Artificial Intelligence (CS 131)
Assignment 04: Inference in Bayes Nets (80 points)

Due Sunday, 31 July, 11:59 PM

Your goal for this assignment is to write a program that reads in an input file; this file will describe a Bayes Net in terms of

a. A list of variables, each with their possible values.

b. A list of variables, each with their parent variables in the BN.

c. A list of variables, each with their CPTs.

As an example, consider the sample file `burglary.txt`:

```
Burglary T F
Earthquake T F
Alarm T F
JohnCalls T F
MaryCalls T F
# Parents
Alarm Burglary Earthquake
JohnCalls Alarm
MaryCalls Alarm
# Tables
Burglary
0.001
Earthquake
0.002
Alarm
T T 0.95
T F 0.94
F T 0.29
F F 0.001
JohnCalls
T 0.90
F 0.05
MaryCalls
T 0.70
F 0.01
```

This file describes a version of the burglary BN from the text (Figure 13.2, p. 414):[1]

1. First, it shows each variable, with its possible values (here, `T` of `F`).
2. Following the `# Parents` tag, we have the name of each variable, followed by its parents.
3. Following the `# Tables` tag, we have the CPTs for each variable. Note that the rows of the CPT are ordered by the parents as given in the previous section of the input (if there are any), and contain numbers for the first $(d-1)$ values of the variable, in the order given in the first section of the file (where d is its domain size).

   For example, each row of the CPT for the `Alarm` variable starts with a value for its first parent (`Burglary`), followed by a value for its second parent (`Earthquake`), followed by the probability $P(Alarm = T \mid Parents)$.

Look at the various input files to make sure you understand all the details before you begin. In particular, *do not assume* that all variables are always boolean. The sample files also include `sprinklers.txt` (based partly upon p. 435, Figure 13.15 (a)), as well as `books.txt` and `data.txt`. For the latter two, it would be worthwhile to draw them out, to again make sure you are confident about what the input format describes.

Your program will read in the BN file-name, using command line arguments, and then create a Bayes Net corresponding to that file. It will then allow the user to enter queries interactively, using standard input. A query will be in the form:

```
<VARIABLE> | <EVIDENCE_1> = <VAL_1>, ..., <EVIDENCE_m> = <VAL_m>
```

If there is no evidence, then the user will simply enter the name of a variable. Output will be in the form of a distribution over all the values of the query variable. When the user types `quit`, the program should terminate.

Your program will use the enumerative query algorithm outlined in the text (p. 429, Figure 13.11).

[1] The probabilities here are a bit different in the CPT for the *Alarm* node, but the same otherwise. If you understand the input format, then you will be able to say what we have changed in our version. If you can't tell, you should ask.

A sample run of the program—in which the user loads the example network given and then makes three queries:

1. $P(Burglary)$
2. $P(JohnCalls \mid Alarm = T)$
3. $P(Burglary \mid JohnCalls = T, MaryCalls = T)$

would be as follows:

```
> java BayesNet burglary.txt

Loading file "burglary.txt".

Burglary
P(T) = 0.001, P(F) = 0.999

JohnCalls | Alarm = T
P(T) = 0.9, P(F) = 0.1

Burglary | JohnCalls = T, MaryCalls = T
P(T) = 0.284, P(F) = 0.716

quit
```

Note: if possible, your output should look exactly like this. It should be as close as possible, in any case. One thing to note is that the decimal values are formatted to have a maximum of 3 digits precision. The download for this assignment contains a number of sample runs.

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.