Artificial Intelligence (CS 131)
Assignment 03: Forward Chaining for Propositional Inference (25 points)

Due Sunday, 17 July, 11:59 PM

This exercise involves implementing the *forward-chaining* inference procedure for propositional logic (PL). Your program needs to run from the command line and take the name of a text-file as input. This file will describe a knowledge-base in definite clause form. That is, it will consist of:

1. 0 or more lines, each defining a single *conditional clause*. Each such clause will consist of a *tail*, which will be a conjunction of propositional symbols—using "`AND`" as our conjunction operator—followed by the conditional operator "`THEN`" and a *head* consisting of a single propositional symbol.

2. 0 or more lines, each defining a single *propositional symbols*. Each such symbol will be of the form `pX` where `X` is some positive integer value.

As an example, a file containing:

```
p1 AND p2 AND p3 THEN p4
p4 THEN p5
p1 THEN p2
p1
p3
```

would correspond to the knowledge-base of PL sentences:

$$\{(p_1 \wedge p_2 \wedge p_3) \Rightarrow p_4,\, p_4 \Rightarrow p_5,\, p_1 \Rightarrow p_2,\, p_1,\, p_3\}$$

You can assume that files are in the proper format, and that no conditional clause or propositional symbol appears more than once.

When run, your program will perform the forward-chaining algorithm given by the pseudo-code in Russell& Norvig, chapter 8, and explained in lecture 09:

a. It will run from the command-line taking in the name of the knowledge-base file as input, as for instance:

```
python forward_chain.py kb_00.txt
```

b. It will first read in the knowledge-base file.

c. It will repeatedly prompt the user for a *query*. The user will enter any string:

- If the user enters `end` the program will terminate.
- If the user enters in a propositional symbol that is entailed by the initial knowledge-base, the program will indicate that this is the case.
- If the user enters anything else, the program will indicate that the input is not entailed by the knowledge-base.

For example, if we ran the program on the knowledge-base given above:

- Entering either `p1` or `p3` would lead to a successful entailment, since those symbols are already part of the set.
- Entering `p2` would also lead to a successful entailment, due to the presence of `p1` and the conditional `p1 THEN p2` in the knowledge-base.
- Entering `p3` would also lead to a successful entailment, due to the same reasoning as the prior step, and the additional presence of `p3` and the conditional `p1 AND p2 AND p3 THEN p4` in the knowledge-base
- Entering something like `p5` or `hello` would lead the program to indicate that those inputs were not entailed by the program.

(The assignment distribution includes some sample runs, along with some sample knowledge-bases; you can easily make more of the latter for testing purposes.)

You will hand in source-code for the program. The instructor will compile it and test it by giving it different input files. All sample files are provided, and you can create your own for testing, but you need to use the same format as given above. I prefer that you code the solution in Python or C++, but if you wish to use another language, ensure that it is something that course staff will easily be able to compile (if necessary) and run. When doing your coding, follow these conventions:

1. Provide a short `README` file with your code, explaining exactly what commands or procedures are needed to make it compile and run properly.
2. Follow basic software engineering principles; that is, your code should be clean and well-structured, with comments explaining each method or function, and the usual format conventions to make it readable.