**Cover Page:**

Final mark allocation: 50:50

**Development Log:**

Key:

Developer 1 (Dev 1) – 160221 Developer 2 (Dev 2) – 193960 Observer – O Driver – D

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Date | Time | Duration | Role(s) | Brief description of what was accomplished. | Candidate Number Signatures |
| Monday October 18th  2021 | 3:30-4:30pm | 1 hour | D = Dev 1  O= Dev 2 | Discussing design/direction, setting up initial classes and files. | 160221  193960 |
| Tuesday October 19th 2021 | 5:30-7:30pm | 2 hours | First half:  D = Dev 2  O = Dev 1  Second half  D = Dev 1  O = Dev 2 | Created file reading methods and some object methods. | 160221  193960 |
| Friday October 22nd 2021 | 6:00-7:30pm | 1 hour and 30 mins | D = Dev 2  O = Dev 1 | Creating more methods, and setting up atomic actions | 160221  193960 |
| Monday October 25th 2021 | 5:00-6:30pm | 1 hour and 30 mins | D = Dev 1  O = Dev 2 | Bug fixing and reducing overhead. Began multithreading. | 160221  193960 |
| Monday November 1st 2021 | 6:00-8:00pm | 2 hours | D = Dev 1  O = Dev 2 | Some streamlining/restructuring. of existing code. | 160221  193960 |
| Tuesday November 2nd 2021 | 5:30-7:30pm | 2 hours | First half:  D = Dev 1  O = Dev 2  Second half:  D = Dev 2  O = Dev 1 | Added file writing, bug fixing, checked coded against specification. | 160221  193960 |
| Friday November 5th 2021 | 9:00-10:00pm | 1 hour | D = Dev 1  O = Dev 2 | Bug fixing, and discussion on remaining tasks to tackle and code commenting. | 160221  193960 |
| Sunday November 7th 2021 | 5:30-7:30pm | 2 hours | D = Dev 2  O = Dev 1 | Running, testing and fixing remaining bugs, and code commenting. | 160221  193960 |
| Tuesday November 9th 2021 | 5:30-7:30pm | 2 hours | First half:  D = Dev 2  O = Dev 1  Second half:  D = Dev 1  O = Dev 2 | Finish report, final bug fixes. Final review of project to make any last minute changes. Submission. | 160221  193960 |

**Design Choices:**

We originally considered handling ensuring that bags contained enough pebbles for the game in the code. With the idea behind this being that when user is entering files, they are really just entering a distribution of weights to create the probability they would like to have for their game (due to the fact that pebbles are drawn at random the only way to increase the probability of a certain weight is to increase the amount of pebbles of that weight in a bag).

Hence, we thought about implementing this by taking this distribution and duplicating it until the total pebbles were greater than 11 to make it valid for 1 player, and then to duplicate this new distribution for as many players as the user entered to ensure the game would be playable for whatever number of players the user entered as well regardless of the original amount of pebbles taken from the flies. This would all be done while maintaining the probability distribution that the user entered as the number of pebbles is simply corrected through duplication so the frequency of a certain pebble weight would never change.

We eventually decided not to handle this for the user as it was quite similar to the idea of handling for the user entering files that that contained a distribution of pebble weights that would make it so that a hand could never hold a weight of 100 where both would situations would require the code to handle user inputs that were correct syntactically but still caused issues for a perfect running of the game. Therefore for continuity’s sake we chose to simply return to the user that they did not have enough pebbles in one of their entered files.

We chose not to handle distributions of pebble weights that make a hand reaching 100 impossible and just to let the code run. This is because it will often be virtually impossible to account for all the permutations of total weight in a hand of the total distribution of pebble weights across all 3 bags. So while it would be feasible to create methods to catch certain constraints (such as checking whether there were pebbles in the bags that make it impossible to get 100 with them in your hand), we found that for it to make enough of a difference that we wanted to implement it the method would have to brute-force its way through permutations which became completely infeasible as number of player/number of pebbles increases, or to use approximations which could still become infeasible and was also quite complicated to implement.