2023300038 Expt - 8

David ~~Double~~ Daniels

The given backtracking algorithm for sum of subset problem has been implemented in C++. It uses DFS structure (recursion) to generate all possible subsets and check if it is valid or not. It uses 2 major optimisations for sum, sum remaining to skip many unnecessary subsets. To visualise the DFS, it is best to use recursion tree.

The time complexity of this algo is $O(2^n)$. This is because, in the worst case, we must make the decision of including or excluding number from subset 2 times for each number.

But the overall time complexity (including printing the valid subarrays is $O(n \times 2^n)$)

The space complexity is $O(n)$ for this algo. This is because a recursive call stack is used. Using DFS, the max number of nodes that will occur is n. Using BFS, the max number is 2^n

Thus, I have learned how does recursion trees and backtracking works.