Fall 2003 CSCI460 Homework4 Due Date: Dec 06 11:59:59pm

Jones, Smith and Clark hold the jobs of *programmer*, *knowledge engineer*, and *manager* (not necessarily in that order). Jones owes the programmer $10. The manager's spouse prohibits borrowing money. Smith is not married. Your task is to figure out which person has which job.

Represent the facts in propositional logic. You should have nine propositional symbols to represent the possible person/job assignments. For example, you might use the symbol ***SM*** to indicate the face that Smith (***S***) is the manager (***M***). You do not need to represent the relation between owing and borrowing, or being married and having a spouse. You can just use these to draw conclusions (e.g., from "Smith is not married" and "the manager's spouse" we know that Smith is ***not*** the manager, which is represented by ***~SM***). The conjunction of all the relevant facts forms a sentence, which you can call ***KB***. The possible answers like ***JP && SK && CM***. There are six such combinations of person/job assignments. Solve the problem by showing that only one of them is implied by ***KB***.

Question 1 (15pts)
What are the nine propositional symbols?
What are the relevant facts?
And what is the KB?

Question 2 (10pts)
What are the six combinations of person/job assignments?

Programming (75pts)

How would we write an agent program to solve the above problem? To check whether a sentence ***S1*** is implied by ***KB*** or not (***KB*** => ***S1***), it is equivalent to check the validity of ***~KB*** || ***S1***. So, your program will be dealing with **proposition variables**, **logical connectives**, and **parentheses**. It can show whether a given sentence is valid or not. You have to account for precedence and associativity. So, your operators will be defined with the following symbols:

NOT *x*	~*x*
x **AND** *y*	*x* && *y*
x **OR** *y*	*x* || *y*

The priority of them are: **NOT** > **AND** > **OR**, and parenthesis is higher than all of them. All variables will be denoted by strings of letters (such as ***JP***, ***SK, CM, P, Q, R …***).

Input:
For the above question, you are required to write your own **problem.txt**. The first line will be your ***KB***, which should be represented by one sentence. The second line is an integer **n**, which specifies the number of problems in the file (e.g., **6** in the example below), that is, the number of questions you wish to ask your agent given its KB. In the following **n** lines, each line is a sentence to be checked, which is composed of variables and logical connectives (~, && and ||). In the above problem, ***JP && SK && CM*** is one of the six sentences to be checked.

The format of the **problem.txt** is as follows, not the real file.

Line Number:	***File Contents:***
1	Your ***KB*** (one sentence)
2	6
3	***JP && SK && CM***
4	sentence 2
5	sentence 3
6	sentence 4
7	sentence 5
8	sentence 6

Output:
All the output should be written to a file called **solution.txt** in the current directory. For each problem in **problem.txt**, there will be a corresponding line in **solution.txt** and it will contain your program output for that sentence. It is either **VALID** or **NOT VALID**.

Caution:

1. Do not expect any command line arguments.
2. There can be any number of white spaces between fields in the input file.
3. Assume that only space or tab will be used for separating the fields in the input file.
4. There will be no "Line Number" string in the problem.txt, don't look for it.

Example test case 1:

problem.txt:

```
~AP && (AP || AQ) && (BP || BQ)
2
AP && BQ
AQ && BP
```

solution.txt

```
NOT VALID
VALID
```

Example test case 2:

problem.txt:

```
P && Q && R
3
P || Q
~P || R
~P && R
```

solution.txt

```
VALID
VALID
NOT VALID
```

Extra test cases will be used when grading your program.