Implementation Details:

- Initially we implemented an algorithm which mimics the way semantic tableaux behaves.
- Similar to the main code, we store the input as a list of lists. We then call the split() function.
- The split function is a recursive function which mimics a semantic tableau. We do not explicitly create a tree but use the recursion tree itself.
- We pick a clause from the list of clauses and split the node into number of children equal to number of literals present in that clause. We then go down on each of the clause one by one, recursively.
- If a leaf closes, then we go back to its parent node i.e. the calling function. Else we proceed down the root until we get a satisfying assignment.
- However, note that it is present only as a proof of concept and should not be used for any real purpose since it is extremely slow, with an average time complexity greater than that of the trivial O(2^n) algorithm by enumerating all entries in the truth table.
- This is because we discard and information gain whenever we close a node and there is no definite systematic way of choosing the order to proceed.

How to run:

Copy the file on your machine. Use “python [path]” to run the script where path is the path of the file. Enter path of the cnf formula when prompted.