**Wumpus world problem**

Implementation of a program that creates a knowledge base and an inference engine for the wumpus world. First of all, the program will create a knowledge base (stored as a text file) storing the rules of the wumpus world, i.e., what we know about pits, monsters, breeze, and stench. Second, it will create an inference engine, that given a knowledge base and a statement determines if, based on the knowledge base, the statement is definitely true, definitely false, or of unknown truth value.

**Command-line Arguments**

The program will be invoked from the commandline as follows:  
  
check\_true\_false wumpus\_rules.txt [additional\_knowledge\_file] [statement\_file]  
  
For example:  
  
check\_true\_false wumpus\_rules.txt kb1.txt statement1.txt

* Argument wumpus\_rules.txt specifies the location of a text file containing the wumpus rules, i.e., the rules that are true in any possible wumpus world, as specified above (once again, note that the specifications above are not identical to the ones in the book).
* Argument [additional\_knowledge\_file] specifies an input file that contains additional information, presumably collected by the agent as it moves from square to square. For example, see [kb3.txt](http://crystal.uta.edu/~gopikrishnav/classes/2020/summer/4308_5360/assmts/optassmt1_files/kb3.txt).
* Argument [statement\_file] specifies an input file that contains a single logical statement. The program will check if, given the information in wumpus\_rules.txt and [additional\_knowledge\_file], the statement in [statement\_file] is definitely true, definitely false, or none of the above.

**Output**

The program will create a text file called "result.txt". Depending on what your inference algorithm determined about the statement being true or false, the output file will contain one of the following four outputs:

* **definitely true**. This will be the output if the knowledge base entails the statement, and the knowledge base does not entail the negation of the statement.
* **definitely false**. This will be the output if the knowledge base entails the negation of the statement, and the knowledge base does not entail the statement.
* **possibly true, possibly false**. This will be the output if the knowledge base entails neither the statement nor the negation of the statement.
* **both true and false**. This will be the output if the knowledge base entails both the statement and the the negation of the statement. This happens when the knowledge base is always false (i.e., when the knowledge base is false for every single row of the truth table).

Note that by "knowledge base" we are referring to the conjunction of all statements contained in wumpus\_rules.txt AND in the additional knowledge file.

**Syntax**

The wumpus rules file and the additional knowledge file contain multiple lines. Each line contains a logical statement. The knowledge base constructed by the program will be a conjunction of all the statements contained in the two files. The sample code (as described later) already does that. The statement file contains a single line, with a single logical statement.  
Statements are given in prefix notation. Some examples of prefix notation are:  
  
(or M\_1\_1 B\_1\_2)  
(and M\_1\_2 S\_1\_1 (not (or M\_1\_3 M\_1\_4)))  
(if M\_1\_1 (and S\_1\_2 S\_1\_3))  
(iff M\_1\_2 (and S\_1\_1 S\_1\_3 S\_2\_2))  
(xor B\_2\_2 P\_1\_2)  
P\_1\_1  
B\_3\_4  
(not P\_1\_1)  
  
Statements can be nested, as shown in the above examples.  
  
Each logical expression should be contained in a single line.

* The wumpus rules file and the additional knowledge file contain a set of logical expressions. The statement file should contain a single logical expression. If it contains more than one logical expression, only the first one is read.
* Lines starting with # are treated as comment lines, and ignored.
* You can have empty lines, but they must be totally empty. If a line has a single space on it (and nothing more) the program will complain and not read the file successfully.

There are six connectives: and, or, xor, not, if, iff. No other connectives are allowed to be used in the input files. Here is some additional information:

* A statement can consist of either a single symbol, or a connective connecting multiple (sub)statements. Notice that this is a recursive definition. In other words, statements are symbols or more complicated statements that we can make by connecting simpler statements with one of the six connectives.
* Connectives "and", "or", and "xor" can connect any number of statements, including 0 statements. It is legal for a statement consisting of an "and", "or", or "xor" connective to have no substatements, e.g., (and). An "and" statement with zero substatements is true. An "or" or "xor" statement with zero substatements is false. An "xor" statement is true if exactly 1 substatement is true (no more, no fewer).
* Connectives "if" and "iff" require exactly two substatements.
* Connective "not" requires exactly one substatement.

The only symbols that are allowed to be used are:

* M\_i\_j (standing for "there is a monster at square (i, j)).
* S\_i\_j (standing for "there is a stench at square (i, j)).
* P\_i\_j (standing for "there is a pit at square (i, j)).
* B\_i\_j (standing for "there is a breeze at square (i, j)).

**The Wumpus Rules**

Here is what we know to be true in any wumpus world, for the purposes of this assignment

* If there is a monster at square (i,j), there is stench at all adjacent squares.
* If there is stench at square (i,j), there is a monster at one of the adjacent squares.
* If there is a pit at square (i,j), there is breeze at all adjacent squares.
* If there is breeze at square (i,j), there is a pit at one or more of the adjacent squares.
* There is one and only one monster (no more, no fewer).
* Squares (1,1), (1,2), (2,1), (2,2) have no monsters and no pits.
* The number of pits can be between 1 and 11.
* We don't care about gold, glitter, and arrows, there will be no information about them in the knowledge base, and no reference to them in the statement.