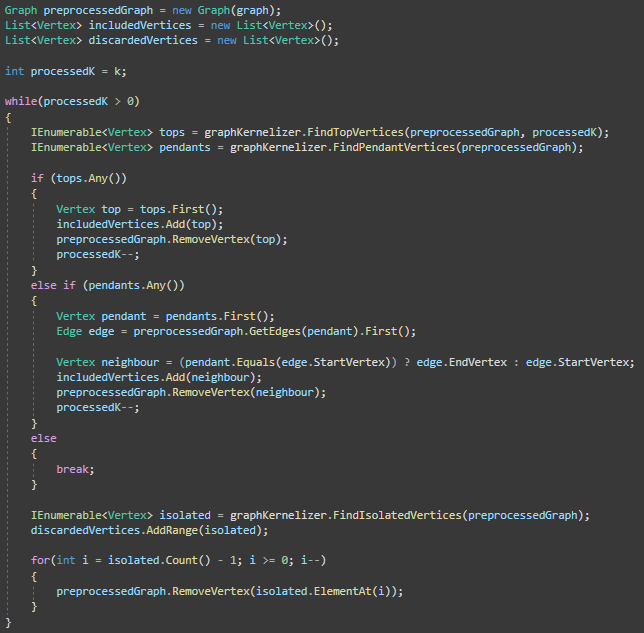
Week 5

# Improved pre-processing

During this week (and the may vacation) we worked on improving the pre-processing for **vertex cover of k size** to include vertex tops, these are vertices that have a degree higher than k, which means that if a vertex cover of k exists, that vertex *must* be in it. We did this by transforming the prepossessing function into one that works iteratively in a loop, kernelizing the graph one vertex at a time so that the remaining degree (k) can be updated on the fly. The loop of this function looks as follows:



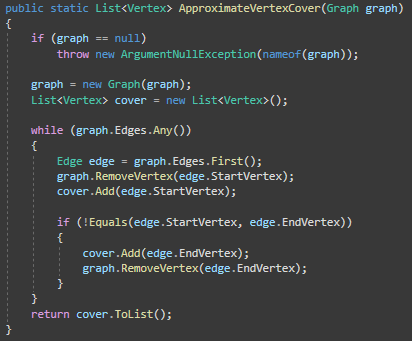
We also updated the corresponding tests to work with the newly defined behaviour. Overall, this refactor has given us a significant improvement to performance when it comes to pre-processing.

# Take 2 approximation

In addition, we also worked on an approximation algorithm for our vertex cover. For this, we decided to implement the take-2 algorithm for approximation vertex cover for 2 primary reasons:

* Simplicity: The algorithm has a rather simple structure, which makes it easy to debug or reason with.
* Approximation factor: The take-2 algorithm has a approximation factor of 2, which means that any generated solution is at most 2 factors removed from the optimum: It is ensured that any solution found is smaller than or equal to 2 times the optimum solution.

This is our implementation of the take-2 approximation algorithm:



We also went ahead and wrote some basic unit tests for this new function. We have not yet added this functionality to the UI of the program, but hope to do so in the upcoming week.

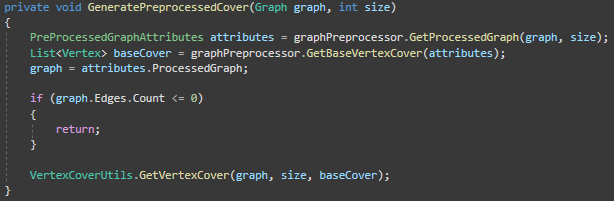
# Performance testing

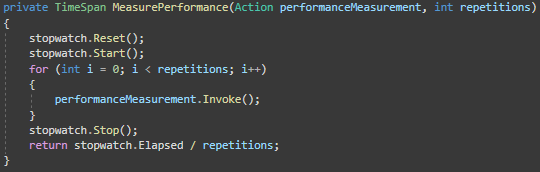
Finally, we started working on implementing a way to measure the performance of 2 approaches to finding a vertex cover:

* Brute force: Running the vertex cover algorithm on its own.
* Pre-processed brute force: Running the vertex cover on a pre-processed graph.

We want to perform this test by processing graphs of different sizes using all both approaches and exporting the results to a CSV file. The code for this looks as follows:







We also added a new data structure to contain a finished performance test:



We have almost completed the implementation of these performance tests, and are currently in the process of writing tests, which we hope to finish in the upcoming week.