In this assignment, you will write a simple Propositional Horn Clause Resolution Engine  in OCaml.

You will first define an oCaml data type to represent the structure of a legitimate Propositional Horn Clause program.

* A program is a sequence of clauses.
* A clause can either be a fact or a rule.
* A fact has a head but no body.
* A rule has a head and a body.
* The head is a positive atom, i.e., a  propositional Letter (represented as a string).
* A body is a sequence of (1 or more) positive atoms, i.e, propositional letters.
* A goal is a sequence of positive atomic formulas.

You will need to develop a back-tracking strategy to explore the resolution search space.   You need to be able to replace a chosen atom in the current goal by subgoals, i.e, the sequence of atoms in the body of a program clause whose head matches the chosen atom in the goal. If a program clause is not found, you need to backtrack to the last step, and try another program clause that may match the chosen subgoal.  For this you will need to stack up goals, and remember where to retry a fresh program clause.

To be deterministic (DFS exploration of resolution space), always pick the first atom in the current goal. Also try the program clauses in the order in which they are listed.

A test program is:

A <- B, C.

A <- D.

B.

C <- E, F.

C <- G.

D <- E, G.

E.

G  <- H, B.

H.

A test goal is:  A, B, C, D