Author: Cuyler Frisby

Date: 8/3/2017

Assignment: Assignment 7.b Comparisons

Comparison with Andrew Tjossem

My program is better than Andrew's in that I included a default constructor for the Person class, even though this was not explicitly required in the assignment description. My understanding is that it is always best practice to include a default constructor if any other constructor is defined for a class. Additionally, I only defined variables in my stdDev function as they were used, while Andrew defined all of his variables at the beginning of the program. I have a preference for defining variables as they are needed so that the variable definition is close to where it is first used.

Andrew's program is better in that he included more documentation than I did. For example, in his declaration for his Person class, he included comments that explained what each function does. I did not include any comments in my Person class declaration. He also had additional comments throughout his stdDev function that make it easier to follow the steps he used to calculate the standard deviation.

Comparison with Aylish Bateman

Aylish did not include a header for her "Person.hpp" or "Person.cpp" files, while I did. Additionally, each time Aylish used an object from the std namespace in "Person.hpp", such as a string, she used the "std::string" notation. I, on the other hand, including "using" statements for each object I used from the std namespace (such as using std::string;). Thus, I did not have to type "std::" each time I used an object from namespace std. While both methods are correct, I believe my approach makes the code cleaner and easier to read.

Aylish did a great job documenting her stdDev function, making it much easier to follow for someone who is unfamiliar with how standard deviation is calculated. My documentation was much briefer and only generally explains what the code is doing. Aylish also included more intermediate variables than I did. While neither method is necessarily incorrect, I believe Aylish's approach would make it easier to debug the code by tracing the values of each variable.

Comparison with Scott Dick

My code does a better job than Scott's in avoiding namespace pollution. Scott simply uses a "using namespace std;" statement at the beginning of each of his files. While this works, it can lead to potential problems when working on larger projects by introducing namespace pollution. To avoid this, I used individual "using" statements for each object I included in my code, such as "using std::cout;"

Scott's stdDev function is more compact than mine and potentially more efficient, as it uses fewer steps to calculate the standard deviation. However, Scott's code did not compile at the time that I tested it, so I cannot definitively say whether his function works as intended.

What have you learned from looking at other people's code, and how can you apply it in future assignments?

I have realized that it is valuable to look at other people's code to see not only their stylistic differences, but their different approaches to solving a problem. There are usually many different ways to obtain the same result in a program. Aylish's code in particular showed me how I can improve my documentation in future assignments. I usually only include very general statements about what the code is doing, but she took the time to write very detailed comments that are easy to understand for someone who may be looking at the code for the first time.

From Aylish's code, I also learned the value of only including a single calculation on each line. While there may be times where writing more compact code is desirable, her code was very easy to follow, not only because of her documentation, but because she was very systematic in only performing one calculation in each statement of her code. In future assignments, I will likely incorporate this style into my own coding, rather than always combining several steps into a single statement.