**Complexity Time for Asgwillanga Cavern by William Timani**

The Big-O complexity of my program can be calculated based on how many rooms are visited. Number of rooms can be represented as N in this case. Each room is passed through the nextRoom() function which will iterate through loops to find rooms connected to the current room. Each loop iterates through a single digit of the id until one of the digit combinations passes all the rules. In this case, the complexity for looping will be X \* Y \* Z, where X is the id limit for the first digit, Y is the id limit for the second digit, and Z is the id limit for the third digit. Each visited room must cycle through this loop therefore we reach N \* X \* Y \* Z. Inside the loop are function calls. The only function call that does not have a constant time complexity is the checkVisitedRooms() function which cycles through all visited rooms. This can have a worst case of O(N). We can now reach the equation O(N\*X\*Y\*Z) + O(N) + O(1); the O(1) represents all function calls within the loop that have a constant time. O(N) and O(1) can be disregarded since they have an insignificant impact on complexity compared to O(N\*X\*Y\*Z). Normally, the coefficients X,Y,Z would also be reduced to one, however in this case they should be left alone since O(N) does not represent the complexity time accurately for the program. Therefore, the final complexity for my program is O(N\*X\*Y\*Z) where N is the number of visited rooms, X is the id limit for the first digit, Y is the id limit for the second digit, and Z is the id limit for the third digit.