Cole Constantino
Cxc820
P2 Writeup

1) **a)** calling G.check_mode() in console returns true

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Fri Nov 15 14:59:40 2019

@author: coleconstantino
"""

from pgmpy.models import BayesianModel
from pgmpy.factors.discrete import TabularCPD

G = BayesianModel([('B', 'G'),( 'F', 'G')])
cpd_B = TabularCPD(variable = 'B', variable_card = 2, values = [[0.8,0.2]])
cpd_F = TabularCPD(variable = 'F', variable_card = 2, values = [[0.9,0.1]])
cpd_G = TabularCPD(variable = 'G', variable_card = 2, values = [[0.1,0.1,0.2,0.9],
                                                                [0.9,0.9,0.8,0.1]],
                                                                evidence = ['B','F'],
                                                                evidence_card = [2, 2])

G.add_cpds(cpd_B, cpd_F, cpd_G)
G.check_model()
```

b) P(F=0|G=0) = $\frac{F(G=0 \mid F=0)P(F=0)}{P(G=0)}$

c)

```
In [26]: runfile('/Users/c
Users/coleconstantino')
Finding Elimination Order:
484.44it/s]
Eliminating: B: 100%|
+------+----------+
| F    |   phi(F) |
+======+==========+
| F(1) |   0.7258 |
+------+----------+
| F(0) |   0.2742 |
+------+----------+
```

d)

1. P(F)
2. P(F|G =0)
3. P(F|B =0)

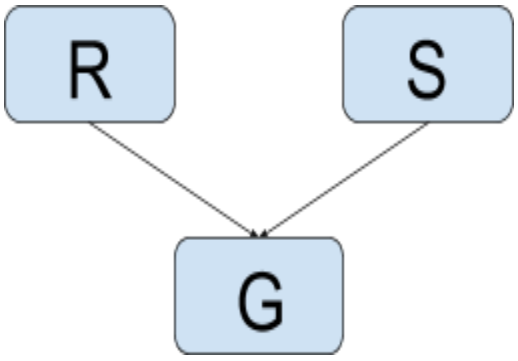
```
infer = VariableElimination(G)
#g_dist = infer.query(['F'], evidence={'G': '0'})
one_dist = infer.query(['F'])
two_dist = infer.query(['F'], evidence={'G': '0'})
three_dist = infer.query(['F'], evidence={'B': '0'})
print()
print(one_dist)
print()
print(two_dist)
print()
print(three_dist)
```

```
Finding Elimination Ord
Eliminating: G: 100%|
+------+----------+
| F    |   phi(F) |
+======+==========+
| F(1) |   0.9000 |
+------+----------+
| F(0) |   0.1000 |
+------+----------+

+------+----------+
| F    |   phi(F) |
+======+==========+
| F(1) |   0.7258 |
+------+----------+
| F(0) |   0.2742 |
+------+----------+

+------+----------+
| F    |   phi(F) |
+======+==========+
| F(1) |   0.9000 |
+------+----------+
| F(0) |   0.1000 |
+------+----------+
```

e) When we read the gauge and see its empty we believe the gauge is empty. Its only when we see the battery is dead that we realize that's why the gauge read zero. So the probability(our belief) of the guage being empty goes back down.

F) It would remain as .9 because it is independent.

2) a)



b)

P(R = 1) = .20
P(S = 1) = .10

P(G = 1| R = 0, S = 0) = 0
P(G = 1| R = 0, S = 1) = 0.95
P(G = 1| R = 1, S = 0) = 1
P(G = 1| R = 1, S = 1) = 1

c)

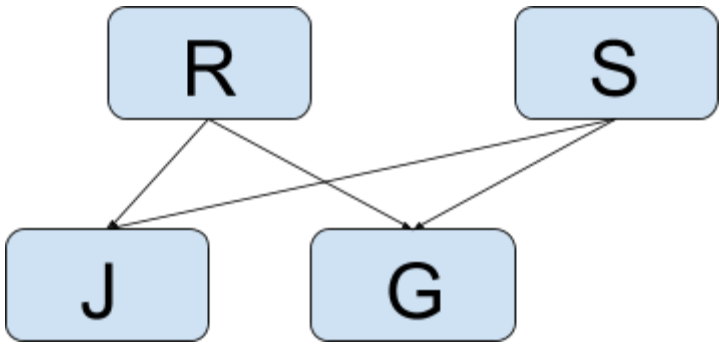
```
5
6 @author: coleconstantino
7 """
8
9 from pgmpy.inference import VariableElimination
10 from pgmpy.models import BayesianModel
11 from pgmpy.factors.discrete import TabularCPD
12
13 G = BayesianModel([('B', 'G'),( 'F', 'G')])
14 cpd_B = TabularCPD(variable = 'R', variable_card = 2, values = [[0.2,0.8]], state_names = {'B' : ['1','0']})
15
16
17 cpd_F = TabularCPD(variable = 'S', variable_card = 2, values = [[0.1,0.9]], state_names = {'F' : ['1','0']})
18
19
20 cpd_G = TabularCPD(variable = 'G', variable_card = 2, values = [[0,0.95,1,1],
21                                                                 [1,0.05,0,0]
22                                                                 ],
23                                                                 evidence = ['R','S'],
24                                                                 evidence_card = [2, 2],
25                                                                 state_names = {'G': ['1','0'],
26                                                                                'B' : ['1','0'],
27                                                                                'F' : ['1','0']})
28
29 G.add_cpds(cpd_B, cpd_F, cpd_G)
30
```

d)

```
4 one_dist = infer.query(['G'], evidence={'R': '0','S': '0'})
5 two_dist = infer.query(['G'], evidence={'R': '0','S': '1'})
6 three_dist = infer.query(['G'], evidence={'R': '1','S': '0'})
7 four_dist = infer.query(['G'], evidence={'R': '1','S': '1'})
```

```
0it [00:00, ?it/s]+-------+
| G    |   phi(G) |
+======+==========+
| G(1) |   1.0000 |
+------+----------+
| G(0) |   0.0000 |
+------+----------+

+------+----------+
| G    |   phi(G) |
+======+==========+
| G(1) |   1.0000 |
+------+----------+
| G(0) |   0.0000 |
+------+----------+

+------+----------+
| G    |   phi(G) |
+======+==========+
| G(1) |   0.9500 |
+------+----------+
| G(0) |   0.0500 |
+------+----------+

+------+----------+
| G    |   phi(G) |
+======+==========+
| G(1) |   0.0000 |
+------+----------+
| G(0) |   1.0000 |
+------+----------+
```

e)



(S really has nothing to do with j but kept it in there for simplicity's sake)

P(R = 1) = .20
P(S = 1) = .10

P(J = 1| R = 0, S = 0) = 0.15
P(J = 1| R = 0, S = 1) = 0.15
P(J = 1| R = 1, S = 0) = 1
P(J = 1| R = 1, S = 1) = 1

```
from pgmpy.inference import VariableElimination
from pgmpy.models import BayesianModel
from pgmpy.factors.discrete import TabularCPD

G = BayesianModel([('R', 'G'),( 'S', 'G'),('R','J'),('S','J')])
cpd_B = TabularCPD(variable = 'R', variable_card = 2, values = [[0.2,0.8]], state_names = {'R' : ['1','0']})


cpd_F = TabularCPD(variable = 'S', variable_card = 2, values = [[0.1,0.9]], state_names = {'S' : ['1','0']})


cpd_G = TabularCPD(variable = 'G', variable_card = 2, values = [[0,0.95,1,1],
                                                                [1,0.05,0,0]
                                                                ],
                                                                evidence = ['R','S'],
                                                                evidence_card = [2, 2],
                                                                state_names = {'G': ['1','0'],
                                                                               'R' : ['1','0'],
                                                                               'S' : ['1','0']})

cpd_J = TabularCPD(variable = 'J', variable_card = 2, values = [[0.15,0.15,1,1],
                                                                [0.85,0.85,0,0]
                                                                ],
                                                                evidence = ['R','S'],
                                                                evidence_card = [2, 2],
                                                                state_names = {'J': ['1','0'],
                                                                               'R' : ['1','0'],
                                                                               'S' : ['1','0']})


G.add_cpds(cpd_B, cpd_F, cpd_G, cpd_J)


infer = VariableElimination(G)

one_dist = infer.query(['J'], evidence={'R': '0','S': '0'})
two_dist = infer.query(['J'], evidence={'R': '0','S': '1'})
three_dist = infer.query(['J'], evidence={'R': '1','S': '0'})
four_dist = infer.query(['J'], evidence={'R': '1','S': '1'})

print(one_dist)
print()
print(two_dist)
print()
print(three_dist)
print()
print(four_dist)
```

```
Console 1/A
Eliminating: G: 100%|
Finding Elimination Order:
Eliminating: G: 100%|
Finding Elimination Order:
Eliminating: G: 100%|
Finding Elimination Order:
Eliminating: G: 100%|
+----------+
| J    |   phi(J) |
+======+==========+
| J(1) |   1.0000 |
+------+----------+
| J(0) |   0.0000 |
+------+----------+

+------+----------+
| J    |   phi(J) |
+======+==========+
| J(1) |   1.0000 |
+------+----------+
| J(0) |   0.0000 |
+------+----------+

+------+----------+
| J    |   phi(J) |
+======+==========+
| J(1) |   0.1500 |
+------+----------+
| J(0) |   0.8500 |
+------+----------+

+------+----------+
| J    |   phi(J) |
+======+==========+
| J(1) |   0.1500 |
+------+----------+
| J(0) |   0.8500 |
+------+----------+
```

f) P(S = 1| J = 1, G = 1)

final_dist = infer.query(['S'], evidence={'J': '1','G': '1'}) ⇒

```
+----------+
| S    |   phi(S) |
+======+==========+
| S(1) |   0.0969 |
+------+----------+
| S(0) |   0.9031 |
+------+----------+
```

g) Because she/we now know that it definitely rained last night which makes sprinkler not matter.

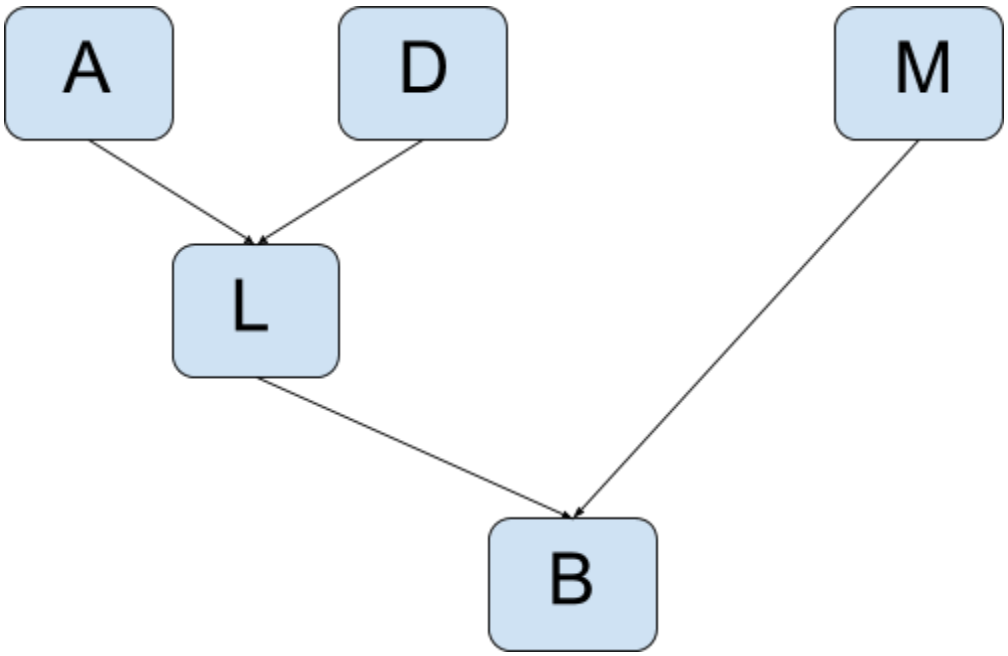
EXTRA CREDIT:

You are trying to determine if your fantasy adventure character can defeat the Best Boss in the game.

How to beat the boss:

- If your combat level(L) is above 75, or your magic level(M) is above 90
- It is still possible with lower requirements but exponentially harder
- Your combat level is 75 or greater if your Attack Level(A) and Defense Level(D) are both 75 or greater

If a variable meets a level requirement its value is 1 else 0



P(A = 1) = .20
P(D = 1) = .30
P(M = 1) = .10

P(L= 1| A = 0, D = 0) = 0
P(L = 1| A = 0, D = 1) = 0
P(L = 1| A = 1, D = 0) = 0

P(L = 1| A = 1, D = 1) = 1

P(B= 1| L = 0, M = 0) = .01
P(B = 1| L = 0, M = 1) = .45
P(B = 1| L = 1, M = 0) = .28
P(B = 1| L = 1, M = 1) = .95

```
8
9 from pgmpy.inference import VariableElimination
10 from pgmpy.models import BayesianModel
11 from pgmpy.factors.discrete import TabularCPD
12
13 G = BayesianModel([('A', 'L'),( 'D', 'L'),('L', 'B'),('M', 'B')])
14
15 cpd_A = TabularCPD(variable = 'A', variable_card = 2, values = [[0.2,0.8]], state_names = {'A' : ['1','0']})
16 cpd_D = TabularCPD(variable = 'D', variable_card = 2, values = [[0.3,0.7]], state_names = {'D' : ['1','0']})
17 cpd_M = TabularCPD(variable = 'M', variable_card = 2, values = [[0.1,0.9]], state_names = {'M' : ['1','0']})
18
19 #combat Level
20 cpd_L = TabularCPD(variable = 'L', variable_card = 2, values = [[0,0,0,1],
21                                                                 [1,1,1,0]
22                                                                 ],
23                                                                 evidence = ['A','D'],
24                                                                 evidence_card = [2, 2],
25                                                                 state_names = {'L': ['1','0'],
26                                                                                'A' : ['1','0'],
27                                                                                'D' : ['1','0']})
28 #Boss capabilites
29 cpd_B = TabularCPD(variable = 'B', variable_card = 2, values = [[0.01,0.45,0.28,0.95],
30                                                                 [0.99,0.55,0.72,0.05]
31                                                                 ],
32                                                                 evidence = ['L','M'],
33                                                                 evidence_card = [2, 2],
34                                                                 state_names = {'B': ['1','0'],
35                                                                                'L' : ['1','0'],
36                                                                                'M' : ['1','0']})
37
38 G.add_cpds(cpd_A, cpd_D, cpd_M, cpd_L, cpd_B)
39
```

```
0
1 infer = VariableElimination(G)
2 l_dist = infer.query(['L'])
3 b_dist = infer.query(['B'])
4 ex1_dist = infer.query(['B'], evidence={'L': '0','M': '0'})
5 ex2_dist = infer.query(['B'], evidence={'L': '0','M': '1'})
5 ex3_dist = infer.query(['B'], evidence={'L': '1','M': '0'})
7 ex4_dist = infer.query(['B'], evidence={'L': '1','M': '1'})
8
```

```
| L    |   phi(L) |
+======+==========+
| L(1) |   0.5600 |
+------+----------+
| L(0) |   0.4400 |
+------+----------+
```

```
+------+----------+
| B    |   phi(B) |
+======+==========+
| B(1) |   0.6159 |
+------+----------+
| B(0) |   0.3841 |
+------+----------+
```

```
+------+----------+
| B    |   phi(B) |
+======+==========+
| B(1) |   0.9500 |
+------+----------+
| B(0) |   0.0500 |
+------+----------+
```

```
+------+----------+
| B    |   phi(B) |
+======+==========+
| B(1) |   0.2800 |
+------+----------+
| B(0) |   0.7200 |
+------+----------+
```

```
+------+----------+
| B    |   phi(B) |
+======+==========+
| B(1) |   0.4500 |
+------+----------+
| B(0) |   0.5500 |
+------+----------+
```

```
+------+----------+
| B    |   phi(B) |
+======+==========+
| B(1) |   0.0100 |
+------+----------+
| B(0) |   0.9900 |
```