CSC 421
Artificial Intelligence
Assignment 3

Parm Johal
V00787710

Q1)

(p | q | -r) & ((-r | q | p) -> ((r | q) & -q & -p))

(p | q | -r) & (-(-r | q | p) | ((r | q) & -q & -p))

(p | q | -r) & ((r & -q & -p) | ((r | q) & -q & -p))

(p | q | -r) & ((r & -q & -p) | (r | q)) & ((r & -q & -p) | -q) & ((r & -q & -p) | -p)

(p | q | -r) & (r | (r | q)) & (-q | (r | q)) & (-p | (r | q)) & (r | -q) & (-q | -q) & (-p | -q) & (r | -p) & (-q | -p) & (-p | -p)

(p | q | -r) & (r | q) & (-q | r | q) & (-p | r | q) & (r |-q) & (-q) & (-p | -q) & (r | -p) & (-q | -p) & (-p)

{p, q, -r}
{r, q}
{-q, r, q}
{-p, r, q}
{r, -q}
{-q}
{-p, -q}
{r, -p}
{-q, -p}
{-p}

Resolution:

1. {p, q, -r}
2. {r, q}
3. {-q, r, q}
4. {-p, r, q}
5. {r, -q}
6. {-q}
7. {-p, -q}
8. {r, -p}
9. {-q, -p}
10. {-p}
11. {p, -r} 1,6
12. {r} 2,6
13. {-p, r} 4,6
14. {q, -r} 1,10
15. {q} 12,14
16. {} 6,15

Q2)

Every horse can outrun every dog.

Some greyhounds can outrun every rabbit.

Show that every horse can outrun every rabbit.

FOL:

∀x.∀y. (Horse(x) & Dog(y) ⇒ Faster(x, y))

∃y. (Greyhound(y) & ∀z.(Rabbit(z) ⇒ Faster(y, z)))

∀y. (Greyhound(y) ⇒ Dog(y)) **(background knowledge)**

∀x.∀y.∀z. (Faster(x, y) & Faster(y, z) ⇒ Faster(x, z)) **(background knowledge)**

-∀x.∀y. (Horse(x) & Rabbit(y) ⇒ Faster(x, y)) **(Negated Conclusion)**

Clausal Form (INSEADO):

<u>*∀x. ∀y. (Horse(x) & Dog(y) ⇒ Faster(x, y)):*</u>

∀x.∀y. (-(Horse(x) & Dog(y)) | Faster(x, y))

∀x.∀y. (-Horse(x) | -Dog(y) | Faster(x, y))

-Horse(x) | -Dog(y) | Faster(x, y)

{-Horse(x), -Dog(y), Faster(x, y)}
∃y. (Greyhound(y) & ∀z.(Rabbit(z) ⇒ Faster(y, z))):
Greyhound(Rocky) & (-Rabbit(z) | Faster(Rocky, z))
{Greyhound(Rocky)}
{-Rabbit(z), Faster(Rocky, z)}

∀y. (Greyhound(y) ⇒ Dog(y)):
∀y. (-Greyhound(y) | Dog(y))
-Greyhound(y) | Dog(y)
{-Greyhound(y), Dog(y)}

∀x. ∀y. ∀z. (Faster(x, y) & Faster(y, z) ⇒ Faster(x, z)):
∀x.∀y.∀z. (-(Faster(x, y) & Faster(y, z)) | Faster(x, z))
∀x.∀y.∀z. (-Faster(x, y) | -Faster(y, z) | Faster(x, z))
-Faster(x, y) | -Faster(y, z) | Faster(x, z)
{-Faster(x, y), -Faster(y, z), Faster(x, z)}

- ∀x. ∀y. (Horse(x) & Rabbit(y) ⇒ Faster(x, y)):
-∀x.∀y. (-(Horse(x) & Rabbit(y)) | Faster(x, y))
-∀x.∀y. (-Horse(x) | -Rabbit(y) | Faster(x, y))
-∀x.∀y. (-Horse(x) | -Rabbit(y) | Faster(x, y))
∃x.∃y. (Horse(x) & Rabbit(y) & -Faster(x, y))
Horse(Boxer) & Rabbit(Bugs) & -Faster(Boxer, Bugs)
{Horse(Boxer)}
{Rabbit(Bugs)}
{-Faster(Boxer, Bugs)}

Resolution:

1. {-Horse(x_1), -Dog(y_1), Faster(x_1, y_1)}
2. {-Rabbit(z_1), Faster(Rocky, z_1)}
3. {-Greyhound(y_2), Dog(y_2)}
4. {-Faster(x_2, y_3), -Faster(y_3, z_2), Faster(x_2, z_2)}
5. {Horse(Boxer)}
6. {Rabbit(Bugs)}
7. {-Faster(Boxer, Bugs)}
8. {-Dog(y_1), Faster(x_1, y_1)} 1,5 mgu = {$x_1 \leftarrow$ Boxer}
9. {Faster(Rocky, Bugs)} 2,6 mgu = {$z_1 \leftarrow$ Bugs}

10. {}   7,9   mgu = {Boxer ← Rocky}

Q3)

All hummingbirds are richly colored.
No large birds live on honey.
Birds that do not live on honey are dull in color.
Conclusion: All hummingbirds are small.

FOL:

$\forall$x.(Hummingbird(x) ⇒ Color(rich))
$\forall$x.(Bird(x) & Large(x) ⇒ -Honey(x))
$\forall$x.(Bird(x) & -Honey(x) ⇒ Color(dull))
$\forall$x.(Hummingbird(x) ⇒ Bird(x))
$\forall$x.(Hummingbird(x) ⇒ Small(x))   **(Conclusion)**

Prover 9 Format:

all x(hummingbird(x) -> richcolor(x)).
all x(bird(x) & large(x) -> -honey(x)).
all x(bird(x) & -honey(x) -> -richcolor(x)).
all x(hummingbird(x) -> bird(x)).
all x(hummingbird(x) -> -large(x)).

(**NOTE:** Following proof formed from Prover 9)

============================== PROOF
=================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.16) seconds.
% Length of proof is 18.
% Level of proof is 5.
% Maximum clause weight is 0.
% Given clauses 0.

1 (all x (hummingbird(x) -> richcolor(x))) # label(non_clause). [assumption].
2 (all x (bird(x) & large(x) -> -honey(x))) # label(non_clause). [assumption].
3 (all x (bird(x) & -honey(x) -> -richcolor(x))) # label(non_clause). [assumption].

4 (all x (hummingbird(x) -> bird(x))) # label(non_clause). [assumption].
5 (all x (hummingbird(x) -> -large(x))) # label(non_clause) # label(goal). [goal].
6 hummingbird(c1). [deny(5)].
7 -hummingbird(x) | richcolor(x). [clausify(1)].
8 -hummingbird(x) | bird(x). [clausify(4)].
9 bird(c1). [resolve(6,a,8,a)].
10 -bird(x) | -large(x) | -honey(x). [clausify(2)].
11 -bird(x) | honey(x) | -richcolor(x). [clausify(3)].
12 -large(c1) | -honey(c1). [resolve(9,a,10,a)].
13 large(c1). [deny(5)].
14 honey(c1) | -richcolor(c1). [resolve(9,a,11,a)].
15 richcolor(c1). [resolve(6,a,7,a)].
16 honey(c1). [resolve(14,b,15,a)].
17 -honey(c1). [resolve(12,a,13,a)].
18 $F. [resolve(16,a,17,a)].

============================== end of proof ==========================

Q4)

My gardener is well worth listening to on military subjects.
No one can remember the battle of Waterloo, unless he is very old.
Nobody is really worth listening to on military subjects, unless he can remember the battle of Waterloo.
Conclusion: My gardener is very old.

FOL:

∃x.(myGardener(x) & listenTo(x))
∀x.(old(x) ⇔ remember(x))
∀x.(remembery(x) ⇔ listenTo(x))
∃x.(myGardener(x) & old(x)) **(Conclusion)**

Prover 9 Format:

exists x(myGardener(x) & listenTo(x)).
all x(old(x) <-> remember(x)).
all x(remember(x) <-> listenTo(x)).
exists x(myGardener(x) & old(x)). **(Conclusion)**

```
================================ PROOF
==================================

% -------- Comments from original proof --------
% Proof 1 at 0.03 (+ 0.11) seconds.
% Length of proof is 13.
% Level of proof is 4.
% Maximum clause weight is 0.
% Given clauses 0.

1 (exists x (myGardener(x) & listenTo(x))) # label(non_clause).  [assumption].
2 (all x (old(x) <-> remember(x))) # label(non_clause).  [assumption].
3 (all x (remember(x) <-> listenTo(x))) # label(non_clause).  [assumption].
4 (exists x (myGardener(x) & old(x))) # label(non_clause) # label(goal).  [goal].
5 -myGardener(x) | -old(x).  [deny(4)].
6 myGardener(c1).  [clausify(1)].
7 remember(x) | -listenTo(x).  [clausify(3)].
8 listenTo(c1).  [clausify(1)].
10 old(x) | -remember(x).  [clausify(2)].
12 -old(c1).  [resolve(5,a,6,a)].
13 -remember(c1).  [resolve(12,a,10,a)].
14 remember(c1).  [resolve(7,b,8,a)].
15 $F.  [resolve(13,a,14,a)].

================================ end of proof
==========================
```

Q5)

$P(p_{13}|b_{12},b_{21})$

$= \alpha \sum_{p22}\sum_{p31} P(b_{12} | p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(p_{31})$

$= \alpha[P(b_{12} | p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(p_{31}) +$
$P(b_{12} | p_{13}, -p_{22}) * P(b_{21} | p_{31}, -p_{22}) * P(p_{13}) * P(-p_{22}) * P(p_{31}) +$
$P(b_{12} | p_{13}, p_{22}) * P(b_{21} | -p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(-p_{31}) +$
$P(b_{12} | p_{13}, -p_{22}) * P(b_{21} | -p_{31}, -p_{22}) * P(p_{13}) * P(-p_{22}) * P(-p_{31})]$

$= \alpha[(1 * 1 * 0.1 * 0.1 * 0.1) + (1 * 1 * 0.1 * 0.9 * 0.1) + (1 * 1 * 0.1 * 0.1 * 0.9) + 0]$

$= \alpha * 0.019$

$P(-p_{13}|b_{12},b_{21})$

$= \alpha \sum_{p22}\sum_{p31} P(b_{12} | -p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(-p_{13}) * P(p_{22}) * P(p_{31})$

$= \alpha[P(b_{12} | -p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(-p_{13}) * P(p_{22}) * P(p_{31}) +$

$P(b_{12} | -p_{13}, -p_{22}) * P(b_{21} | p_{31}, -p_{22}) * P(-p_{13}) * P(-p_{22}) * P(p_{31}) +$

$P(b_{12} | -p_{13}, p_{22}) * P(b_{21} | -p_{31}, p_{22}) * P(-p_{13}) * P(p_{22}) * P(-p_{31}) +$

$P(b_{12} | -p_{13}, -p_{22}) * P(b_{21} | -p_{31}, -p_{22}) * P(-p_{13}) * P(-p_{22}) * P(-p_{31})]$

$= \alpha[(1 * 1 * 0.9 * 0.1 * 0.1) + 0 + (1 * 1 * 0.9 * 0.1 * 0.9) + 0]$

$= \alpha * 0.09$

$\alpha = 1/(0.019 + 0.09)$

$\therefore$ $P(p_{13}|b_{12},b_{21})$ = 17.4% & $P(-p_{13}|b_{12},b_{21})$ = 82.6%

$P(p_{31}|b_{12},b_{21})$

$= \alpha \sum_{p22}\sum_{p13} P(b_{12} | p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(p_{31})$

$= \alpha[P(b_{12} | p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(p_{31}) +$

$P(b_{12} | p_{13}, -p_{22}) * P(b_{21} | p_{31}, -p_{22}) * P(p_{13}) * P(-p_{22}) * P(p_{31}) +$

$P(b_{12} | -p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(-p_{13}) * P(p_{22}) * P(p_{31}) +$

$P(b_{12} | -p_{13}, -p_{22}) * P(b_{21} | p_{31}, -p_{22}) * P(-p_{13}) * P(-p_{22}) * P(p_{31})]$

$= \alpha[(1 * 1 * 0.1 * 0.1 * 0.1) + (1 * 1 * 0.1 * 0.9 * 0.1) + (1 * 1 * 0.9 * 0.1 * 0.1) + 0]$

$= \alpha * 0.019$

$P(-p_{31}|b_{12},b_{21})$

$= \alpha \sum_{p22}\sum_{p13} P(b_{12} | p_{13}, p_{22}) * P(b_{21} | -p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(-p_{31})$

$= \alpha[P(b_{12} | p_{13}, p_{22}) * P(b_{21} | -p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(-p_{31}) +$

$P(b_{12} | p_{13}, -p_{22}) * P(b_{21} | -p_{31}, -p_{22}) * P(p_{13}) * P(-p_{22}) * P(-p_{31}) +$

$P(b_{12} | -p_{13}, p_{22}) * P(b_{21} | -p_{31}, p_{22}) * P(-p_{13}) * P(p_{22}) * P(-p_{31}) +$

$P(b_{12} | -p_{13}, -p_{22}) * P(b_{21} | -p_{31}, -p_{22}) * P(-p_{13}) * P(-p_{22}) * P(-p_{31})]$

$= \alpha[(1 * 1 * 0.1 * 0.1 * 0.9) + 0 + (1 * 1 * 0.9 * 0.1 * 0.9) + 0]$

$= \alpha * 0.09$

$\alpha = 1/(0.019 + 0.09)$

$\therefore$ $P(p_{31}|b_{12},b_{21})$ = 17.4% & $P(-p_{31}|b_{12},b_{21})$ = 82.6%

$P(p_{22}|b_{12},b_{21})$
$= \alpha \sum_{p13}\sum_{p31} P(b_{12} | p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(p_{31})$
$= \alpha[P(b_{12} | p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(p_{31}) +$
$P(b_{12} | -p_{13}, p_{22}) * P(b_{21} | p_{31}, p_{22}) * P(-p_{13}) * P(p_{22}) * P(p_{31}) +$
$P(b_{12} | p_{13}, p_{22}) * P(b_{21} | -p_{31}, p_{22}) * P(p_{13}) * P(p_{22}) * P(-p_{31}) +$
$P(b_{12} | -p_{13}, p_{22}) * P(b_{21} | -p_{31}, p_{22}) * P(-p_{13}) * P(p_{22}) * P(-p_{31})$
$= \alpha[(1 * 1 * 0.1 * 0.1 * 0.1) + (1 * 1 * 0.9 * 0.1 * 0.1) + (1 * 1 * 0.1 * 0.1 * 0.9) +$
$(1 * 1 * 0.9 * 0.1 * 0.9)]$
$= \alpha * 0.1$

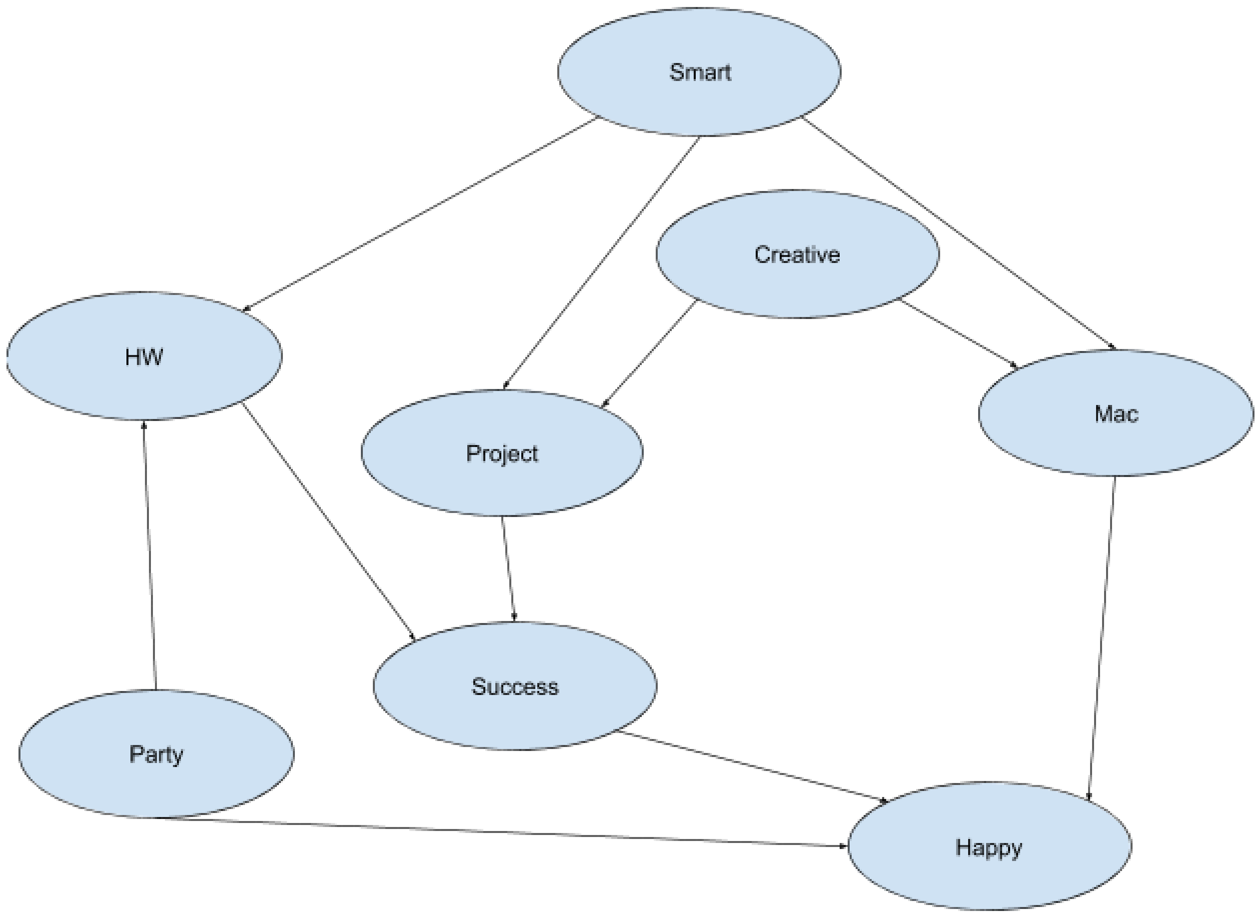
$P(-p_{22}|b_{12},b_{21})$
$= \alpha \sum_{p13}\sum_{p31} P(b_{12} | p_{13}, -p_{22}) * P(b_{21} | p_{31}, -p_{22}) * P(p_{13}) * P(-p_{22}) * P(p_{31})$
$= \alpha[P(b_{12} | p_{13}, -p_{22}) * P(b_{21} | p_{31}, -p_{22}) * P(p_{13}) * P(-p_{22}) * P(p_{31}) +$
$P(b_{12} | -p_{13}, -p_{22}) * P(b_{21} | p_{31}, -p_{22}) * P(-p_{13}) * P(-p_{22}) * P(p_{31}) +$
$P(b_{12} | p_{13}, -p_{22}) * P(b_{21} | -p_{31}, -p_{22}) * P(p_{13}) * P(-p_{22}) * P(-p_{31}) +$
$P(b_{12} | -p_{13}, -p_{22}) * P(b_{21} | -p_{31}, -p_{22}) * P(-p_{13}) * P(-p_{22}) * P(-p_{31})$
$= \alpha[(1 * 1 * 0.1 * 0.9 * 0.1) + 0 + 0 + 0]$
$= \alpha * 0.009$

$\alpha = 1/(0.1 + 0.009)$
$\therefore P(p_{22}|b_{12},b_{21}) = 91.7\%$ & $P(-p_{22}|b_{12},b_{21}) = 8.3\%$

A probabilistic agent will never choose to go to [2,2].
A logical agent would choose either squares [1,3], [2,2], [3,1] because they look the same, each with an equal chance of being chosen (1/3). By doing that, the agent will die with a chance of about 1/3 if [2,2] is chosen.

Q6)

1.)



2.) See Excel Sheet.

3.)

P (Happy | Party, Smart, -Creative)

= $\alpha \sum_{HW}\sum_{Project}\sum_{Success}\sum_{Mac}$P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*
P(HW | Party,Smart)*P(Success | Project,HW)*P(Mac | -Creative,Smart)*
P(Happy | Party,Success,Mac)

= α[P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | Project,HW)*P(Mac | -Creative,Smart)*P(Happy | Party,Success,Mac)
+

P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | Project,HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | Project,HW)*P(Mac | -Creative,Smart)*P(Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | Project,HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,-Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | -Project,HW)*P(Mac | -Creative,Smart)*P(Happy | Party,Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | -Project,HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | -Project,HW)*P(Mac | -Creative,Smart)*P(Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | -Project,HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,-Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | Project,-HW)*P(Mac | -Creative,Smart)*P(Happy | Party,Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | Project,-HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | Project,-HW)*P(Mac | -Creative,Smart)*P(Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | Project,-HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,-Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | -Project,-HW)*P(Mac | -Creative,Smart)*P(Happy | Party,Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | -Project,-HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,Success,-Mac)
+

P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | -Project,-HW)*P(Mac | -Creative,Smart)*P(Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | -Project,-HW)*P(-Mac | -Creative,Smart)*P(Happy | Party,-Success,-Mac)]

P (-Happy | Party, Smart, -Creative)

= $\alpha \sum_{HW}\sum_{Project}\sum_{Success}\sum_{Mac}$P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*
P(HW | Party,Smart)*P(Success | Project,HW)*P(Mac | -Creative,Smart)*
P(-Happy | Party,Success,Mac)

= α[P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | Project,HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | Project,HW)*P(-Mac | -Creative,Smart)*P(-Happy | Party,Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | Project,HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | Project,HW)*P(-Mac | -Creative,Smart)*P(-Happy | Party,-Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | -Project,HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(Success | -Project,HW)*P(-Mac | -Creative,Smart)*P(-Happy | Party,Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | -Project,HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(HW | Party,Smart)*
P(-Success | -Project,HW)*P(-Mac | -Creative,Smart)*P(-Happy | Party,-Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | Project,-HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,Success,Mac)
+

P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | Project,-HW)*P(-Mac | -Creative,Smart)*P(-Happy | Party,Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | Project,-HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | Project,-HW)*P(-Mac | -Creative,Smart)*P(-Happy | Party,-Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | -Project,-HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(Success | -Project,-HW)*P(-Mac | -Creative,Smart)*P(-Happy | Party,Success,-Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | -Project,-HW)*P(Mac | -Creative,Smart)*P(-Happy | Party,-Success,Mac)
+
P(-Creative)*P(Smart)*P(Party)*P(-Project | -Creative,Smart)*P(-HW | Party,Smart)*
P(-Success | -Project,-HW)*P(-Mac | -Creative,Smart)*P(-Happy |
Party,-Success,-Mac)]

α = 1/(P (Happy | Party, Smart, -Creative) + P (-Happy | Party, Smart, -Creative))
No need to plug in the numbers and compute the sums and alpha. That would be too tedious.
P(Happy = T | Party = T, Smart = T, Creative = F) = 0.6922 (using the tool)

(**NOTE:** The following questions are answered using the AIspace tool specified by the assignment instructions)

4.) P(Happy | Smart,Creative) = 0.58155

5.) P(Happy | -Party,HW,Project) = 0.32045

6.) P(Happy | Mac) = 0.56271

7.) P(Party | Smart) = 0.6022

8.) P(Party | Smart,Happy) = 0.79264