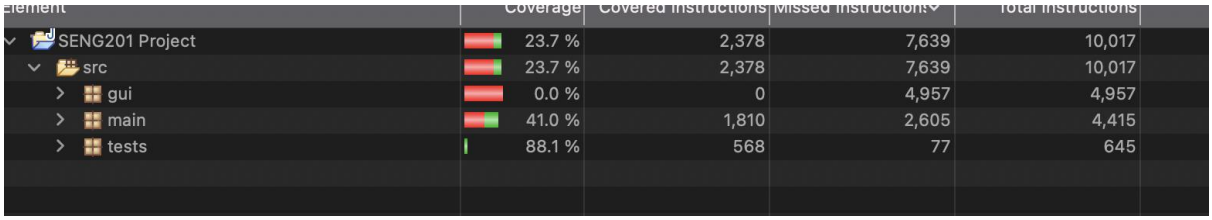
Seng201 Assignment report **Angus Burrowes 15875443**
Jed Atkinson 23463669

Structure and design

Our project implemented object orientated programing for our game to run. Our game is structured into classes which stand outside of our user interface. Our classes come in varying levels of complexity and these classes are managed by an external game manager. Our game manager also reduced the amount of calls we needed for each object.
This made it so we could make our code as maintainable and reusable as possible. For easy debugging we made it, so our classes preserve their own attributes by setting them to private. This is so we know exactly the class where the issue is occurring, and this helped us tremendously when we came across issues while developing the game and made the debugging process much simpler than it could have been. A design decision we decided to make when planning our project was to use inheritance which would make our program less complex and easier to debug. This is because in our game we have many similar classes that only have slight differences. For example, we have a parent class monster with 6 sub classes for different type of monsters who all have a slightly different implementation. We also used an interface purchasable which is implemented in monster and item classes. We did this to reduce complexity as both classes were to be purchasable from our store.

Junit testing

While we were developing our classes for our project, we thought it would be a good idea to try testing while we developed as it would allow us to incrementally build up our program with small parts that we knew worked. We thought would help us with debugging. While doing this we found it difficult to think of useful Junit tests, that weren't overly complicated or mindlessly simple. This is what lead to us having a lower than anticipated j-unit test coverage. Instead of using J-unit testing, we did our own manual testing. We set out our GUI in a way that we could test the core logic of the game without having to have GUI j unit tests. We also manually tested the GUI, and purposely tried to break our game. The more we tested the more streamline the game became, and by the end of development, both Jed and I were happy with the game's performance and lack of fatal errors. So, despite the low figure, the functionality of the game has been tested extensively. There were multiple methods that we were unable to junit test, due to the fact it involved random number generation. So instead, we manually tested these classes. Our initial inspection of our J unit test coverage came in at 23.7%. This is well below what we think it should be. Part of this low coverage is because all the GUI testing was done manually as this was the easiest way to do it. Therefore, the GUI shows a test coverage of 0% where its likely to be closer to 80%. Our main code has a test coverage of 41%. This is also likely to be higher as we didn't test any subclasses with junit test as we had already tested their parent classes. We also didn't test getters and setters as they are too simple for us to need to do that and we can easily manually verify them.



Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
SENG201 Project	23.7 %	2,378	7,639	10,017
src	23.7 %	2,378	7,639	10,017
gui	0.0 %	0	4,957	4,957
main	41.0 %	1,810	2,605	4,415
tests	88.1 %	568	77	645

Feedback and thoughts

Prior to starting this assignment, neither me nor Jed, had never developed anything like our monster fighter game. Despite the project being extremely daunting to begin with, we both had a lot of fun in-betweens those stressful moments and felt like the project was a worthwhile learning experience for the both of us. I believe the project significantly improved our java abilities and gave me the opportunity to develop a game, something I had wanted to do since I was a child. Despite the work required, I found this project to be one of the most enjoyable projects I have done at university.

Reflection

We had close to 2 months to work on our monster fighter project, and this seemed like ample time for us to complete our project, however with both of us being relatively new to java and both had never coded anything nearly this sophisticated we found it difficult to make progress when we were starting out due to us having some many queries and unknowns. As the project went on, Jed and I became much more comfortable using java and once we had sketched out A UML use case diagram and class diagram, we felt much more comfortable with what we had to achieve, and the task seemed much less daunting, and we were able to make good progress. I think that Jed and I worked particularly well as a team. We both have a different skill set, with me studying financial engineering, and Jed studying computer science, but I think we utilized our skills effectively. For example, Jed was extremely comfortable with the GUI, so he took the lead for that, and I took the lead for planning, UML and creating classes. One part of the project that could have gone better was our time management. During the last week of the assignment, what we had left to do turned out to be more than expected, however we worked hard and managed to get our project complete. If we were to do this again, I think it would have been a good idea for us to implement some design patterns or architectural patterns, to make our code more dynamic and have a better structure. For example, we could have used the builder design pattern. I also think it would have been smart for us to have done more work during the holidays and distribute our time more evenly, rather than having to put in a lot of work in the last week. However overall, we found this project to be a huge learning curve and regardless of our final mark we are proud of what we have accomplished.

Effort and contribution.

Jed and I meet up the week the project was announced to begin planning how to tackle the project. Since then, we have average 8 hours a week, with significantly more time spent during the final week. (Close to 20 hours each).

During development Angus did most of the UML, planning and coding of classes and report, while Jed focused on the command line application, and the GUI coding. We both spent a similar amount of time testing. Overall, we believe we both put in 50% of the effort into completing the project. Having a group partner, we could both rely on was extremely important in completing the assignment.