Jacob Burton
Assignment 7 comparisons

For each pairwise comparison, where do you think your program is better? Why? Where do you think the other program is better? Why? (be detailed)

7.a

Jacob vs. Rand Black:

I felt the biggest difference between our 7.a programs was indentation. His file did not follow normal coding conventions of indenting code that belongs within a specific block of code. To me this is a big deal because, in long and complex programs, it can greatly affect readability. Especially when trying to look at another person's code and follow the logic to determine what that code is doing. I also feel like my variable naming is a little bit more descriptive of the data it holds. Yeah, using m1 and m2 to indicate the two median values could be considered a little vague, but with the context of the program and what it was trying to do I felt that was descriptive enough for its purpose. One thing Rand's code did better than mine was not be broken. I made the mistake of not dividing the sum of m1 and m2 by 2.0, which made the average between the two median values an integer. He handle the problem simply by casting his two median values as doubles.

Jacob vs. Michele Costello:

I felt Michele did a great job with this program. The code is clean, she follows syntax and coding conventions, so her program is very easy to read and understand. She also did some very detailed commenting. She didn't just comment methods/functions, but also steps taken/calculations throughout her code so that you can very easily follow what she's trying to do. She also included code to check for invalid inputs (for example, checking for array sizes inputs of <= 0), which is a great habit to have and can help eliminate hard to find bugs. These little things she's doing will greatly benefit her once she starts working on more complex programs, and I want to make a conscious effort to start doing that myself. As mentioned in the previous section, her code also worked. One thing I feel I did better in mine (and this may just be preference) is doing as much of the calculations needed outside of my conditional if/else statement. From my experience (this may just be my personal preference), keeping conditional statements as small/simple as possible greatly enhances readability of code.

Jacob vs. William Jones:

I also felt Williams code was very clean and easy to read. I also approve of his use of a mean variable = size / 2 to find the median values for both odd and even. He didn't have as many variable declarations as I did and was still able to keep his conditional statement very simple and readable. His use of comments also helped clarify what the if/else conditions were checking for. It is also probably worth mentioning again that William's code worked. One thing I personally like better about my program is spacing between calculations done in the code (for example

size % 2 == 1). I like to space each piece of a line of code so that it is easier to read. Once again probably just a personal preference more than anything else.

7.b

Jacob vs. Rand Black:

The only real issue I take with Rand's code for 7.b is code conventions. He used capitalized variable names for his parameters, which is usually reserved for class objects. He also has the same indentation issue I mention in my comparison of 7.a, which I feel greatly reduces readability of his code. I felt like my use of variable naming was more descriptive of what those variables stored was an advantage my code had over Rand's. I also used more commenting within my code, especially in Person.cpp to describe what the constructor/method were for.

Jacob vs. Michele Costello:

Once again, I felt Michele's code was great. Very clean, well commented, great variable naming. I definitely need to follow her lead in commenting more. One thing I like better about my code is the simplicity of my for loops. Like I stated before with conditional if/else statements, I like to keep those as short and simple as possible for readability reasons.

Jacob vs. William Jones:

I also felt that William's 7.b was very good. Very clean and easy to read. I also like that he commented his constructors and methods in his header file (though I would have just called the methods "methods" and not "getters". I like that his stdDev file was also simple, and minimized the number of variables needed to write a functioning program. One thing I prefer about mine over his is the use of the variable name "variance". For someone looking at this code who hasn't looked at the definition of the standard deviation of a population, I think a more descriptive variable is useful. I also chose to use a more descriptive name for the length of my array ("var" vs "length").

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

One thing I took from other's code is the variety in style of writing code. William did a great job of not creating a lot of variables he didn't necessarily need to. While I think this is a great practice, I do also understand the desire to create a variable that makes your code's functionality more apparent. Another thing reading their code reinforced is the importance of following code formatting conventions, and the value of comments. Both of these greatly enhance readability of code, which I find to be extremely important when reading someone else's code or going back to look at a program you wrote some time in the past. Being able to quickly understand what's going in a programs functions, constructors, methods, steps, etc., is essential in the workplace. I'd definitely like to try and emphasize these practices in the future when writing code.