**Team33 commits to this code base!**
Codebase: Team 5

## Language and Libraries
Both of our teams worked in the same language (Python) with mostly the same packages (StdLib). The only difference between packages was that Team 13 used Numpy and Team 5 did not. We decided that we can always go forward without Numpy, or add it in later. It can sometimes be better to use fewer dependencies anyway.

## Features
Team 5 implemented more classes to represent the objects of the game (such as a Stone class, which wraps an Enum).

Team 13 did not make much use of class structures for things like Points and Stones. Team 13 did not implement many unit tests for 5.2, so we are not as confident that their Player component is implemented correctly. We think the Team 5 Player “make-a-move” function might work for n > 1 because Team 5 has unit tests for that.

Besides a few distinctions, most of the features we implemented were the same. A lot of the differences between the codebases came down to design decisions.

## Design Decisions
Team 5 wrote parsing library functions that are used inside the test driver, so that components only interface with instances of models. Ex: a board class will use a StoneEnum object instead of a string.

Team 5 tried to avoid coupling components with what the test inputs look like, so that future components can make use of those components without knowledge of what the testing inputs look like.

Team 13 made the decision to make their Board class have no knowledge of the “rules” of the game, so the RuleChecker class plays the moves, checks if the ko rule is violated, etc. The Board class only has knowledge of what is physically on a board, and only handles interactions that change what Stones are on certain Points of the Board.