CS-171, Intro to A.I. — Final Exam — Fall Quarter, 2013

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 10 pages, as numbered 1-10 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. (15 pts total, 5 pts each, -1 each error, but not negative) Bayesian Networks.

2. (10 pts total, -1 for each error, but not negative) Mini-Max, Alpha-Beta Pruning.

3. (10 pts total, -2 for each error, but not negative) Conversion to CNF.

4. (10 points total, 2 pts each) Constraint Satisfaction Problems.

5. (5 pts total, -1 for each error, but not negative) Constraint Graph.

6. (5 pts total, -2 for each error, but not negative) k-Nearest-Neighbor Classifier.

7. (10 pts total, 2 pts each) Search.

8. (10 pts total) Decision Tree Classifier Learning.

9. (5 pts total, -1 pt each wrong answer, but not negative) Search Properties.

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

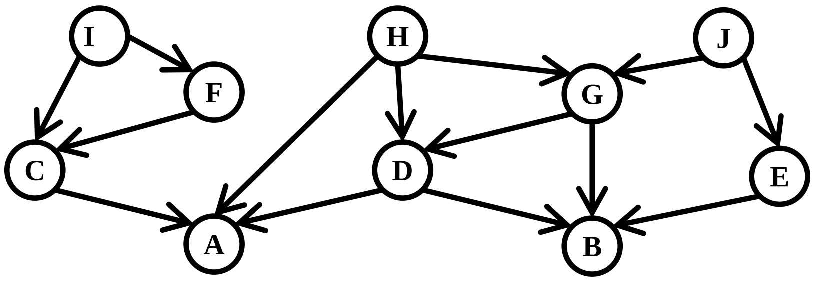
11. (10 pts total, 2 pts each) FOPC Knowledge Engineering In The Blocks World.

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

See Section 14.2.

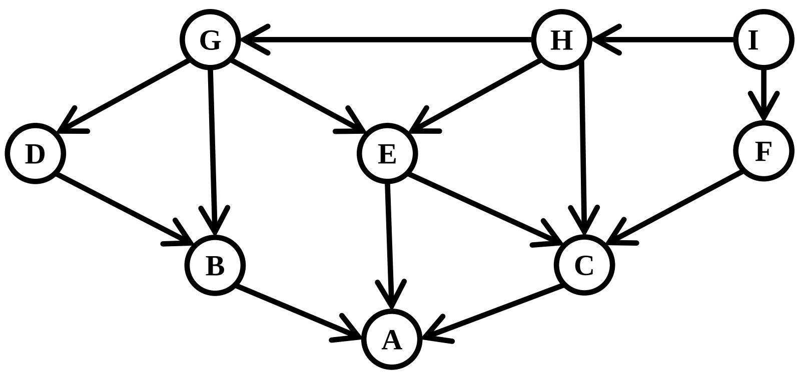
1. (15 pts total, 5 pts each, -1 each error, but not negative) Bayesian Networks.
1a. (5 pts) Write down the factored conditional probability expression that corresponds to the graphical Bayesian Network shown.

P(A | C,D,H) .P(B | D,E,G) P(C | F,I) P(D | G,H) P(E | J) P(F | I) P(G | H,J) P(H) P(I) P(J)

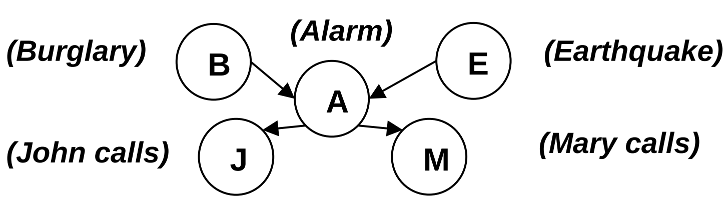


1b. (5 pts) Draw the Bayesian Network that corresponds to this conditional probability:

P(A | B,C,E) P(B | D,G) P(C | E,F,H) P(D | G) P(E| G,H) P(F | I) P(G | H) P(H | I) P(I)



1.c. (5 pts) Shown below is the Bayesian network corresponding to the Burglar Alarm problem, P(J | A) P(M | A) P(A | B, E) P(B) P(E).



P(E)
.002

P(B)
.001

A	P(M)
t	.70
f	.01

A	P(J)
t	.90
f	.05

B	E	P(A)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

The probability tables show the probability that variable is True, e.g., P(M) means P(M=t). Write down an expression that will evaluate to P(j=f ∧ m=t ∧ a=t ∧ b = t ∧ e = f). Express your answer as a series of numbers (numerical probabilities) separated by multiplication symbols. You do not need to carry out the multiplication to produce a single number (probability). **SHOW YOUR WORK.**

P(j=f ∧ m=t ∧ a=t ∧ b = t ∧ e = f)
= P(j=f | a=t) * P(m=t | a = t) * P(a=t | b=t ∧ e=f) * P(b=t) * P(e=f)
= .10 * .70 * .94 * .001 * .998

2. (10 pts total, -1 for each error, but not negative) Mini-Max, 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.

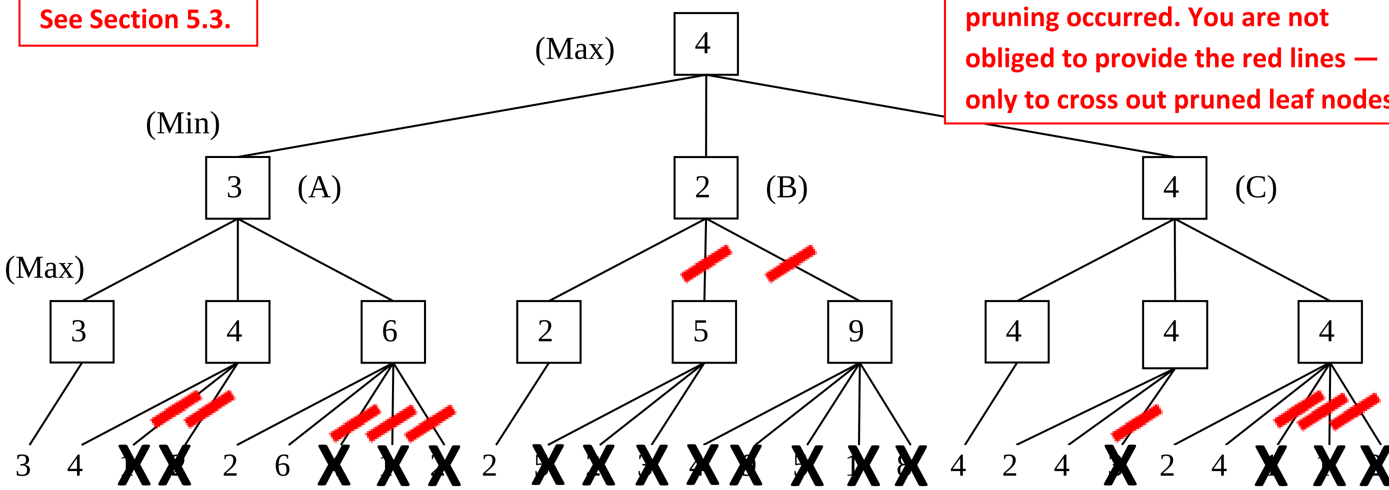
(1) Perform Mini-Max search and label each branch node with its value.

(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)? C

See Section 5.3.

Red lines indicate where in the tree pruning occurred. You are not obliged to provide the red lines — only to cross out pruned leaf nodes.



3. (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, 5 pts).**

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

See Section 7.5.2.

/* convert (A ⇒ B) into (¬ A ∨ B), cancel double negation */

(Q ⇔ P) ∨ Q

/* convert (A ⇔ B) into [(A ⇒ B) ∧ (B ⇒ A)] */

[(Q ⇒ P) ∧ (P ⇒ Q)] ∨ Q

/* convert (A ⇒ B) into (¬ A ∨ B) */

[(¬ Q ∨ P) ∧ (¬ P ∨ Q)] ∨ Q

/* distribute "∨ Q" over ∧ */

[(¬ Q ∨ P ∨ Q) ∧ (¬ P ∨ Q ∨ Q)]

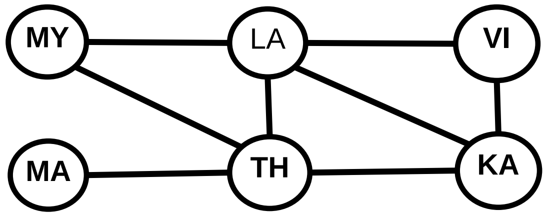
/* simplify */

[(True) ∧ (¬ P ∨ Q)]

/* simplify */

(¬ P ∨ Q)

4. (10 points total, 2 pts each) Constraint Satisfaction Problems. See Chapter 6.



KA=Kampuchia
LA=Laos
MA=Malaysia
MY=Myanmar
TH=Thailand
VI=Vietnam

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

4a. (2pts total, -1 each wrong answer, but not negative) FORWARD CHECKING. Cross out all values that would be eliminated by Forward Checking, after variable KA has just been assigned value R as shown:

KA	LA	MA	MY	TH	VI
R	~~R~~ G B	R G B	R G B	~~R~~ G B	~~R~~ G B

4b. (2pts total, -1 each wrong answer, but not negative) ARC CONSISTENCY. KA and TH 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).

KA	LA	MA	MY	TH	VI
R	~~R~~ ~~G~~ B	R ~~G~~ B	R ~~G~~ ~~B~~	G	~~R~~ G ~~B~~

4c. (2pts total, -1 each wrong answer, but not negative) MINIMUM-REMAINING-VALUES HEURISTIC. Consider the assignment below. TH is assigned and constraint propagation has been done. List all unassigned variables that might be selected by the Minimum-Remaining-Values (MRV) Heuristic: **KA, LA, MA, MY** .

KA	LA	MA	MY	TH	VI
G B	G B	G B	G B	R	R G B

4d. (2pts total, -1 each wrong answer, but not negative) DEGREE HEURISTIC. Consider the assignment below. (It is the same assignment as in problem 4c above.) TH is assigned and constraint propagation has been done. List all unassigned variables that might be selected by the Degree Heuristic: **LA** .

KA	LA	MA	MY	TH	VI
G B	G B	G B	G B	R	R G B

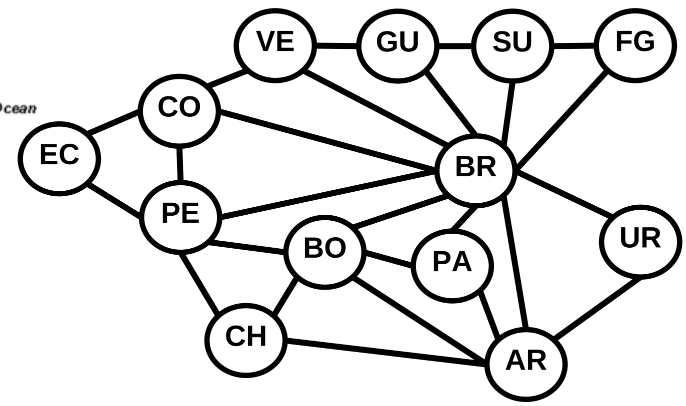
4e. (2pts total) MIN-CONFLICTS HEURISTIC. Consider the complete but inconsistent assignment below. LA has just been selected to be assigned a new value during local search for a complete and consistent assignment. What new value would be chosen below for LA by the Min-Conflicts Heuristic? **R** .

KA	LA	MA	MY	TH	VI
B	**?**	R	G	G	B

5. (5 pts total, -1 for each error, but not negative) Constraint Graph.
You are a map-coloring robot assigned to color this South America map. Adjacent regions must be colored a different color. Draw the constraint graph corresponding to the map.

AR=Argentina
BO=Bolivia
BR=Brazil
CH=Chile
CO=Colombia
EC=Ecuador
FG=French Guiana
GU=Guyana
PA=Paraguay
PE=Peru
SU=Suriname
UR=Uruguay
VE=Venezuela

Ignore territories not listed above

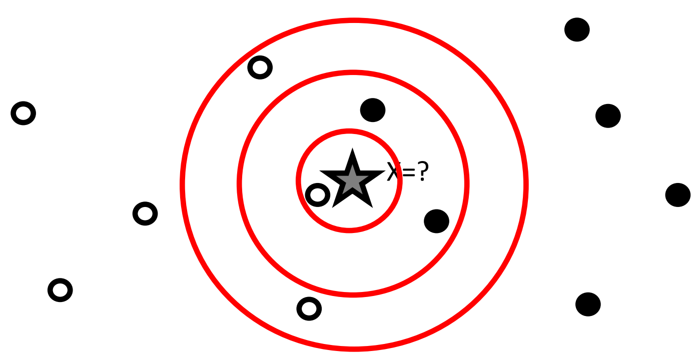


6. (5 pts total, -2 for each error, but not negative) k-Nearest-Neighbor Classifier.
You have trained a k-Nearest Neighbor classifier on the points shown below, which are labeled as either Black (B) or White (W). What will the classifier return as the class for the point X, shown as a star, in the following scenarios? Answers should be Black (B), White (W), or Tie (T).

When k = 1? ____W_____

When k = 3? ____B_____

When k = 5? ____W_____



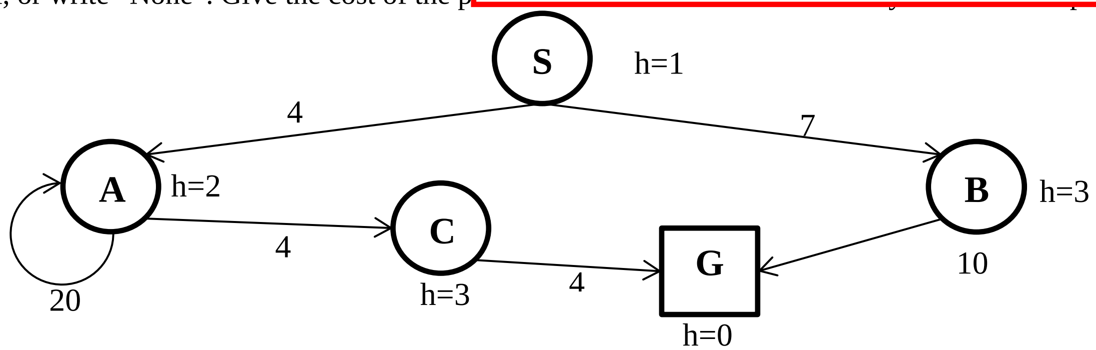
7. (10 pts total, 2 pts each) Execute Tree Search through this graph (i.e., do not remember visited nodes). Ste[...]o each arc. Heuristic values are next to each node (as h=x). The successors of eac

See Chapter 3.

the arrows out of

Please see the lecture slides for Uninformed Search, topic "When to do Goal-Test? When generated? When popped?" for clarification about exactly what to do in practical cases.

For each search strategy, show the o
means that its children are generated), endin
goal, or write "None". Give the cost of the p



7.a. BREADTH FIRST SEARCH:

See Section 3.4.1 and Fig. 3.11.

sion: S A B G

BFS does the Goal-test before the child is pushed onto the queue. The goal is found when C is expanded.

Path found: S B G Cost of path found: 17

7.b. (2 pts) DEPTH FIRST SEARCH:

See Section 3.4.3 and Fig. 3.17.

sion: S A A A A A A etc.

DFS can get caught in loops during Tree Search (= do not remember visited nodes).

Path found: None Cost of path found: None

7.c. (2 pts) UNIFORM COST SEARCH:

See Section 3.4.2 and Fig. 3.14.

sion: S A B C G

UCS does goaltest when node is popped off queue.

Path found: S A C G Cost of path found: 12

7.d. (2 pts) GREEDY (BEST-FIRST) SEARCH:

See Section 3.5.1 and Fig. 3.23.

sion: S A A A A A A etc.

A always has lower h(=2) than any other node on queue.

Path found: None Cost of path found: None

7.e. (2 pts) ITERATED DEEPENING SEARCH:

See Sections 3.4.4-5 and Figs. 3.18-19.

n: S S A B G

IDS does goaltest when node is generated. Goal G1 is found when B is expanded.

Path found: S B G Cost of path found: 17

7.f. (2 pts) A* SEARCH:

A* does goaltest when node is popped off queue.

See Section 3.5.2 and Figs. 3.24-25.

de expansion: S A B C G

: S A C G Cost of path found: 12

Is the heuristic admissible (Yes or No)? Yes

Is the heuristic consistent (Yes or No)? Yes

8. (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:

See Section 18.3.

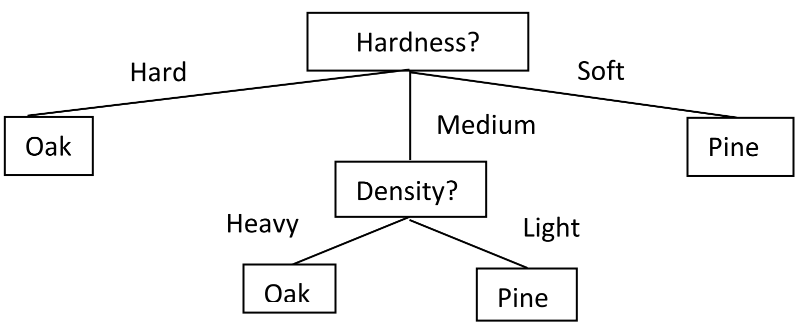
Example	Density	Grain	Hardness	Class
Example #1	Light	Small	Hard	Oak
Example #2	Heavy	Large	Hard	Oak
Example #3	Light	Large	Hard	Oak
Example #4	Heavy	Small	Medium	Oak
Example #5	Light	Small	Medium	Pine
Example #6	Heavy	Large	Soft	Pine
Example #7	Light	Large	Soft	Pine
Example #8	Heavy	Small	Soft	Pine

If root is Density:
Heavy = OOPP, Light = OOPP
If root is Grain:
Small = OOPP, Large = OOPP
If root is Hardness:
Hard=OOO, Medium=OP, Soft=PPP
(O = Oak, P = Pine)

8a. (2 pts) Which attribute would information gain choose as the root of the tree?

______Hardness_______________.

8b. (4 pts) Draw the decision tree that would be constructed by recursively applying information gain to select roots of sub-trees, as in the Decision-Tree-Learning algorithm.



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

Classify these new examples as Oak or Pine using your decision tree above.
8c. (2 pts) What class is [Density=Light, Grain=Small, Hardness=Soft]? __Pine__
8d. (2 pts) What class is [Density=Heavy, Grain=Large, Hardness=Medium]? __Oak__

9. (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

10. (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)
S4: (¬P ¬Q ¬S)
S5: (¬P R S)

See Section 7.5.2.

Other proofs are OK if they are correct.

It is OK if you wrote the Boolean expression instead of S1, S2, etc. It is also OK if you put disjunction symbols inside the CNF clauses.

Your input goal sentence is: (P ∧ Q ∧ R).

10a. (2 pts) Write the negated goal sentence in CNF.

S6: (¬P ¬Q ¬R)

It is OK to insert the disjunction symbol here.

10b. (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. The shortest proof I know of is only six lines long.

Resolve Si S5 with Sj S6 to produce resolvent **S7:** (¬P ¬Q S)

Resolve Si S4 with Sj S7 to produce resolvent **S8:** (¬P ¬Q)

Resolve Si S3 with Sj S8 to produce resolvent **S9:** (¬Q)

Resolve Si S2 with Sj S9 to produce resolvent **S10:** (¬P)

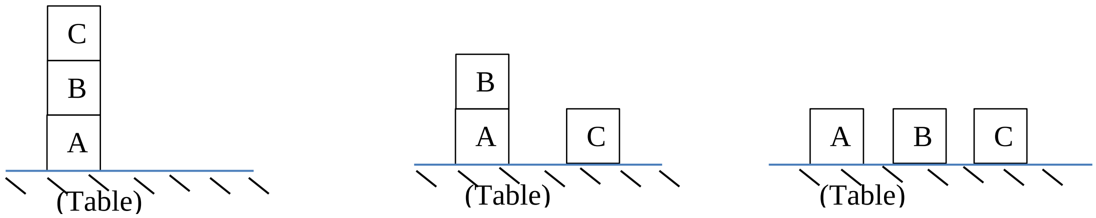
Resolve Si S1 with Sj S10 to produce resolvent **S11:** (Q)

Resolve Si S11 with Sj S9 to produce resolvent **S12:** ( )

A bright and attentive student found a shorter proof (and was awarded TWO (2) BONUS POINTS).

Resolve (¬P R S) with (¬P ¬Q ¬R) to yield (¬P ¬Q S)
Resolve (¬P ¬Q ¬S) with (¬P ¬Q S) to yield (¬P ¬Q)
Resolve (P ¬Q) with (¬P ¬Q) to yield (¬Q)
Resolve (P Q) with (¬P Q) to yield (Q)
Resolve (¬Q) with (Q) to yield ( )

11. (10 pts total, 2 pts each) FOPC KNOWLEDGE ENGINEERING IN THE BLOCKS WORLD. You are a Knowledge Engineer assigned to the Blocks World, which involves controlling a robot arm that stacks children's blocks one atop another, or on a table, into a desired configuration. For example, you are concerned with configurations such as:



Here we wish only to describe static (unmoving) configurations that will become goals (targets) in the Blocks World. A separate module, not your concern, will move the robot arm to achieve these goals. You need only to describe correctly in FOPC the static goal (target) configurations.

Write the FOPC sentence that best represents the corresponding English sentence. Assume that all objects in the world are blocks, i.e., there is no need for Block(x) guard predicates. Use "Stacked(x, y)" to mean that "Block x is stacked directly on top of block y."

The first one is done for you as an example.

11.a. Assert that "Block x is stacked on block y" implies "Block y is not stacked on block x."

$\forall x, y$ Stacked(x, y) $\Rightarrow \neg$Stacked(y, x)

11.b. (2pts) Define a predicate "Clear(x)" to mean that no block y is stacked on block x.

$\forall x, y$ Clear(x) $\Leftrightarrow \neg$Stacked(y, x)

11.c. (2pts) Define a predicate "OnTable(x)" to mean that block x is on the table, i.e., not on any block.

$\forall x, y$ OnTable(x) $\Leftrightarrow \neg$Stacked(x, y)

11.d. (2pts) Define a predicate "Above(x, y)" to mean that x is above y in a stack including both x and y.

$\forall x, y$ Above(x, y) $\Leftrightarrow$ [Stacked(x, y) $\vee$ ($\exists z$ Stacked(z, y) $\wedge$ Above(x, z))]

11.e. (2pts) State that at least one block must have no other block stacked upon it. You may use the Clear(x) predicate that you defined in (10.b) above

$\exists x$ Clear(x)
$\exists x \forall y \neg$Stacked(y, x)

11.f. State that at least one block must be on the table. You may use the OnTable(x) predicate that you defined in (10.c) above

$\exists x$ OnTable(x)
$\exists x \forall y \neg$Stacked(x, y)

****** THIS IS THE END OF THE FINAL EXAM ******