Homework 7: More Bar Graphs

Assigned: Monday 15 April
Due Date: Monday 22 April

Objectives

More practice with nested `while` loops.

Assignment

This assignment is a continuation of Classwork 7. (NOTE: You must still submit Classwork 7 and Homework 7 separately.) For this homework assignment, embellish your program from Classwork 7 with the following features:

- Since bar graphs do not make sense for negative numbers, add another inner while loop to your program that keeps pestering the user until they enter a number strictly greater than zero. NOTE: This means you'll have to modify your if-else if-else block from Classwork 7.
- After the *outer* while loop terminates, print out the *largest* number entered by the user.
- After the *outer* while loop terminates, print out the *average* number entered by the user. Use a comment when you're doing this calculation to explain why you can calculate largest number inside of the outer while loop, but average is best computed outside of it.

Example Compilation and Execution

```
[arsenaul@linux1 hw7]$ gcc -Wall bar2.c
[arsenaul@linux1 hw7]$ ./a.out
Enter a positive number (type 'quit' to end) 30
******************************
Enter a positive number (type 'quit' to end) 29
*****************************
Enter a positive number (type 'quit' to end) 17
*****************
Enter a positive number (type 'quit' to end) 28
****************************
Enter a positive number (type 'quit' to end) 50
**************************************************
Enter a positive number (type 'quit' to end) 67
*******************************************************************
Enter a positive number (type 'quit' to end) 36
************************************
Enter a positive number (type 'quit' to end) 19
*******************
Enter a positive number (type 'quit' to end) 17
*****************
Enter a positive number (type 'quit' to end) quit
Sum = 293, Count = 9
```

```
Largest number found = 67
Average number found = 32.5556

[arsenaul@linux1 hw7]% ./a.out
Enter a positive number (type ’quit’ to end) -2
*** -2 is not positive, re-enter. ***
Enter a positive number (type ’quit’ to end) 12
************
Enter a positive number (type ’quit’ to end) 14
**************
Enter a positive number (type ’quit’ to end) 9
*********
Enter a positive number (type ’quit’ to end) 0
*** 0 is not positive, re-enter. ***
Enter a positive number (type ’quit’ to end) -2
*** -2 is not positive, re-enter. ***
Enter a positive number (type ’quit’ to end) 8
********
Enter a positive number (type ’quit’ to end) quit
Sum = 43, Count = 4
Largest number found = 14
Average number found = 10.75
[arsenaul@linux1 hw7]$
```

Notes

- Make a copy of your program from Classwork 7.

  ```
  [arsenaul@linux1 ~]$ cp ~/cs104/cw7/bar.c ~/cs104/hw7/bar2.c
  ```

- Think about how to calculate the sum of the numbers entered by the user. Use a `float` variable so you get a floating-point value when you compute the average.
- Since the numbers entered by the user must be positive, you can initialize the maximum value to zero. **Explain why this works in the variable declaration comment**.
- Make sure that you handle the case where the user types in *quit* without entering any numbers.
- NOTE: You cannot use a `break` statement to break out of two nested while loops. A `break` statement in an inner while loop only breaks from the inner-most loop that contains the `break` statement. It will NOT exit from the outer loop. Instead, an option is to use another variable to keep track of when a `break` in the outer loop would be appropriate.

Grading Rubric

- Header comment: 2 points
- Body comments: 3 points
- Compiles: 40 points

- Gets Max: 15 points
- Gets Avg: 5 points
- Gets Sum: 10 points
- Gets Count: 10 points
- Catches negative numbers: 15 points

What to Submit

Use the `script` command to record yourself compiling your program and running your program 3 times using different numbers. Use `exit` to terminate the recording. Only record yourself compiling and running your program. **DO NOT** record yourself editing your program. If you mistakenly start up `nano` while running script, just exit from script and start over. (The new typescript file will overwrite the old one.) When you are done, check the contents of your typescript file. Make sure it does not include you editing your program, and that it is not garbled. Then submit your program and typescript file.

```
[arsenaul@linux1 hw7]$ cat typescript
    NOTE: The contents of your file should display here.
[cos1@linux1 hw7]$ submit cmsc104_arsenaul hw7 bar2.c typescript
```

Verify Submission

If you *think* you submitted the assignment, but the `submitls` command doesn't show you your file names, then the files were **not** submitted and no grade will be given.

```
[arsenaul@linux1 hw7]$ submitls cmsc104_arsenaul hw7
```

Last modified: 08 February 2023