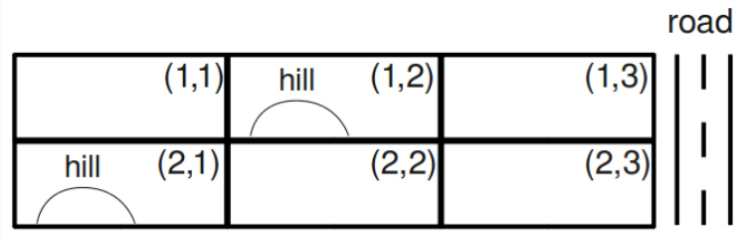
Q1 CSP- Campus Layout
44 Points

You are asked to determine the layout of a new, small college. The campus will have four structures: an administration structure (A), a bus stop (B), a classroom (C), and a dormitory (D). Each structure (including the bus stop) must be placed somewhere on the grid shown below.



The layout must satisfy the following constraints:

i.The bus stop (B) must be adjacent to the road.

ii. The administration structure (A) and the classroom (C) must both be adjacent to the bus stop (B).

iii. The classroom (C) must be adjacent to the dormitory (D).

iv. The administration structure (A) must not be adjacent to the dormitory (D).

v. The administration structure (A) must not be on a hill.

vi. The dormitory (D) must be on a hill or adjacent to the road.

vii. All structures must be in different grid squares.

Here, adjacent means that the structures must share a grid edge, not just a corner.

We recommend you work out the solutions to the following questions on a sheet of scratch paper, and then enter your results below.

Q1.1
4 Points

Which of the constraints above are unary constraints?

- [x] i
- [ ] ii
- [ ] iii
- [ ] iv
- [x] v
- [x] vi
- [ ] vii
- [ ] None

Q1.2
4 Points

Select the domains of all variables after unary constraints have been applied.

A

- [x] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [x] (2,2)

Homework 2 — GRADED

STUDENT
Rushikesh Machhindra Khamkar

TOTAL POINTS
100 / 100 pts

QUESTION 1
CSP- Campus Layout — **44** / 44 pts

1.1	(no title)	**4** / 4 pts
1.2	(no title)	**4** / 4 pts
1.3	(no title)	**4** / 4 pts
1.4	(no title)	**4** / 4 pts
1.5	(no title)	**4** / 4 pts
1.6	(no title)	**4** / 4 pts
1.7	(no title)	**4** / 4 pts
1.8	(no title)	**4** / 4 pts
1.9	(no title)	**4** / 4 pts
1.10	(no title)	**4** / 4 pts

- [x] (2,3)

B

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [ ] (2,2)
- [x] (2,3)

C

- [x] (1,1)
- [x] (1,2)
- [x] (1,3)
- [x] (2,1)
- [x] (2,2)
- [x] (2,3)

D

- [ ] (1,1)
- [x] (1,2)
- [x] (1,3)
- [x] (2,1)
- [ ] (2,2)
- [x] (2,3)

Q1.3
4 Points

Let's start from the table above (the answer to Part 2) and enforce arc consistency. Initially, the queue contains all arcs (in alphabetical order).

Let's examine what happens when enforcing A → B. After enforcing unary constraints, the domains of A and B are:

A	B
(1,1) (1,3) (2,2) (2,3)	(1,3) (2,3)

Which of the following contains the correct domains after enforcing A → B?
Pay attention to which variable's domain changes and which side of the arc it's on.

A	B	A	B	A	B	A	B

A	B
(1,1)	
	(1,2)
(1,3)	(1,3)
(2,2)	
(2,3)	(2,3)

i

A	B
(1,3)	(1,3)
(2,2)	
(2,3)	(2,3)

ii

A	B
(1,1)	
(1,3)	
(2,2)	
(2,3)	(2,3)

iii

A	B
(1,1)	
(1,3)	(1,3)
(2,2)	
(2,3)	

iv

- i
- ii
- iii
- iv

Q1.4
4 Points

Starting from the answer to Part 2 (in which unary constraints are enforced), select the domains of all variables after A → B is enforced.

A

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [x] (2,2)
- [x] (2,3)

B

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [ ] (2,2)
- [x] (2,3)

C

- [x] (1,1)
- [x] (1,2)
- [x] (1,3)
- [x] (2,1)
- [x] (2,2)
- [x] (2,3)

D

- [ ] (1,1)
- [x] (1,2)

- [x] (1,3)
- [x] (2,1)
- [ ] (2,2)
- [x] (2,3)

Q1.5
4 Points

You should verify that enforcing consistency for A → C, A → D, B → A, B → C, B → D, and C → A do not change the domains of any variables. After enforcing these arcs, the next is C → B.

Continuing from the previous parts, select the domains of all variables after C → B is enforced.

A

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [x] (2,2)
- [x] (2,3)

B

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [ ] (2,2)
- [x] (2,3)

C

- [ ] (1,1)
- [x] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [x] (2,2)
- [x] (2,3)

D

- [ ] (1,1)
- [x] (1,2)
- [x] (1,3)
- [x] (2,1)
- [ ] (2,2)
- [x] (2,3)

Q1.6
4 Points

What arcs got added to the queue while enforcing C → B? Remember that the queue contained C → D, D → A, D → B, and D →C prior to enforcing C → B. Remember, when enforcing an arc (i.e., remove values from a variable), all arcs pointing to that variable will be re-added to the queue.

- [ ] A → B
- [x] A → C
- [ ] A → D
- [ ] B → A
- [x] B → C
- [ ] B → D
- [ ] C → A
- [ ] C → B
- [ ] C → D
- [ ] D → A
- [ ] D → B
- [ ] D → C
- [ ] None

Q1.7
4 Points

Continuing from the previous parts, select the domains of all variables after enforcing arc consistency until the queue is empty. Remember that the queue currently contains C → D, D → A, D → B, D → C, and any arcs that were added while enforcing C → B.

A

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [x] (2,2)
- [x] (2,3)

B

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [ ] (2,2)
- [x] (2,3)

C

- [ ] (1,1)
- [x] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [x] (2,2)
- [x] (2,3)

D

- [ ] (1,1)
- [x] (1,2)
- [x] (1,3)
- [x] (2,1)
- [ ] (2,2)
- [ ] (2,3)

Q1.8
4 Points

If arc consistency had resulted in all domains having a single value left, we would have already found a solution. Similarly, if it had been found that any domain had no values left, we would have already found that no solution exists. Unfortunately, this is not the case in our example (as you should have found in the previous part). To solve the problem, we need to start searching. Use the MRV (minimum remaining values) heuristic to choose which variable gets assigned next (breaking any ties alphabetically).

Which variable gets assigned next?

- ○ A
- ◉ B
- ○ C
- ○ D

Q1.9
4 Points

The variable you selected should have two values left in its domain. We will use the least-constraining value (LCV) heuristic to decide which value to assign before continuing with the search. To choose which value is the least-constraining value, enforce arc consistency for each value (on a scratch piece of paper). For each value, count the total number of values remaining over all variables.

Which value has the largest number of values remaining (and therefore is the least constraining value)?

- ○ (1,1)
- ○ (1,2)
- ○ (1,3)
- ○ (2,1)
- ○ (2,2)
- ◉ (2,3)

Q1.10
4 Points

After assigning a variable, backtracking search with arc consistency enforces arc consistency before proceeding to the next variable.

Select the domains of all variables after assignment of the least-constraining value to the variable you selected and enforcing arc consistency. Note that you already did this computation to determine which value was the LCV.

A

- [ ] (1,1)
- [ ] (1,2)
- [x] (1,3)
- [ ] (2,1)
- [ ] (2,2)
- [ ] (2,3)

B

- [ ] (1,1)
- [ ] (1,2)
- [ ] (1,3)
- [ ] (2,1)
- [ ] (2,2)
- [x] (2,3)

C

- [ ] (1,1)
- [ ] (1,2)
- [ ] (1,3)
- [ ] (2,1)
- [x] (2,2)
- [ ] (2,3)

D

- [ ] (1,1)
- [ ] (1,2)
- [ ] (1,3)
- [x] (2,1)
- [ ] (2,2)
- [ ] (2,3)

Q1.11
4 Points

Is the answer to the previous part a solution to the CSP?

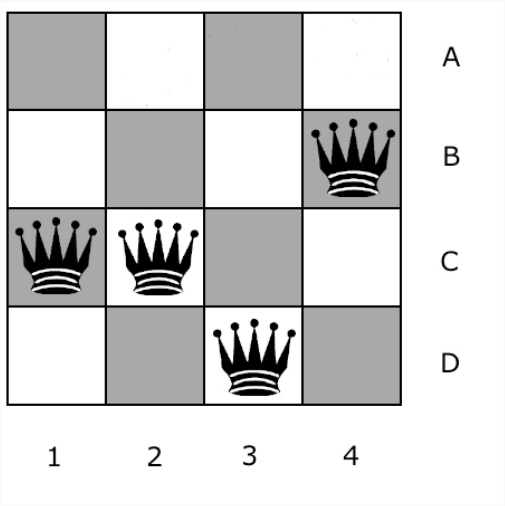
- (•) yes
- ( ) no

Q2 CSP- N-Queen

14 Points

The min-conflicts algorithm attempts to solve CSPs iteratively. It starts by assigning some value to each of the variables, ignoring the constraints when doing so. Then, while at least one constraint is violated, it repeats the following: (1) randomly choose a variable that is currently violating a constraint, (2) assign to it the value in its domain such that after the assignment the total number of constraints violated is minimized (among all possible selections of values in its domain).

In this question, you are asked to execute the min-conflicts algorithm on a simple problem: the 4-queens problem in the figure shown below. Each queen is dedicated to its own column (i.e. we have variables Q_1, Q_2, Q_3, and Q_4 and the domain for each one of them is $\{A, B, C, D\}$. In the configuration shown below, we have $Q_1 = C$,Q_2 = C, $Q_3 = D$, $Q_4 = B$. Two queens are in conflict if they share the same row, diagonal, or column (though in this setting, they can never share the same column).



You will execute min-conflicts for this problem three times, starting with the state shown in the figure above. When selecting a variable to reassign, min-conflicts chooses a conflicted variable at random. For this problem, assume that your random number generator always chooses the leftmost conflicted queen. When moving a queen, move it to the square in its column that leads to the fewest conflicts with other queens. If there are ties, choose the topmost square among them.

We recommend you work out the solutions to the following questions on a sheet of scratch paper, and then enter your results below.

Q2.1

4 Points

Starting with the queens in the configuration shown in the above figure, which queen will be moved, and where will it be moved to?

Queen

- (•) 1
- ( ) 2
- ( ) 3
- ( ) 4

Position

- (•) A
- ( ) B
- ( ) C
- ( ) D

Q2.2

5 Points

Continuing off of Part 1, which queen will be moved, and where will it be moved to?
Queen

- ○ 1
- ◉ 2
- ○ 3
- ○ 4

Position

- ◉ A
- ○ B
- ○ C
- ○ D

Q2.3
5 Points

Continuing off of Part 2, which queen will be moved, and where will it be moved to?
Queen

- ◉ 1
- ○ 2
- ○ 3
- ○ 4

Position

- ○ A
- ○ B
- ◉ C
- ○ D

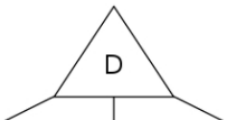
Q3 CSP- Arc Consistency Properties
1 Point

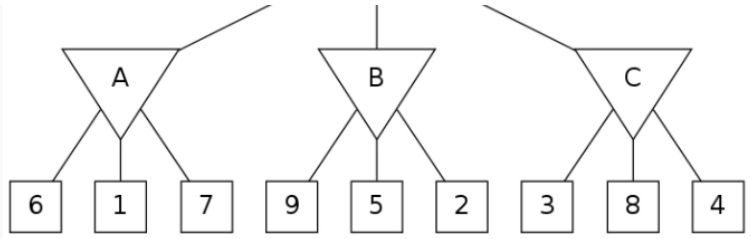
Assume you are given a CSP and you enforce arc consistency. Which of the following are true?

- ☐ If the CSP has no solution, it is guaranteed that enforcement of arc consistency resulted in at least one domain being empty.
- ☐ If the CSP has a solution, then after enforcing arc consistency, you can directly read off the solution from resulting domains.
- ☑ In general, to determine whether the CSP has a solution, enforcing arc consistency alone is not sufficient; backtracking may be required.
- ☐ None of the above.

Q4 GameTree-MiniMax
7 Points

Consider the zero-sum game tree shown below. Triangles that point up, such as at the top node (root), represent choices for the maximizing player; triangles that point down represent choices for the minimizing player. Outcome values for the maximizing player are listed for each leaf node, represented by the values in squares at the bottom of the tree. Assuming both players act optimally, carry out the minimax search algorithm. Enter the values for the letter nodes in the boxes below the tree.





Input Answers Here

A:

1

B:

2

C:

3

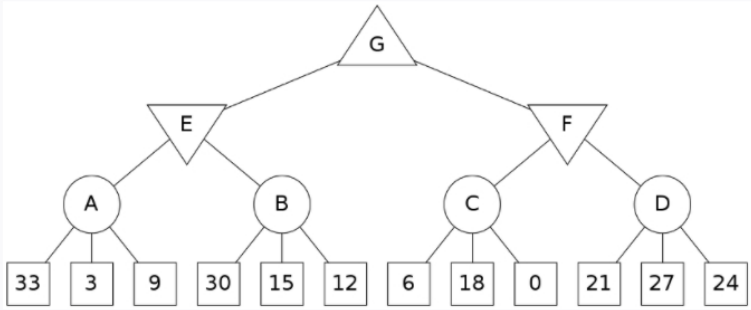
D:

3

Q5 GameTree-Expectiminimax

8 Points

Consider the game tree shown below. As in the previous problem, triangles that point up, such as the top node (root), represent choices for the maximizing player; triangles that point down represent choices for the minimizing player. The circular nodes represent chance nodes in which each of the possible actions may be taken with equal probability. The square nodes at the bottom represent leaf nodes. Assuming both players act optimally, carry out the expectiminimax search algorithm. Enter the values for the letter nodes in the boxes below the tree.



Input Answers Here

A:

15

B:

19

C:

8

D:

24

E:

15

F:

8

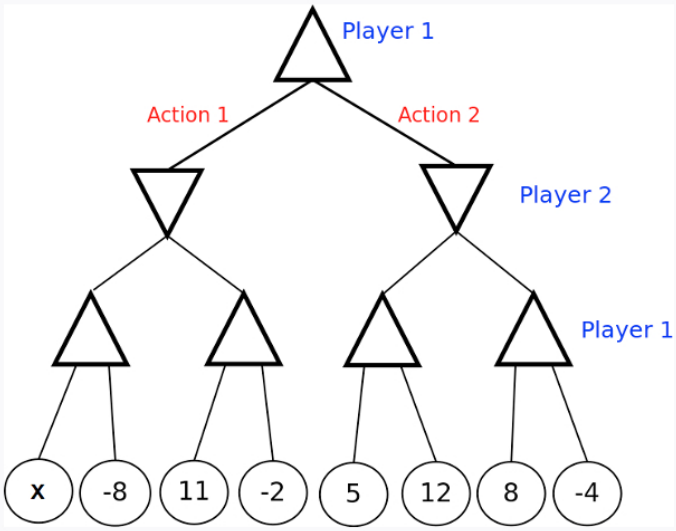
G:

15

Q6 GameTree-Unknown Leaf Value

12 Points

Consider the following game tree, where one of the leaves has an unknown payoff, x. Player 1 moves first, and attempts to maximize the value of the game.



Each of the next 3 questions asks you to write a constraint on x specifying the set of values it can take. Please specify your answer in one of the following forms:

```
Write All if x can take on all values
Write None if x has no possible values
Use an inequality in the form x<value, x>value, or value1<x<value2 to specify
an interval of values.  As an example, if you think x an take on all values
larger than 16, you should enter x>16.
```

Note that the answer check is sensitive to ordering, spacing, and capitalization, so do not use spaces in your answers and make sure the form matches the examples above.

Q6.1

3 Points

Assume Player 2 is a minimizing agent (and Player 1 knows this). For what values of x is Player 1 guaranteed to choose Action 1 for their first move?

x>8

Q6.2

3 Points

Assume Player 2 chooses actions at random with each action having equal probability (and Player 1 knows this). For what values of x is Player 1 guaranteed to choose Action 1?

x>9

Q6.3

3 Points

Denote the minimax value of the tree as the value of the root when Player 1 is the maximizer and Player 2 is the minimizer. Denote the expectimax value of the tree as the value of the root when Player 1 is the maximizer and Player 2 chooses actions at random (with equal probability).

For what values of x is the minimax value of the tree worth more than the expectimax value of the tree?

none

Q6.4
3 Points

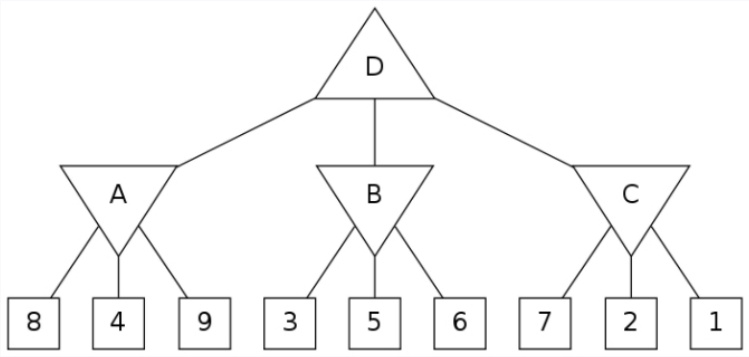
Is it possible to have a game, where the minimax value is strictly larger than the expectimax value?

- ○ yes
- ◉ no

Q7 GameTree- Alpha-Beta Pruning
8 Points

Consider the game tree shown below. Triangles that point up, such as at the top node (root), represent choices for the maximizing player; triangles that point down represent choices for the minimizing player. Assuming both players act optimally, use alpha-beta pruning to find the value of the root node. The search goes from left to right; when choosing which child to visit first, choose the left-most unvisited child. In the first set of boxes below, enter the values of the labeled nodes. Then, select the leaf nodes that don't get visited due to pruning.

Hint: Note that the value of a node where pruning occurs is not necessarily the maximum or minimum (depending on which node) of its children. When you prune on conditions $V > \beta$ or $V < \alpha$, assume that the value of the node is V.



Enter the values of the labeled nodes

A:

4

B:

3

C:

2

D:

4

Select the boxes for leaf values that don't get visited due to pruning

- ☐ 8
- ☐ 4
- ☐ 9
- ☐ 3
- ☑ 5

- [x] 6
- [ ] 7
- [ ] 2
- [x] 1

Q8 Non-Zero-Sum Games
6 Points

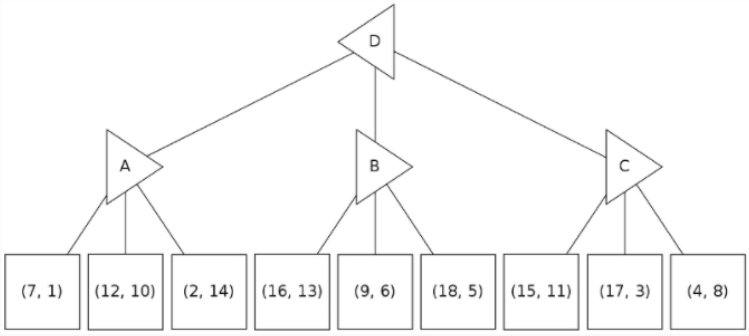
Q8.1
3 Points

The standard minimax algorithm calculates worst-case values in a zero-sum two player game, i.e. a game for which in all terminal states s, the utilities for players A (MAX) and B (MIN) obey $U_A(s) + U_B(s) = 0$. Thus, in this zero-sum setting, we know that $U_A(s) = -U_B(s)$, so we can think of player B as simply minimizing U_A.

In this problem, you will consider the non-zero-sum generalization, in which the sum of the two players' utilities are not necessarily zero. The leaf utilities are now written as pairs (U_A, U_B). In this generalized setting, A seeks to maximize U_A, the first component, while B seeks to maximize U_B, the second component.

Consider the non-zero-sum game tree below. Note that left-pointing triangles (such as the root of the tree) correspond to player A, who maximizes the first component of the utility pair, whereas right-pointing triangles (nodes on the second layer) correspond to player B, who maximizes the second component of the utility pair.

Propagate the terminal utility pairs up the tree using the appropriate generalization of the minimax algorithm on this game tree. In case of ties, choose the leftmost child. Select the correct values for the letter nodes below the tree.



Your answer should be in the format (X,Y), where X is the value of Player A and Y is the value of Player B at a node (for example "(7,1)").
Note that the answer check is sensitive to formatting, so be sure to include the parenthesis and do not use any spaces in your answers.

A:

(2,14)

B:

(16,13)

C:

(15,11)

D:

(16,13)

Q8.2

3 Points

In this problem, you will again consider the non-zero-sum generalization, in which the sum of the two players' utilities are not necessarily zero. The leaf utilities are now written as pairs (U_A, U_B). In this generalized setting, A seeks to maximize U_A, the first component, while B seeks to maximize U_B, the second component.

Assume that your generalization of the minimax algorithm calculates a value (U_A^*, U_B^*) for the root of the tree. Assume no utility value for A or for B appears more than once in the terminal nodes (this means there will be no need for tie-breaking).
Which of the following statements are true?

- [x] Assuming A and B both play optimally, player A's outcome is guaranteed to be exactly U_A^*.
- [x] Assuming A and B both play optimally, player B's outcome is guaranteed to be exactly U_B^*.
- [ ] Assuming B plays sub-optimally (but A plays optimally), A's outcome is guaranteed to be at least U_A^*.