FUNDAMENTALS OF ARTIFICIAL INTELLIGENCE - CS161

Programming Assignment 8 - Due 11:59 p.m. Tuesday, March 5

DO NOT CONSULT ANY OUTSIDE REFERENCES OR SOURCES FOR THIS ASSIGNMENT

Your assignment is to write a resolution theorem prover for propositional logic. Write a top-level function called "refute" which takes a single parameter that is a list of clauses, and returns a proof tree resulting in a contradiction, or returns the atom FAIL if all the clauses are consistent.

The representation of the propositional expressions will be as follows:
- Every symbol except NOT is interpreted as a proposition. e.g. rainy, sunny.
- The symbol "not" will be used to indicate a negation. It will appear only as the first element of a two element list. e.g. (not rainy), (not sunny).
- A clause is a list of literals, where each literal is either a proposition or a negated proposition. e.g. (tuesday thursday (not lecture-day))
- The input knowledge base will be a list of clauses.
  e.g. (((not rainy) (not sunny)) (sunny) (rainy))

The output is a tree of successful resolutions that contributed to proving the contradiction. Since the proof tree is binary, it can be represented by a three element list where the first element is the clause that results from resolving the clauses represented by the second and third elements. The leaf nodes will be single-element lists containing an input clause.

Test your program on the examples below and others as well. Your grade will be based on the following criteria, in order of decreasing importance:
1. Correctness - returns valid proofs when proof exist, returns FAIL when clauses are consistent.
2. Optimality - returns a proof of shortest length, measured by tree depth.
3. Efficiency - uses time and space efficiently.

```
-> (refute '((a) ((not a))))
 (NIL
   ((A))
   (((NOT A))))

-> (refute '(((not rainy) (not sunny)) (sunny) (rainy)))
 (NIL
   ((RAINY))
   (((NOT RAINY))
    (((NOT RAINY) (NOT SUNNY)))
    ((SUNNY))))

-> (refute '((a) (b) (a (not b))))
 FAIL

-> (refute
   '(((not study)) (study (not pass)) (pass (not graduate)) (graduate)))
 (NIL
   (((NOT PASS))
    (((NOT STUDY)))
    ((STUDY (NOT PASS))))
   ((PASS)
    ((PASS (NOT GRADUATE)))
    ((GRADUATE))))

-> (refute
   '((study) (study (not pass)) (pass (not graduate)) ((not graduate))))
 FAIL

-> (refute
   '((p)
     ((not p) (not q) r)
     ((not s) q)
     ((not t) q)
     (t)
     ((not r))))
```

```
  (NIL
    ((P))
    (((NOT P))
     ((Q)
      (((NOT T) Q))
      ((T)))
     (((NOT P) (NOT Q))
      (((NOT P) (NOT Q) R))
      (((NOT R))))))

 -> (refute
    '(((not mon) (not tue))
      ((not tue) (not wed))
      ((not tue) (not fri))
      (mon wed (not lecture))
      (fri (not recitation))
      (lecture recitation (not class))
      (tue)
      (class)))
  (NIL
    ((TUE))
    (((NOT TUE))
     (((NOT LECTURE) (NOT TUE))
      (((NOT MON) (NOT TUE)))
      (((NOT LECTURE) MON (NOT TUE))
       (((NOT TUE) (NOT WED)))
       ((MON WED (NOT LECTURE)))))
     ((LECTURE (NOT TUE))
      (((NOT RECITATION) (NOT TUE))
       (((NOT TUE) (NOT FRI)))
       ((FRI (NOT RECITATION))))
      ((LECTURE RECITATION)
       ((LECTURE RECITATION (NOT CLASS)))
       ((CLASS)))))))
```