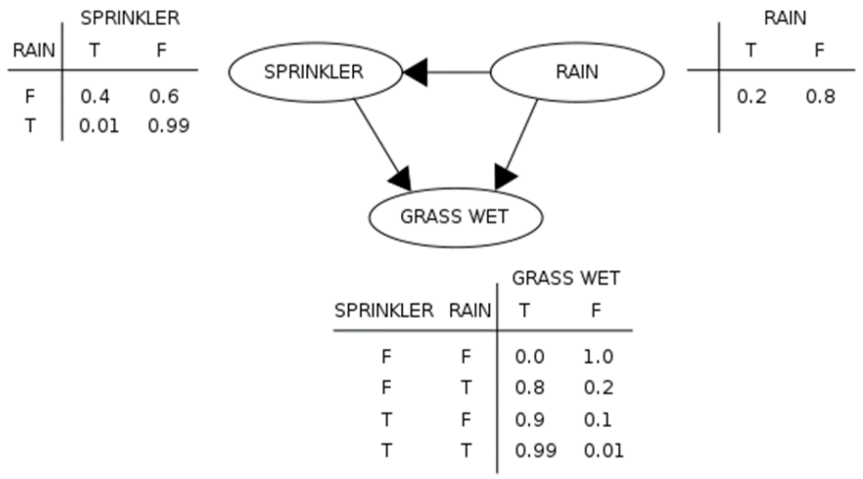
2. Data Structures and Algorithms

Background: A “factor” is a function over a set of variables. A factor maps each instantiation of the variables in a set to a non-negative number. For example, each of the three conditional probability tables in the Bayesian network below—with child, C and parents, $\mathbf{P}$—can be represented as a factor over $\mathrm{C} \cup \mathbf{P}$. We use capital letters (e.g., C) to represent random variables and bold capital letters (e.g., $\mathbf{P}$) to represent a set of variables. (For the purposes of this exercise, we only deal with discrete variables.) We use the lowercase letter $\mathbf{c}$ to represent some instantiation that the variable C can have. Likewise, we use the bold lowercase letter $\mathbf{p}$ to represent some instantiation of values for the variables in $\mathbf{P}$.

Given two sets of variables $\mathbf{X}$ and $\mathbf{Y}$ and the set of variables shared between them, $\boldsymbol{Z} = \boldsymbol{X} \cap \boldsymbol{Y}$, the instantiations $\mathbf{x}$ and $\mathbf{y}$ are consistent, if the instantiation of the shared variables $\mathbf{z}$ in $\mathbf{x}$ and $\mathbf{y}$ are the same.

Below is the classic Sprinkler Bayesian network:



	SPRINKLER	
RAIN	T	F
F	0.4	0.6
T	0.01	0.99

RAIN	
T	F
0.2	0.8

		GRASS WET	
SPRINKLER	RAIN	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

The factors for the network are:

Factor A over Rain, Sprinkler	Factor B over Rain	Factor C over Sprinkler, Rain, Grass Wet
$(\mathrm{F},\mathrm{T}) \rightarrow 0.4$	$(\mathrm{T}) \rightarrow 0.2$	$(\mathrm{F},\mathrm{F},\mathrm{T}) \rightarrow 0.0$ $(\mathrm{F},\mathrm{F},\mathrm{F}) \rightarrow 1.0$
$(\mathrm{F},\mathrm{F}) \rightarrow 0.6$	$(\mathrm{F}) \rightarrow 0.8$	$(\mathrm{F},\mathrm{T},\mathrm{T}) \rightarrow 0.8$ $(\mathrm{F},\mathrm{T},\mathrm{F}) \rightarrow 0.2$
$(\mathrm{T},\mathrm{T}) \rightarrow 0.01$		$(\mathrm{T},\mathrm{F},\mathrm{T}) \rightarrow 0.9$ $(\mathrm{T},\mathrm{F},\mathrm{F}) \rightarrow 0.1$
$(\mathrm{T},\mathrm{F}) \rightarrow 0.99$		$(\mathrm{T},\mathrm{T},\mathrm{T}) \rightarrow 0.99$ $(\mathrm{T},\mathrm{T},\mathrm{F}) \rightarrow 0.01$

Below are two operations that can be performed on factors to produce a new factor:

- **Summing-out operation**. Let f be a factor over variables $\mathbf{X}$ and let X be a variable in $\mathbf{X}$. The result of *summing out* variable X from factor f is another factor over variables $\mathbf{Y} = \mathbf{X} \setminus \{\mathrm{X}\}$ (the set of variables $\mathbf{X}$ with X removed), which is denoted by $\Sigma_{\mathrm{X}} f$. The value of applying $\Sigma_{\mathrm{X}} f$ to instantiation $\mathbf{y}$ is defined as

$$\left(\sum_{X} f\right)(\mathbf{y}) \stackrel{\text{def}}{=} \sum_{x} f(x, \mathbf{y})$$

Essentially, we sum up the probabilities of the variable instantiations that are consistent when disregarding the summed-out variable.

The result of summing out Sprinkler from Factor C is the following factor:

Factor over Rain, Grass Wet
$(\mathrm{F}, \mathrm{F}) \to 1.1$
$(\mathrm{F}, \mathrm{T}) \to 0.9$
$(\mathrm{T}, \mathrm{F}) \to 0.21$
$(\mathrm{T}, \mathrm{T}) \to 1.79$

- **Multiplying operation**. The result of *multiplying* factors $f_1(\mathbf{X})$ and $f_2(\mathbf{Y})$ is another factor over variables $\mathbf{Z} = \mathbf{X} \cup \mathbf{Y}$, denoted by $f_1 f_2$. The resulting factor of applying $f_1 f_2$ to the instantiation $\boldsymbol{z}$ is defined as

  $$(f_1 f_2)(\mathbf{z}) \stackrel{\text{def}}{=} f_1(\mathbf{x}) f_2(\mathbf{y})$$

  where $\mathbf{x}$ and $\mathbf{y}$ are consistent with $\boldsymbol{z}$.

  The result of multiplying Factor A and Factor B is the following factor:

  Factor over Rain, Sprinkler
  $(\mathrm{F}, \mathrm{T}) \to 0.32$
  $(\mathrm{F}, \mathrm{F}) \to 0.42$
  $(\mathrm{T}, \mathrm{T}) \to 0.002$
  $(\mathrm{T}, \mathrm{F}) \to 0.198$

Task: Implement a data structure to represent factors, like the ones described above, along with functions/methods to perform the summing-out and multiplying operations on them. Note that these operations form the core of various inference algorithms used by Bayesian networks and it is important that they are fast. Include unit tests to demonstrate the correctness of your functions.

DELIVERABLES:

1) All code used to implement factors along with unit tests.
2) A short memo that answers the following questions. Include any assumptions you may make.
   a. What are the time and space complexities of the summing-out and multiplying functions?
   b. What are the tradeoffs in terms of time, storage, etc. did you make while designing your data structure?