CS-171, Intro to A.I. — Final Exam — Winter Quarter, 2012

NAME AND EMAIL ADDRESS: ____

YOUR ID: ________ ID TO RIGHT: ________ ROW: _____ NO. FROM RIGHT: _____

The exam will begin on the next page. Please, do not turn the page until told.

When you are told to begin the exam, please check first to make sure that you have all 11 pages, as numbered 1-11 in the bottom-left corner of each page.

The exam is closed-notes, closed-book. No calculators, cell phones, electronics.

Please clear your desk entirely, except for pen, pencil, eraser, an optional blank piece of paper (for optional scratch pad use), and an optional water bottle.
Please turn off all cell phones now.

This page summarizes the points available for each question so you can plan your time.

1. (10 pts total) Decision Tree Classifier Learning.

2. (5 pts total, -1 pt each wrong answer, but not negative) Search Properties.

3. (10 pts total) Naïve Bayes Classifier Learning.

4. (10 pts total, 1 pt each) Bayesian Networks.

5. (10 points total, 2 pts each) Constraint Satisfaction Problems.

6. (5 pts total, -1 for each error, but not negative) Alpha-Beta Pruning.

7. (10 pts total, -2 for each error, but not negative) Conversion to CNF.

8. (10 pts total, -2 for each error, but not negative) Resolution Theorem Proving.

9. (10 pts total, 2 pts each) State-Space Search.

10. (8 pts total, 1 pt each) Puzzle-Solving.

11. (2 pts total, 1 pt each) Heuristics.

12. (10 pts total, 2 pts each) English to FOL Conversion.

The Exam is printed on both sides to save trees! Work both sides of each page!

See Section 18.3.

1. (10 pts total) Decision Tree Classifier Learning. You are a robot in a lumber yard, and must learn to discriminate Oak wood from Pine wood. You choose to learn a Decision Tree classifier. You are given the following examples:

Example	Density	Grain	Hardness	Class
Example #1	Heavy	Small	Hard	Oak
Example #2	Heavy	Large	Hard	Oak
Example #3	Heavy	Small	Hard	Oak
Example #4	Light	Large	Soft	Oak
Example #5	Light	Large	Hard	Pine
Example #6	Heavy	Small	Soft	Pine
Example #7	Heavy	Large	Soft	Pine
Example #8	Heavy	Small	Soft	Pine

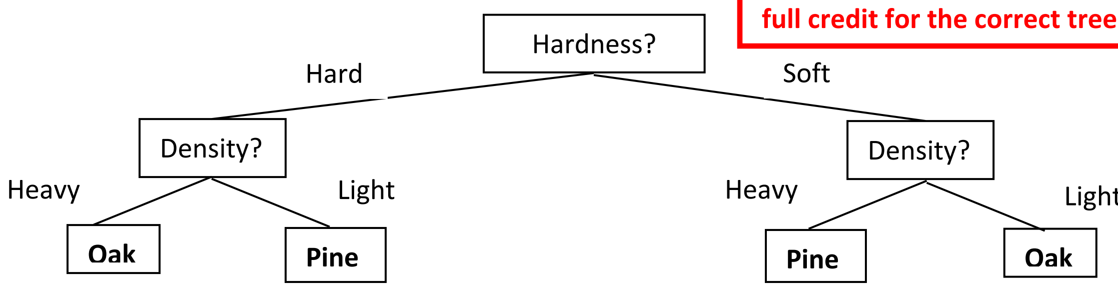
If root is Density:
Heavy = OOOPPP, Light = OP
If root is Grain:
Small = OOPP, Large = OOPP
If root is Hardness:
Hard = OOOP, Soft = OPPP
(O = Oak, P = Pine)

1a. (2 pts) Which attribute would information gain choose as the root of the tree?

______Hardness__________________.

1b. (4 pts) Draw the decision tree that would be constructed by recursively applying information gain to select roots of sub-trees, as in the

Half credit for the correct root; half credit for wrong root but correct classification; full credit for the correct tree.



Classify these new examples as Oak or Pine using your decision tree above.
1c. (2 pts) What class is [Density=Light, Grain=Small, Hardness=Hard]? __Pine__
1d. (2 pts) What class is [Density=Light, Grain=Small, Hardness=Soft]? __Oak__

Full credit if your answers are right for the tree you drew, even if the tree itself is wrong.

2. (5 pts total, -1 pt each wrong answer, but not negative) Search Properties. Fill in the values of the four evaluation criteria for each search strategy shown. Assume a tree search where b is the finite branching factor; d is the depth to the shallowest goal node; m is the maximum depth of the search tree; C* is the cost of the optimal solution; step costs are identical and equal to some positive ε; and in Bidirectional search both directions use breadth-first search.

Note that these conditions satisfy all of the footnotes of Fig. 3.21 in your book.

See Figure 3.21.

Criterion	Complete?	Time complexity	Space complexity	Optimal?
Breadth-First	Yes	O(b^d)	O(b^d)	Yes
Uniform-Cost	Yes	O(b^(1+floor(C*/ε))) O(b^(d+1)) also OK	O(b^(1+floor(C*/ε))) O(b^(d+1)) also OK	Yes
Depth-First	No	O(b^m)	O(bm)	No
Iterative Deepening	Yes	O(b^d)	O(bd)	Yes
Bidirectional (if applicable)	Yes	O(b^(d/2))	O(b^(d/2))	Yes

See Sections 13. 5.2 and 20.2.2.

3. (10 pts total) Naïve Bayes Classifier Learning. You are a robot in an animal shelter, and must learn to discriminate Dogs from Cats. You choose to learn a Naïve Bayes classifier. You are given the following (noisy) examples:

Example	Sound	Fur	Color	Class
Example #1	Meow	Coarse	Brown	Dog
Example #2	Bark	Fine	Brown	Dog
Example #3	Bark	Coarse	Black	Dog
Example #4	Bark	Coarse	Black	Dog
Example #5	Meow	Fine	Brown	Cat
Example #6	Meow	Coarse	Black	Cat
Example #7	Bark	Fine	Black	Cat
Example #8	Meow	Fine	Brown	Cat

Recall that Baye's rule allows you to rewrite the conditional probability of the class given the attributes as the conditional probability of the attributes given the class. As usual, α is a normalizing constant that makes the probabilities sum to one.

P(Class | Sound, Fur, Color) = α P(Sound, Fur, Color | Class) P(Class)

3a. (2 pts) Now assume that the attributes (Sound, Fur, and Color) are conditionally independent given the Class. Rewrite the expression above, using this assumption of conditional independence (i.e., rewrite it as a Naïve Bayes Classifier expression).

α P(Sound, Fur, Color | Class) P(Class) =
α P(Sound | Class) P(Fur | Class) P(Color | Class) P(Class)

3b. (4 pts total; -1 for each wrong answer, but not negative) Fill in numerical values for the following expressions. Leave your answers as common fractions (e.g., 1/4, 3/5).

P(Dog)= 1/2 P(Cat)= 1/2

P(Sound=Meow | Class=Dog)= 1/4 P(Sound=Meow | Class=Cat)= 3/4

P(Sound=Bark | Class=Dog)= 3/4 P(Sound=Bark | Class=Cat)= 1/4

P(Fur=Coarse | Class=Dog)= 3/4 P(Fur=Coarse | Class=Cat)= 1/4

P(Fur=Fine | Class=Dog)= 1/4 P(Fur=Fine | Class=Cat)= 3/4

P(Color=Brown | Class=Dog)= 1/2 P(Color=Brown | Class=Cat)= 1/2

P(Color=Black | Class=Dog)= 1/2 P(Color=Black | Class=Cat)= 1/2

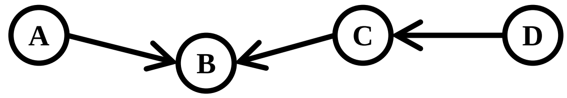
3c. (2 pt each) Consider a new example (Sound=Bark ^ Fur=Coarse ^ Color=Brown). Write these class probabilities as the product of α and common fractions from above.

P(Class=Dog | Sound=Bark ^ Fur=Coarse ^ Color=Brown) = α(3/4)(3/4)(1/2)(1/2)=9/10
P(Class=Cat | Sound=Bark ^ Fur=Coarse ^ Color=Brown) = α(1/4)(1/4)(1/2)(1/2)=1/10

See Section 14.2.

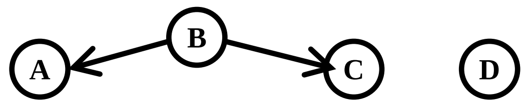
4. (10 pts total, 1 pt each) Bayesian Networks.
Draw the Bayesian Network that corresponds to the conditional probability equation.

4a. P(B|A,C) P(A) P(C|D) P(D)

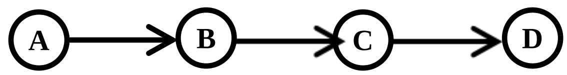


4b. P(A) P(B) P(C) P(D)

4c. P(A|B) P(C|B) P(B) P(D)

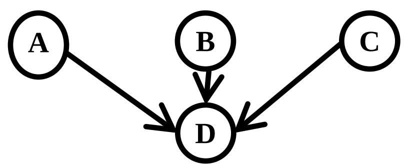


4d. P(D|C) P(C|B) P(B|A) P(A)

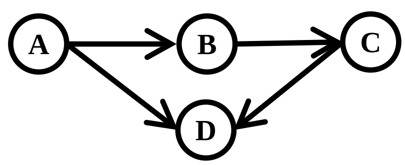


4e. P(B|A) P(A) P(C|D) P(D)

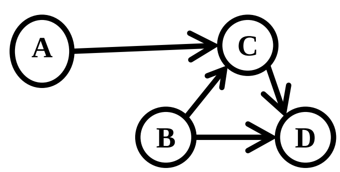
Write down the factored conditional probability equation that corresponds to the graphical Bayesian Network shown.



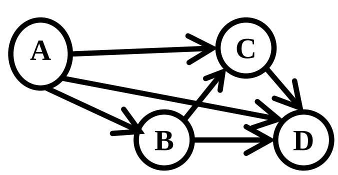
4f. P(D|A,B,C) P(A) P(B) P(C)



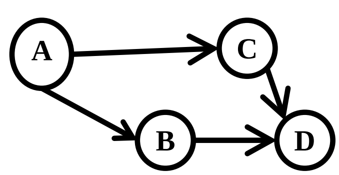
4g. P(D|A,C) P(C|B) P(B|A) P(A)



4h. P(D|B,C) P(C|A,B) P(B) P(A)



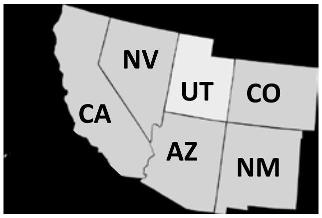
4i. P(D|A,B,C) P(C|A,B) P(B|A) P(A)

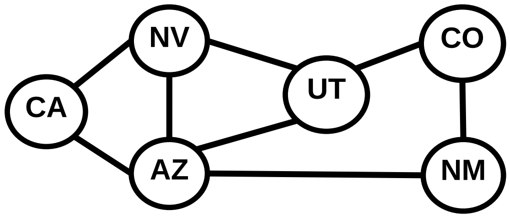


4j. P(D|B,C) P(C|A) P(B|A) P(A)

5. (10 points total, 2 pts each) Constraint Satisfaction Problems.

See Chapter 6.





You are a map-coloring robot assigned to color this Southwest USA map. Adjacent regions must be colored a different color (R=Red, B=Blue, G=Green). The constraint graph is shown.

5a. (2pts total, -1 each wrong answer, but not negative) FORWARD CHECKING. Cross out all values that would be eliminated by Forward Checking, after variable AZ has just been assigned value R as shown:

CA	NV	AZ	UT	CO	NM
~~R~~ G B	~~R~~ G B	R	~~R~~ G B	R G B	~~R~~ G B

5b. (2pts total, -1 each wrong answer, but not negative) ARC CONSISTENCY. CA and AZ have been assigned values, but no constraint propagation has been done. Cross out all values that would be eliminated by Arc Consistency (AC-3 in your book).

CA	NV	AZ	UT	CO	NM
B	~~R~~ G ~~B~~	R	~~R~~ ~~G~~ B	R G ~~B~~	~~R~~ G B

5c. (2pts total, -1 each wrong answer, but not negative) MINIMUM-REMAINING-VALUES HEURISTIC. Consider the assignment below. NV is assigned and constraint propagation has been done. List all unassigned variables that might be selected by the Minimum-Remaining-Values (MRV) Heuristic: **CA, AZ, UT** .

CA	NV	AZ	UT	CO	NM
R B	G	R B	R B	R G B	R G B

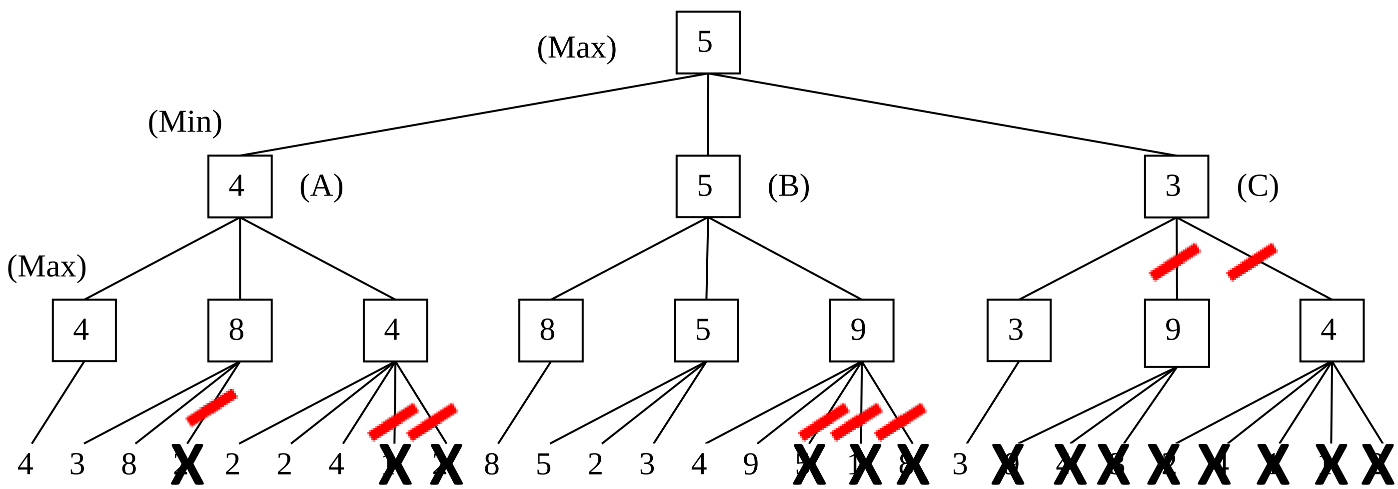
5d. (2pts total, -1 each wrong answer, but not negative) DEGREE HEURISTIC. Consider the assignment below. (It is the same assignment as in problem 5c above.) NV is assigned and constraint propagation has been done. List all unassigned variables that might be selected by the Degree Heuristic: **AZ** .

CA	NV	AZ	UT	CO	NM
R B	G	R B	R B	R G B	R G B

5e. (2pts total) MIN-CONFLICTS HEURISTIC. Consider the complete but inconsistent assignment below. AZ has just been selected to be assigned a new value during local search for a complete and consistent assignment. What new value would be chosen below for AZ by the Min-Conflicts Heuristic? **R** .

CA	NV	AZ	UT	CO	NM
B	G	**?**	G	G	B

6. (5 pts total, -1 for each error, but not negative) Alpha-Beta Pruning. In the game tree below it is **Max**'s turn to move. At each leaf node is the estimated score of that resulting position as returned by the heuristic static evaluator.
(1) Perform Mini-Max search and label each branch node with its value.
(2) Cross out each leaf node that would be pruned by alpha-beta pruning.
(3) What is Max's best move (A, B, or C)? B



7. (10 pts total, -2 for each error, but not negative) Conversion to CNF. Convert this Propositional Logic wff (well-formed formula) to Conjunctive Normal Form and simplify. **Show your work (correct result, 0 pts; correct work, 10 pts).**

$$P \Rightarrow [\neg (Q \Leftrightarrow P)]$$

See Section 7.5.2.

$P \Rightarrow [\neg \{(Q \Rightarrow P) \wedge (P \Rightarrow Q)\}]$ /* convert $(A \Leftrightarrow B)$ into $(A \Rightarrow B) \wedge (B \Rightarrow A)$ */

$\neg P \vee [\neg \{(\neg Q \vee P) \wedge (\neg P \vee Q)\}]$ /* convert $(A \Rightarrow B)$ into $(\neg A \vee B)$ */

$\neg P \vee [(Q \wedge \neg P) \vee (P \wedge \neg Q)]$ /* apply DeMorgan's Laws */

$(\neg P \vee Q \vee P) \wedge (\neg P \vee Q \vee \neg Q) \wedge (\neg P \vee \neg P \vee P) \wedge (\neg P \vee \neg P \vee \neg Q)$ /* distribute */

$True \wedge True \wedge True \wedge (\neg P \vee \neg Q)$ /* simplify */

$(\neg P \vee \neg Q)$ /* simplify */

It is OK to omit the disjunction symbol here.

8. (10 pts total, -2 for each error, but not negative) Resolution Theorem Proving. You are a robot in a logic-based question answering system, and must decide whether or not an input goal sentence is entailed by your Knowledge Base (KB). Your current KB in CNF is:

S1: (P Q)
S2: (¬P Q)
S3: (P ¬Q)

See Section 7.5.2.

Your input goal sentence is: (P ∧ Q).

8a. (2 pts) Write the negated goal sentence in CNF.

S4: ___(¬P ¬Q)___

It is OK to insert the disjunction symbol here.

8b. (8 pts total, -2 for each error, but not negative) Use resolution to prove that the goal sentence is entailed by KB, or else explain why no such proof is possible. For each step of the proof, fill in Si and Sj with the sentence numbers of previous CNF sentences that resolve to produce the CNF result that you write in the resolvent blank. The resolvent is the result of resolving the two sentences Si and Sj. Use as many steps as necessary, ending by producing the empty clause; or else explain why no such proof is possible.

The first one is done for you as an example.

Resolve Si ___S1___ with Sj ___S2___ to produce resolvent **S5:** ___(Q)___

Resolve Si ___S1___ with Sj ___S3___ to produce resolvent **S6:** ___(P)___

Resolve Si ___S4___ with Sj ___S5___ to produce resolvent **S7:** ___(¬P)___

Resolve Si ___S6___ with Sj ___S7___ to produce resolvent **S8:** ___( )___

Resolve Si ________ with Sj __________ to produce resolvent **S9:** ______________

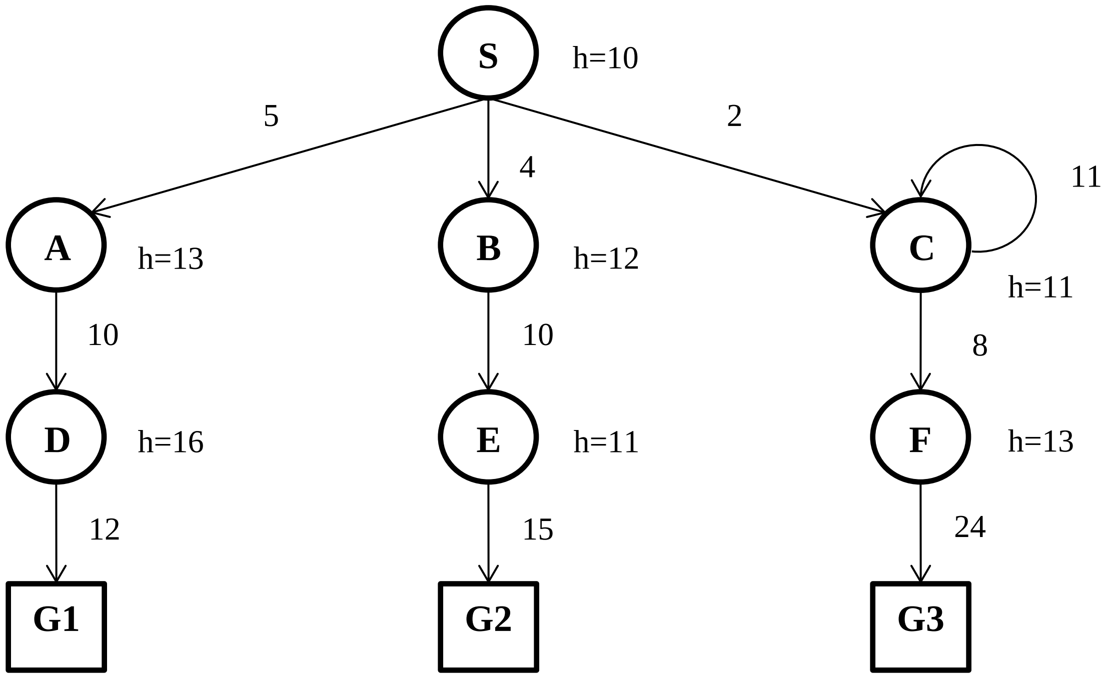
Resolve Si ________ with Sj __________ to produce resolvent **S10:** ______________

Add additional lines below if needed; or, if no such resolution proof is possible, use the space below to explain why not:

Other proofs are OK as long as they are correct. E.g., you might instead resolve S4 with S6 to produce resolvent S7 as (¬ Q), then resolve that with S5 to produce S8 ( ).

9. (10 pts total, 2 pts each) State-Space Search. Execute Tree Search through this graph (do not remember visited nodes, so repeated nodes are possible). It is not a tree, but pretend you don't know that. Step costs are given next to each arc, and heuristic values are given next to each node (as h=x). The successors of each node are indicated by the arrows out of that node. **(Note: C is a successor of itself).** As usual, successors are returned in left-to-right order.

For each search strategy below, indicate the order in which nodes are expanded.



9.a. (2 pts, -1 for each wrong answer, but not negative) UNIFORM COST SEARCH.

See Section 3.4.2

S C B A F C E D F C G1

9.b. (2 pts, -1 for each wrong answer, but not negative) GREEDY BEST-FIRST SEARCH.

See Section 3.5.1

S C C C C C C C C C C etc.

9.c (2 pts, -1 for each wrong answer, but not negative) ITERATIVE DEEPENING SEARCH.

See Section 3.4.5

S S A B C S A D B E C F C S A D G1

9.d. (2 pts, -1 for each wrong answer, but not negative) A* SEARCH.

See Section 3.5.2

S C B A F C E G2

9.e. (2 pts, -1 for each wrong answer, but not negative) OPTIMALITY.

Did Uniform Cost Search find the optimal goal? Yes

Why or why not? Step costs are $\geq \varepsilon > 0$

Did A* Search find the optimal goal? No

Why or why not? heuristic is not admissible (at D)

10. (8 pts total, 1 pt each) Puzzle-Solving. The sliding-tile puzzle has three black tiles (B), three white tiles (W), and an empty space (blank). The starting state is:

B	B	B		W	W	W

The goal is to have all the white tiles to the left of all the black tiles; the position of the blank is not important.

See Chapter 3.

The puzzle has two legal moves with associated costs:
(1) A tile may move into an adjacent empty location. This has a cost of 1.
(2) A tile may hop over one or two other tiles into the empty location. This has a cost equal to the number of tiles jumped over.

10a. What is the branching factor? ____6_____

10b. Does the search space have loops (cycles)? (Y=yes, N=no) __Y_____

10c. Is breadth-first search optimal? (“Y" = yes, “N" = no) ___N_____

10d. Is uniform-cost search optimal? (“Y" = yes, “N" = no) ___Y_____

10e. Consider a heuristic function h1(n) = the number of black tiles to the left of the left-most white tile. Is this heuristic admissible? (“Y" = yes, “N" = no) ___Y_____

10f. Consider a heuristic function h2(n) = the number of black tiles to the left of the right-most white tile. Is this heuristic admissible? (“Y" = yes, “N" = no) ___Y_____

10g. Consider a heuristic function h3(n) = the number of black tiles to the left of the right-most white tile plus the number of white tiles to the right of the left-most black tile. Is this heuristic admissible? (“Y" = yes, “N" = no) _____N_____

10h. Consider a heuristic function h4(n) = h3(n) / 2. Is this heuristic admissible? (“Y" = yes, “N" = no) _____Y_____

11. (2 pts total, 1 pt each) Heuristics. Suppose that there is no good step cost or path cost for a problem, i.e., no cost-so-far function g(n). However, there is a good comparison method: a binary test to tell whether one node is cheaper than another, but not to assign numerical values to either. Answer Y (= yes) or N (= no).

See Chapter 3.

~~**11a.** Is this enough to do a greedy best-first search? __Y_____~~

Question 11a was discarded as ambiguous. Everyone automatically gets it right.

11b. Suppose you also have a consistent heuristic, h(n). Is this enough to do A* search and guarantee an optimal solution? ____N_____

12. (10 pts total, 2 pts each) English to FOL Conversion. For each English sentence below, write the FOL sentence that best expresses its intended meaning. Use Person(x) for "x is a person," Food(x) for "x is food," and Likes(x, y) for "x likes y."

The first one is done for you as an example.

See Section 8.2.6

12a. (2 pts) "Every person likes every food."

$\forall x\ \forall y\ [\ Person(x) \land Food(y)\ ] \Rightarrow Likes(x, y)$

12b. (2 pts) "For every food, there is a person who likes that food."

$\forall y\ \exists x\ Food(y) \Rightarrow [\ Person(x) \land Likes(x, y)\ ]$

12c. (2 pts) "There is a person who likes every food."

$\exists x\ \forall y\ Person(x) \land [Food(y) \Rightarrow Likes(x, y)\ ]$

12d. (2 pts) "Some person likes some food."

$\exists x\ \exists y\ Person(x) \land Food(y) \land Likes(x, y)$

12e. (2 pts) "There is a food that every person likes."

$\exists y\ \forall x\ Food(y) \land [\ Person(x) \Rightarrow Likes(x, y)\ ]$

12f. (2 pts) "For every person, there is a food that the person likes."

$\forall x\ \exists y\ Person(x) \Rightarrow [\ Food(y) \land Likes(x, y)\ ]$