Green flag Candidate number: **GF153**

<u>Differentiate between Mundane Tasks, Formal Tasks and Expert Tasks with 2 examples of each. In which category of tasks are humans currently outperforming computer systems? Justify your answer</u>

1. A mundane task is that is based on perception, such as speech and vision. One key example of this is natural language and its understanding, generation and translation. Mundane tasks are usually difficult to process because they can involve analog signals which are usually very noisy. Examples of this are robot control (basic assembly line work) or data entry (common sense reasoning).
   A formal task usually follow well-defined procedures and rules that can be defined by a computer, such as game playing and theorem proving. Computers are usually good at these tasks because these tasks can be precisely defined and programmed, and they are also good at exploring a large number of paths and solutions. This becomes an issue with combinatorially explosive problems. Examples of this are, applications in chess and other games, and mathematical calculations such as integral calculations.
   Expert tasks are ones which require a specialised knowledge and complex decision-making to process. Expert tasks are usually seen in engineering design or new scientific discovery. There are some early applications of AI which are designed to deal with these types of tasks.

   Currently, humans are outperforming computers doing mundane tasks such as translation work as natural language processing tasks require a deeper understanding of idioms and sarcasm than computers are currently able to process as accurately. For formal tasks, humans are outperforming computers to a lesser extent. Humans still currently outperform computers where contextual understanding is required and the information is incomplete. One example of this could be error correction, such as proofreading a document. This would be due to nuances and grammar. Finally, when it comes to expert tasks, where humans are currently predominantly outperforming computers. An example of this is a specialised doctor, where a rare disease could be recognised and diagnosed based on a patient's symptoms. A computer may struggle with this as it may not be properly trained in specialised areas so misdiagnoses may be made. I believe humans are outperforming computers mainly for expert tasks.
2. <u>Discuss in around 250 words, in your opinion, the three most relevant ethical concerns due to the increasing prevalence of AI.</u>

   Firstly, I believe that the first ethical concerns that prevail due to the increasing prevalence of AI is issues with bias and discrimination. AI algorithms are trained on data so tend to contain similar biases to the data they are trained on. An example of this may be facial recognition that is mainly trained using white Caucasian males, having an issue identifying females or people from a different race. This could have detrimental consequences in sectors such as hiring or sentencing decisions.

   Secondly, privacy is another issue that pertains to the increasing prevalence of AI. Personal information can be used by AI but may not be used for the intended purposes and without consent. AI could be used to collect data and be used for surveillance but may infringe on an individual's right to privacy. It may be used to predict political affiliation or sexual orientation which could expose people to harassment or discrimination.

Finally, I believe that job displacement may become an issue. AI is becoming better and better at automating many tasks performed by humans. This would lead to increased productivity but lead to jobs being lost in many sectors. This could be particularly impactful when jobs are taken but not created. Also, this could be impactful on communities and social classes where many of the jobs displaced are localised to a social level or local area.

3.

a.

<u>Give one-line descriptions of the different components of an agent?</u>

An agent must use sensing and perception to perceive their environment through the use of inputs to detect changes in environments.
An agent uses actuators to convert energy into motion and perform the role of controlling the system.
An agent uses effectors which are devices which affect the environment around them.

b.

<u>Would a rational or an irrational agent be more suited to solve a game of Tic Tac-Toe? Why?</u>

A rational agent would be better suited to solve a game of Tic Tac-Toe. This is due to it being a game with defined rules and includes a finite number of moves that are possible. A rational agent select actions that maximise its expected utility. This uses its information of the environment, in this case, prior moves and all possible future moves. An irrational agent may not always choose the most optimal move in the current state of the game.

c.

<u>For the game of Tic-Tac-Toe, what would be the Start state, Legal moves, Cost evaluation, Goal State and the Test for a Goal State?</u>

- The start state would be an empty 3x3 grid.
- The legal moves would be each player taking turns placing their mark (X or O) on one of the empty squares of the grid. The legal moves are the set of all empty squares of the grid.
- The cost evaluation is irrelevant as all moves are equal in value.
- The goal state is when one play places three of their marks (X or O) in a column, row or diagonal. This player is the winner. Or the game ends as a draw when the set of empty square is equal to 0 without the prior condition being met.
- The test for goal state is, if one player has three of their marks in a row, column or diagonal. If there are no more empty squares and no player has won then the goal state is a draw.

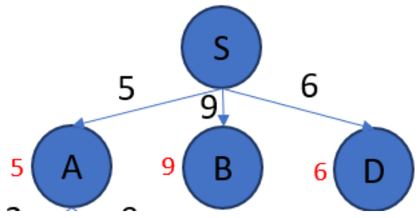
4.

<u>Apply the Uniform Cost Search algorithm to the graph given below, where S is the start state and G1,G2 and G3 are the goal states. Show the sorted list at every step of the algorithm and also show at every step, the solution state space as it grows. Finally, clearly indicate the path cost from the start node to the goal node and the path taken within the solution tree.</u>

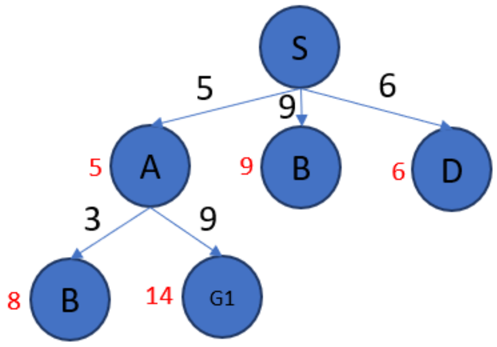
Initially, the visited nodes is set to []

The state space is:

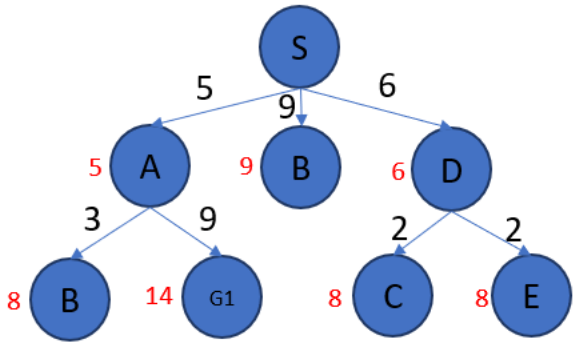
Next, all the nodes that connect to this are added and S is added to visited [S]:



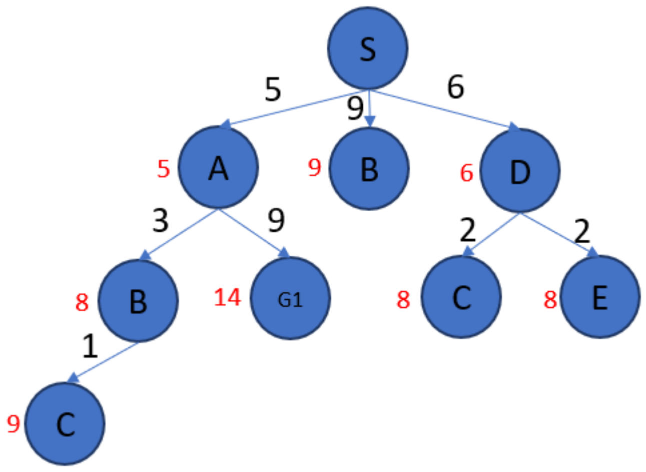
The total cost of the three new paths is calculated and the path to A is the shortest with a length of 5. A is then expanded:



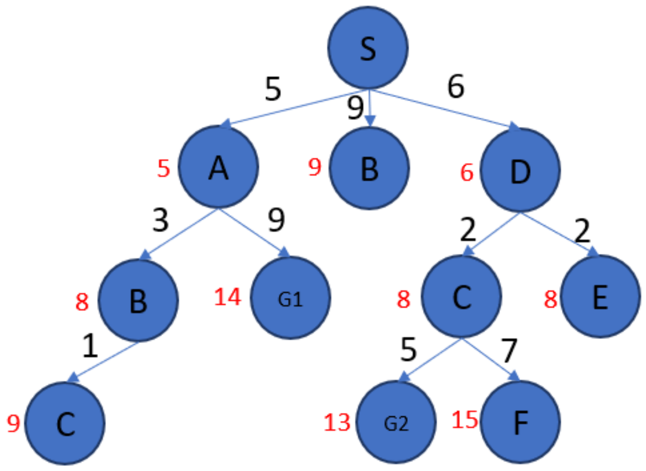
The path costs are calculated and A is added to the visited list [S, A]. Now the next optimal path from the original set is expanded. D:



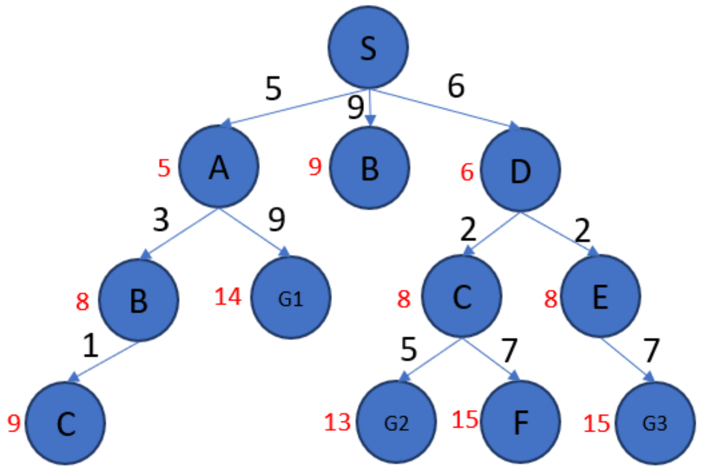
D is now added to the visited list [S, A, D]. There are three paths that all have a cost of 8 now so I will use alphabetical order and expand B next:



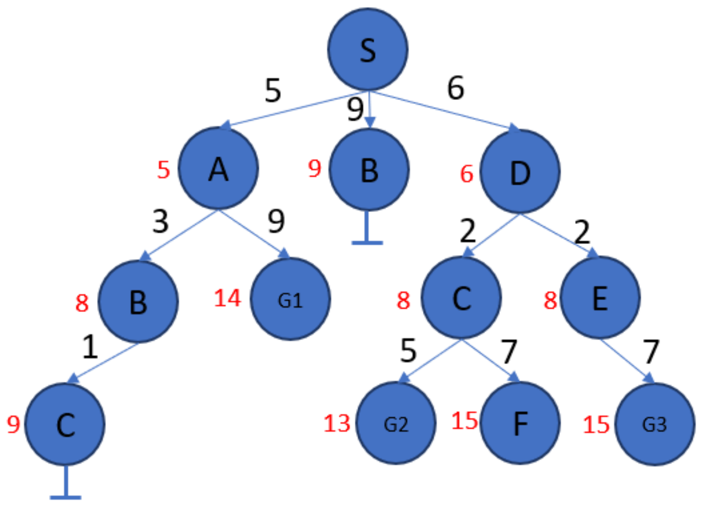
B is now added to the visited list [S, A, D, B]. Now the next value with 8 is search in alphabetical order, so C is expanded:



C is now added to the visited list [S, A, D, B, C]. E is now expanded:



E is now added to the visited list [S, A, D, B, C, E]. All the active nodes are now checked, if they have been visited at a lower cost on a different path then it is treated as a dead end:



G2 is found as a goal state with the lowest cost, being 13. Therefore the final lowest path to a goal state is S→D→C→G2 with a length of 13.

5.

Write the A* algorithm based on your understanding. Explain why should the A* heuristic always underestimate the cost from a vertex to a goal node. How will the algorithm behave if estimate is very close to the actual value of the cost? Finally explain how will the algorithm behave if the heuristic overestimates?

Pseudocode:

```
function A*(start, goal, h):
    openSet := {start}
    cameFrom := {}
    gScore := {}   // cost from start to node
    gScore[start] := 0
    fScore := {}   // estimated total cost from start to goal
    fScore[start] := h(start, goal)

    while openSet is not empty:
        current := node in openSet with lowest fScore
        if current = goal:
            return reconstructPath(cameFrom, current)

        openSet.remove(current)
        for neighbor in current.neighbors:
            tentativeGScore := gScore[current] + cost(current, neighbor)
            if tentativeGScore < gScore[neighbor]:
                cameFrom[neighbor] := current
                gScore[neighbor] := tentativeGScore
                fScore[neighbor] := gScore[neighbor] + h(neighbor, goal)
                if neighbor not in openSet:
                    openSet.add(neighbor)

    return failure
```

```
function reconstructPath(cameFrom, current):
    totalPath := [current]
    while current in cameFrom:
        current := cameFrom[current]
        totalPath.prepend(current)
    return totalPath
```

The A* algorithm must always be admissible as it ensures that it will always find the shortest path available. This is due to the algorithm potentially searching sub-optimal paths for the end-state.
If the estimated value is close to the actual value then the algorithm will still work but it may be slower. This is because more nodes may be evaluated when finding the optimal path so it could increase the search time.
If the heuristic overestimates the cost then a solution may be found but it may not be the most optimal. This may be because the algorithm prematurely discards nodes and is known as admissibility.

6.

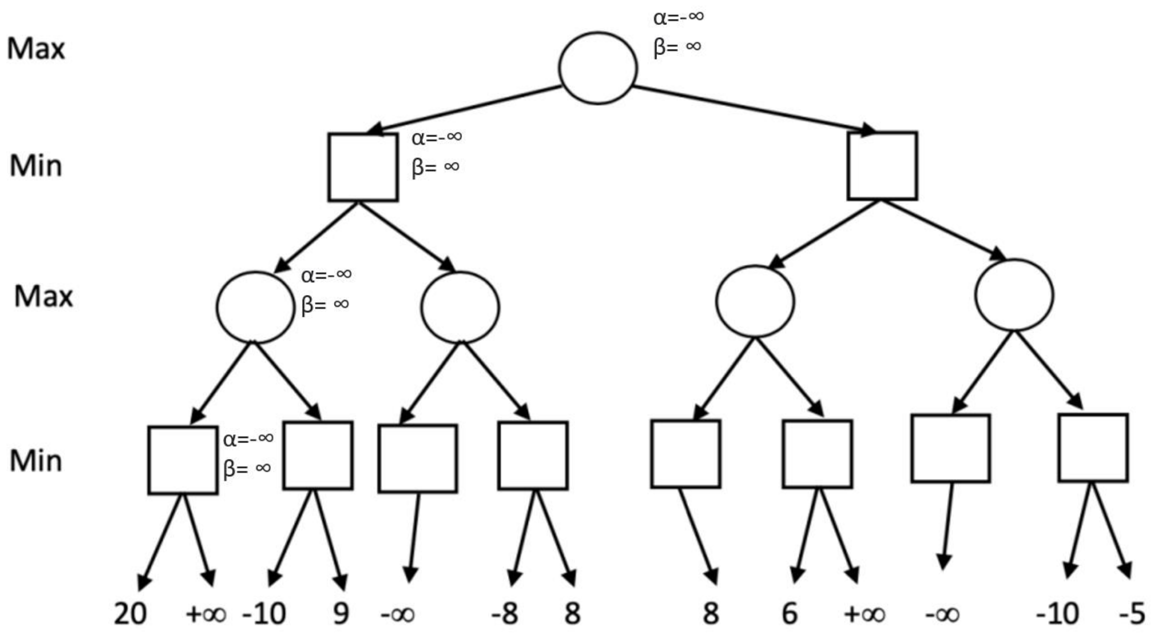
Determine whether the following sentences are valid, satisfiable, unsatisfiable, contradiction or neither. Justify your answer.

a. $Smoke \Longrightarrow Smoke$
Valid, as propositional logic dictates when both the antecedent and consequent are both true.
b. $Smoke \Longrightarrow \mathrm{Fire}$
Satisfiable as when Smoke is false or when Fire is true, the implication of this is true.
c. $(Smoke{\Longrightarrow}Fire){\Longrightarrow}(\neg Smoke{\Longrightarrow}\neg Fire)$
This can be simplified using multiple steps:
$(\neg Smoke \vee Fire){\Longrightarrow}(Smoke \vee \neg Fire)$
(Smoke ∧ Fire) V $(Smoke \vee \neg Fire)$
$(Smoke \vee \neg Fire)$ ∧ True
Fire ⟹ Smoke
Therefore, this is Satisfiable as the implication is true.
d. $Smoke \vee Fire \vee \neg Fire$
This is a valid statement as the statement "Fire V ¬Fire" is always true.
e. $Big \vee Dumb \vee (Big{\Longrightarrow}Dumb)$
This can be change to:
Big V Dumb V ¬Big V Dumb
Big V ¬Big V Dumb
Either way, if big is true or false, this is a valid statement.

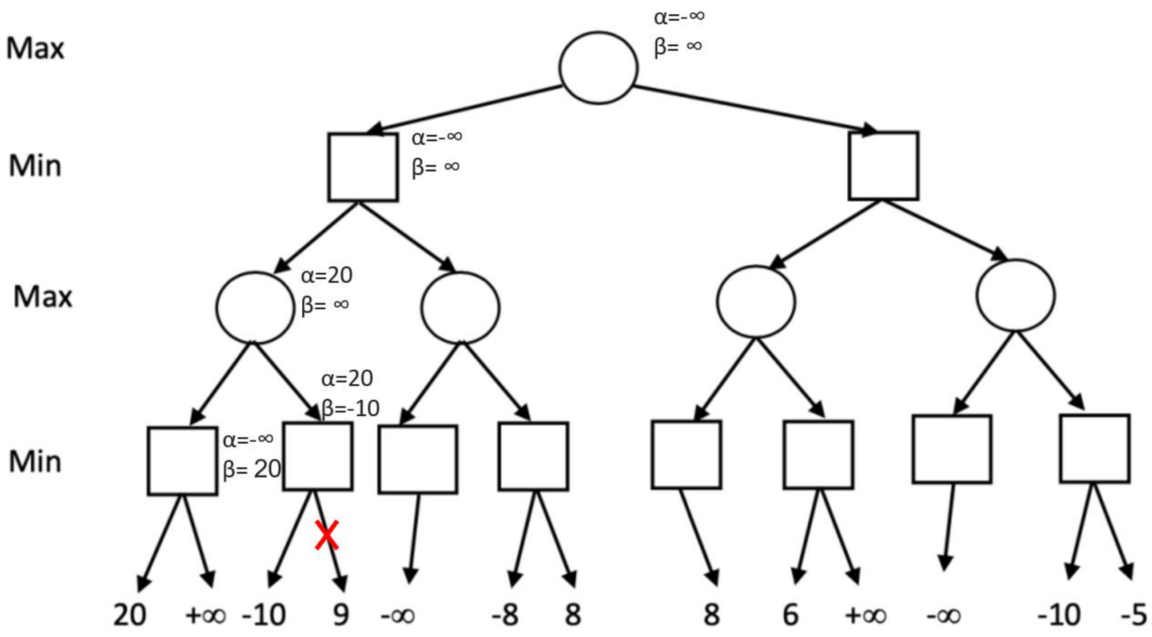
7.

For the minimax tree given below, explain which branch would constitute the MAX player's best move. Also indicate if any branches can be pruned. Justify these prunes by showing the values of α and β at every level.

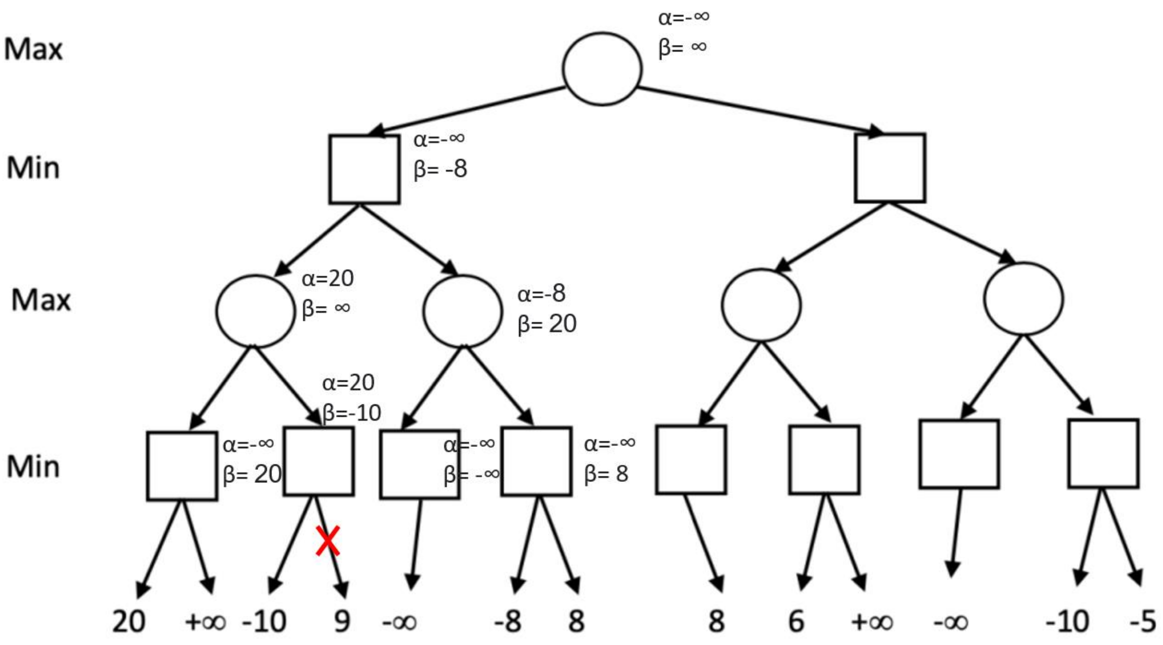
I will be using minmax searching with alpha beta pruning to complete this.

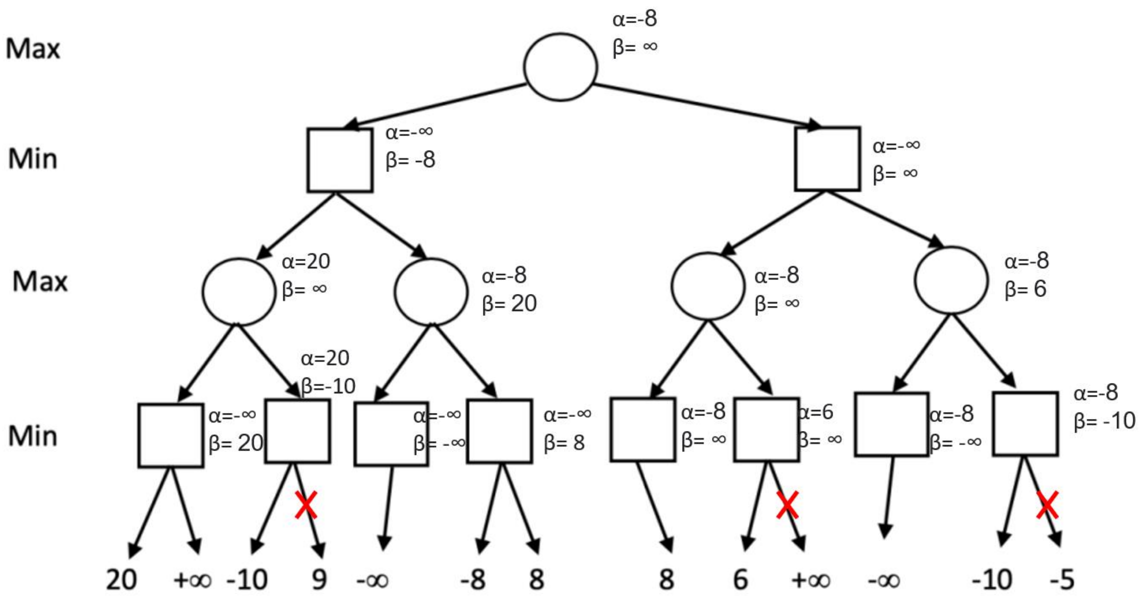


Max
Min
Max
Min
α=-∞
β= ∞
α=-∞
β= ∞
α=-∞
β= ∞
α=-∞
β= ∞
20
+∞
-10
9
-∞
-8
8
8
6
+∞
-∞
-10
-5

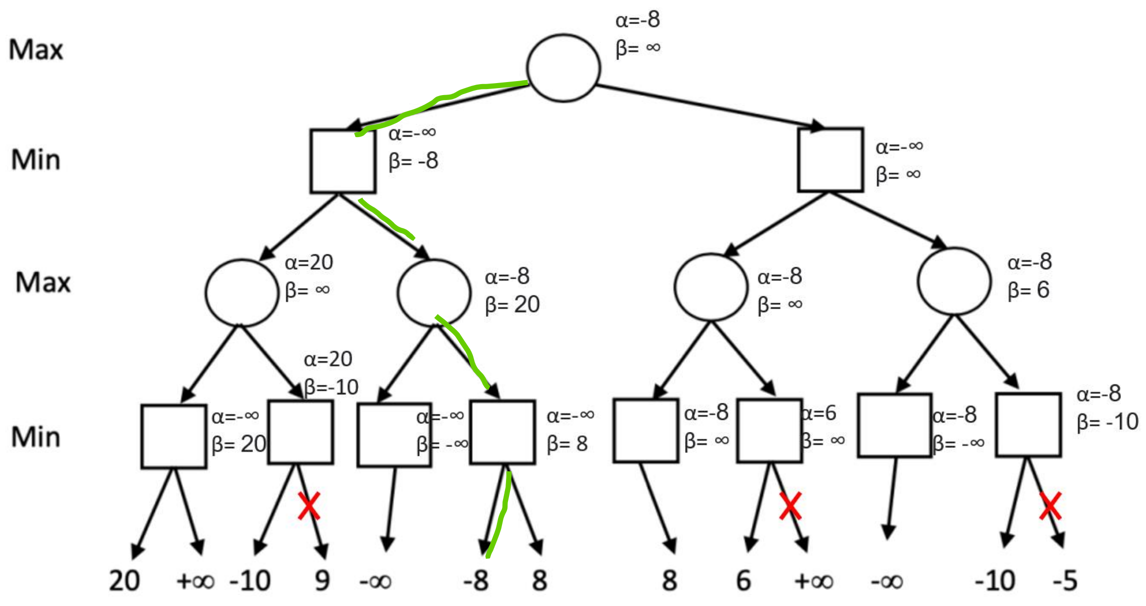


Max
Min
Max
Min
α=-∞
β= ∞
α=-∞
β= ∞
α=20
β= ∞
α=20
β=-10
α=-∞
β= 20
20
+∞
-10
9
-∞
-8
8
8
6
+∞
-∞
-10
-5





The best move for the max player would be


Max
Min
Max
Min
α=-8
β= ∞
α=-∞
β= -8
α=-∞
β= ∞
α=20
β= ∞
α=-8
β= 20
α=-8
β= ∞
α=-8
β= 6
α=-∞
β= 20
α=20
β=-10
α=-∞
β= -∞
α=-∞
β= 8
α=-8
β= ∞
α=6
β= ∞
α=-8
β= -∞
α=-8
β= -10
20
+∞
-10
9
-∞
-8
8
8
6
+∞
-∞
-10
-5