CSCI 566 **Spring 2015**

Grad Assignment 2
Arrays

Overview

In this assignment, you will analyze three types of household utility expenses (gas, electricity and water) over 12 months.

You will read the expenses from a file into 3 one-dimensional arrays of type double to store the three utility expenses for 12 months. You will need to declare a symbolic constant that represents the size of a single array:

```
const int NUM_MON = 12;      //number of months;
```

You need to do the following analysis on the data you read in:

1. For each type of utility expenses, you need to calculate and print out:
2. `the sum over 12 months;`
3. `the average of 12 months;`
4. `the highest cost;`
5. `the lowest cost;`
6. For each month, you need to calculate and print out the sum of 3 expenses of each month; Then sort the sum of monthly expenses in ascending order; print out the sorted monthly expense; print out the highest monthly expense, the lowest monthly expense, and the average monthly expense.

   Numbers should be printed such that there are 2 digits after the decimal point.

Input

Your program will read from a file by explicitly opening the file.

The file's format is as follows: Each line is either a string or a double. It starts with a string that describes the type of utility of the next 12 lines (doubles) that are the expenses from month 1 to month 12. Then (at line 14), it is again a string that describes the type of next 12 numbers. Since there are 3 types of expenses, the file has a total of 39 lines:

```
gas
```

```
207.14              <-- expense for month 1
177.34              <-- expense for month 2
...
170.90              <-- expense for month 12
electricity
70.34
60.45
...
50.90
water
40.56
42.67
...
49.34
```

You can get the file "expenses.txt" from here.

How to open and read from a file

1. Declare *#include <fstream>*.
2. In your *buildArrays()* function, declare an input file stream variable:

   ```
   ifstream inFile;
   ```

3. "Open" the file (connect the program to the file). Also check to make sure that the file was opened successfully:
4. `inFile.open("expenses.txt");`
5. `if (inFile.fail())`
6. `   {`
7. `   cout << "Unable to open expenses.txt file\n";`
8. `   exit(1);`
9. `   }`
10. To read a series of strings and expenses from the file until the end of file is reached, use your input file stream variable instead of *cin* with the following loop:
11. `  read a string into some temporary string variable. (You don't need to use it in this assignment.)`
12.
13. `  use a for loop to read the 12 doubles into one of the dimensional arrays:`
14. `    inFile >> gas[month];  // this reads a value into one spot of the array`

    Repeat the above process 2 more times: once for the electricity array and once for the water array

    Remember to use the defined constant for the total number of months.

15. After you have read the last value from the input file and reached end of file, you should "close" the file:

    ```
    inFile.close();
    ```

Functions to Write and Use

Write and use the following functions for this program:

void buildArrays(double gas[], double electricity[], double water[])

This function will be called to load all the data into the three one-dimensional arrays. Use the logic presented in step 4 above.

void printUtilityStat(string caption, double array[], int size)

This function will be called 3 times for three utilities in the program. It in turn calls the statistical functions (mean, sum, high, low) to print out the statistics.

void getSumArray(double gas[], double electricity[], double water[], double sums[])

This function takes in four one-dimensional arrays. This function takes each of the utility values for a specific month and adds them up. The resulting sum is then stored in the corresponding location of the sums array.

For example, sums[0] is equal to the sum of gas[0], electricity[0], and water[0].

void printArray(string caption, double array[], int size)

This function should print the current contents of a single array, with a caption (title) which is supplied as the first argument, and another array of int. When printing the numbers, print one line for the caption. Then print out the contents of the array one entry per line. Add a '$' before prinitng each amount.

This function will be called twice in main: once to show the contents of the sums array before it is sorted, and once to show the contents after it has been sorted.

double mean(double array[], int size)

This function returns the mean (average) of the numbers in the array passed to it. This function (and the three functions below) will be used for calculating the means for each utility and for monthly total. Note that this function can call the sum() below to avoid duplicated logic.

double sum(double array[], int size)

This function returns the sum of the numbers in the array passed to it.

double low(double array[], int size)

This function returns the smallest value in the array.

double high(double array[], int size)

This function returns the largest value in the array.

void sortArray(double array[], int size)

This function sorts the array given by the first argument (in ascending order). Use the Selection Sort logic similar to that given in the book and lecture. You can write sub-functions used by *sortArray()* (as given in the book) or not, your choice. The second argument gives the number of numbers in the array to sort.

Here's another take on the Selection Sort logic, with C/C++ source code shown. (You need to change the data type to double.)

Sample Output

(these numbers are made up and are in no way consistent or correct):

```
*** gas ***

sum over 12 months: $1000.45          average: $150.45

highest cost: $225.45                 lowest cost: $30.23

(same for electricity and water)


*** monthly total ***

$240.30
...
$187.90


*** sorted monthly total ***

$90.34
...
$240.30


The highest monthly expense is $240.30.

The lowest monthly expense is $90.34.

The average monthly expense is $180.78.
```

Other Implementation Details

1. Your *main()* function is (as usual) the boss. It should call *buildArrays()* to fill in the three arrays. Then, call *printUtilityStat()* three times to print the statistics for the three utilities, which in turn calls the statistical functions (mean, sum, high, low).

   Then call *getSumArray()* to get the one dimensional array of monthly sum of three utilities After that, print the unsorted array of monthly sum first, then sort the array, and print them again to show the sorted contents. Supply meaningful labels to the *printArray()* calls.

   Finally you need to print out the mean, low and high of the monthly sum array. Note that you will not call the printUtilityStat() this time. In fact, you only need to call the mean() but not low() or high(), because you already have a sorted array and you know where are the highest and lowest numbers. Print them according to the format shown above.

   How do you know if your results are correct? Make up a small data file of your own, with, say 4 or 5 months worth of data. You can do this using the Quincy editor and save it. Do the calculations by hand - and then run the program using the file. See if the two results agree. If they don't, figure out why. If they agree, you can be reasonably sure your program is correct.

2. Document your program according to the course Documentation Standards.
3. Hand in your source code printout, and submit the program electronically via the web.