that aren't clear (e.g., x, foo, bar).	(0-9 marks) More than six variables have names that aren't clear.
Demonstration	(15 marks) Your program works exactly according to the specification for all test cases. You are also able to clearly explain your code and answer questions about the effect of changing it.	(10—14 marks) Your program works according to the specification, with the exception of up to three test cases. You have some difficulty explaining your code or require some prompting from the TA to be able to describe the effect of changes to your code.	(0-9 marks) Your program produces incorrect output for most of the test cases. You cannot explain your code or cannot answer questions about the effect of changing it.
Analysis and design	(15-20 marks) Your choice of variables and statements make your solution clear.	(10-14 marks) In up to six places, you choose a variable type or statement that is inappropriate.	(0-9 marks) In more than six places, you choose variables and statements that are inappropriate.

The demonstration must be completed within 1 week of the due date. The student must demo the code which was submitted to the TA. The TA has the right to assign a mark as low as 0 as a final grade for the whole assignment if he is not satisfied with the demonstration portion of the assignment. The TA may deduct up to 5% from the assignment's final mark for errors in spelling and grammar.

Working Together

If you decide to work with someone from the class or to use resources that you found online or in a book (besides the course textbooks), you **must** cite these sources. When handing in your assignment, please specify who you worked with and list these sources. You will be required to demonstrate your knowledge of how the code performs its task to the TA to get full marks on the assignment. If the TA feels that you do not fully understand what you have written, he may decide to reduce your assignment mark. An example question that the TA could ask would be “what would happen if I changed this line of code to this” (explains the change).

Handing in your assignment

For this assignment, email your programs to your TA on or before the due date. Make sure that your email client program saves a copy of the email you send to your TA. In the event of email problems, we need the header information from your original email to ensure that you submitted your assignment on time.