QA Concept

In our quality assurance concept, we want to target 3 main things: the efficacy of our code, the documentation of the methods and processes and the total lines of code per class. These 3 main things will help us understand our code better, thus making it easier to improve the project moving further. The total lines of code will also help us visualize how we've progressed through the analyzed points in time.

The *code coverage* will help us determine what parts of our code hasn't been tested yet. The main component we have chosen to test in this part is our *ProtocolExecutor.class*, that exectues all commands from the client, for example, the starting of a game, sending a message to all players and other such similar client/server communications. There are a variety of things to test here. Testing the logging process of each client, testing the card sizes at the beginning of each game, testing that the cards are being dealt correctly at the beginning is all a part of this class, and thus a huge contributing factor to the success of the game. The coverage will also show us how many methods we have in our program.

The *documentation* analysis will show us how well we're commenting our code and describing each process, method and class. Documentation is extremely important, especially working in a team, as the other team members need to be able to understand what each individual was trying to achieve. We all write code quite differently, and as such, analyzing how many comment/javadoc descriptions have been written is very useful to see what methods and classes will need to be improved.

The *total lines of code* metric will show us which classes we have invested the most work in and will also show the average length of classes. This is very useful to have an oversight of all classes and to see if there are classes where things can be shortened/branched off. It will also help keep track of our project through the the timeline and to compare with our GANTT Project Timeline. It is a good way to see how well we've stuck to our project plan and if the parts of code were expanded around the time they were supposed to be expanded at.

We will also take a look at how many Logging statements we used (Apache Log4j2) to determine where and how well we used this library for debugging and logging purposes.

servLog (25.) – 66
Client – 37

total: 103

Metrics

The data points that we have chosen are from the 5th of April, the 20th of April and the 26th of April. The 5th marked the end of MS3, the 20th was right after most of our GUI was completely implemented, and the 26th was the day before the MS4 deadline.

We have compiled