Assignment 1 – part 3

Write a class called Formula<yourName> which provides the following data structures and functionalities for the formula.

1. An array to track of the truth values assigned to the variables. Initially, all variables are unassigned. i.e. set to “final int na = 0”.
2. A linked list to keep track of all the sublists of clauses that are satisfied so far, and also the sublist of clauses that remain to be satisfied. The sublists are separated by a special marker (= -1). Initially, it contains all the clause labels ending with a marker. Note that each clause is represented in this list by the array index where it is stored.
3. functionalities:
4. read(<inputFileName>). This uses a FormulaReader<yourName> object to read the formula in <inputFileName>. Makes a local copy of the number of variables, clauses and the formula.

Initializes the linked list representation of the formula using clause labels.

Initializes the assignment array.

1. isFormulaEmpty() – returns true if the formula is empty else returns false. Uses the linked list representation of the formula. Formula is empty if the first int in it is the marker. Note that the formula that we are testing here is only the unsatisfied part of the formula.
2. isClauseEmpty(<clauseNo>) – returns true if a clause specified by <clauseNo> is empty else returns false. A clause is empty if all its variables are already assigned but still the clause is not satisfied. Uses the first sublist in the linked list representation.
3. hasEmptyClause() – returns true if a formula contains an empty clause else returns false. Even if one clause is not satisfied then the formula is not satisfied. Uses the first sublist in the linked list representation.
4. firstAvailable() – returns the first available variable for backtracking. Uses the assignment array.
5. separateClauses() – moves the first sublist in the linked list representation to a temp list.
6. assign() – assigns a truth value to the first available variable. Splits the first sublist in the linked list representation into two – one containing the clause labels that are satisfied by the latest assignment; and, the other containing the clause labels that are not satisfied by the latest assignment.
7. isClauseSatisfied(**int** cno) - returns true if the clause with label cno is satisfied else returns flase. a clause is satisfied if at least one of its variables is true.
8. printAssignment() - displays the current assignment of the values of the variables.
9. printFormula() - displays the formula in the 2D format.
10. Print() - displays the linked list representation of the clauses and the current assignment of the values of the variables.
11. Any other helper function that you may deem necessary.