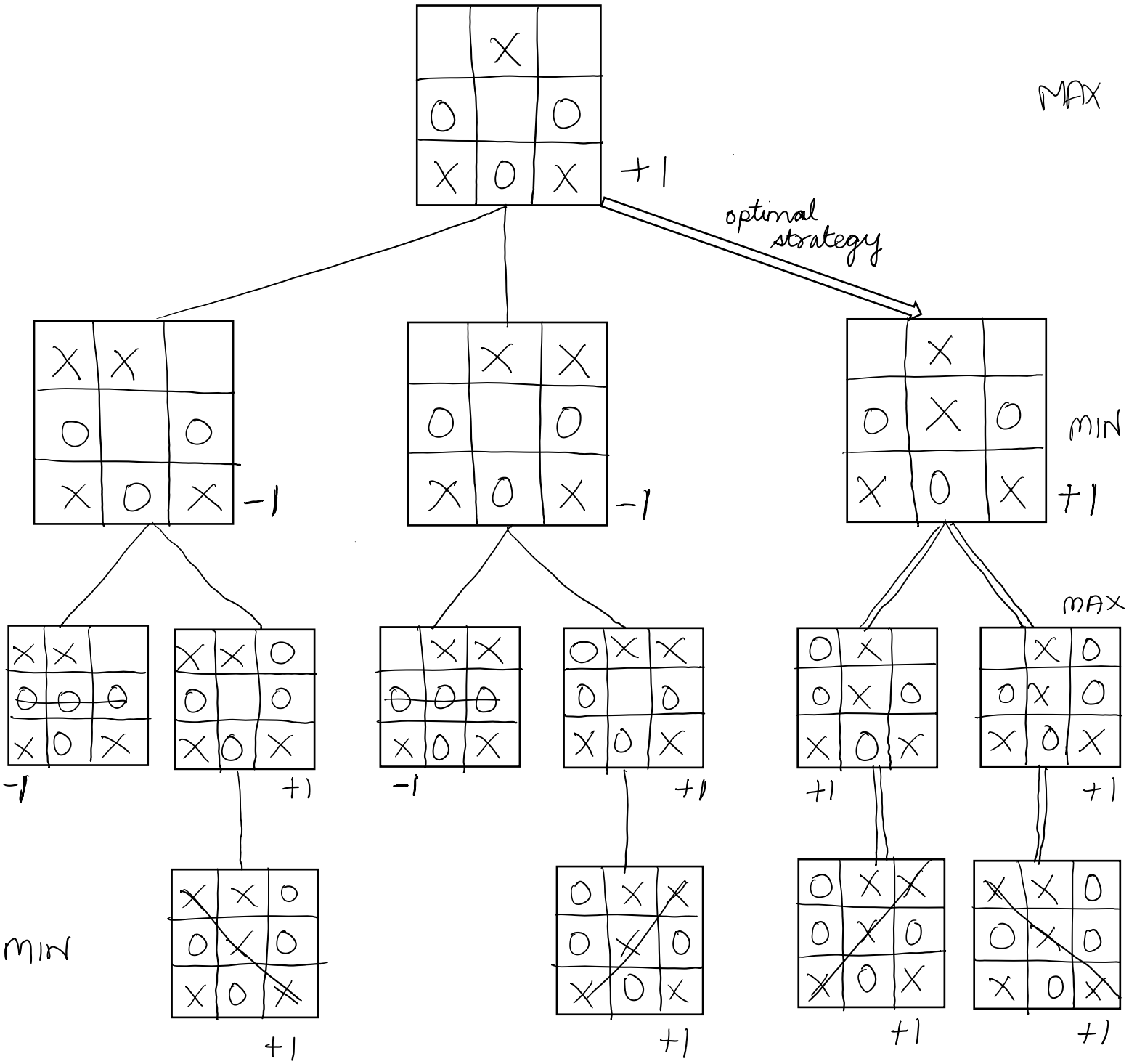
Artificial Intelligence
Assignment - 3

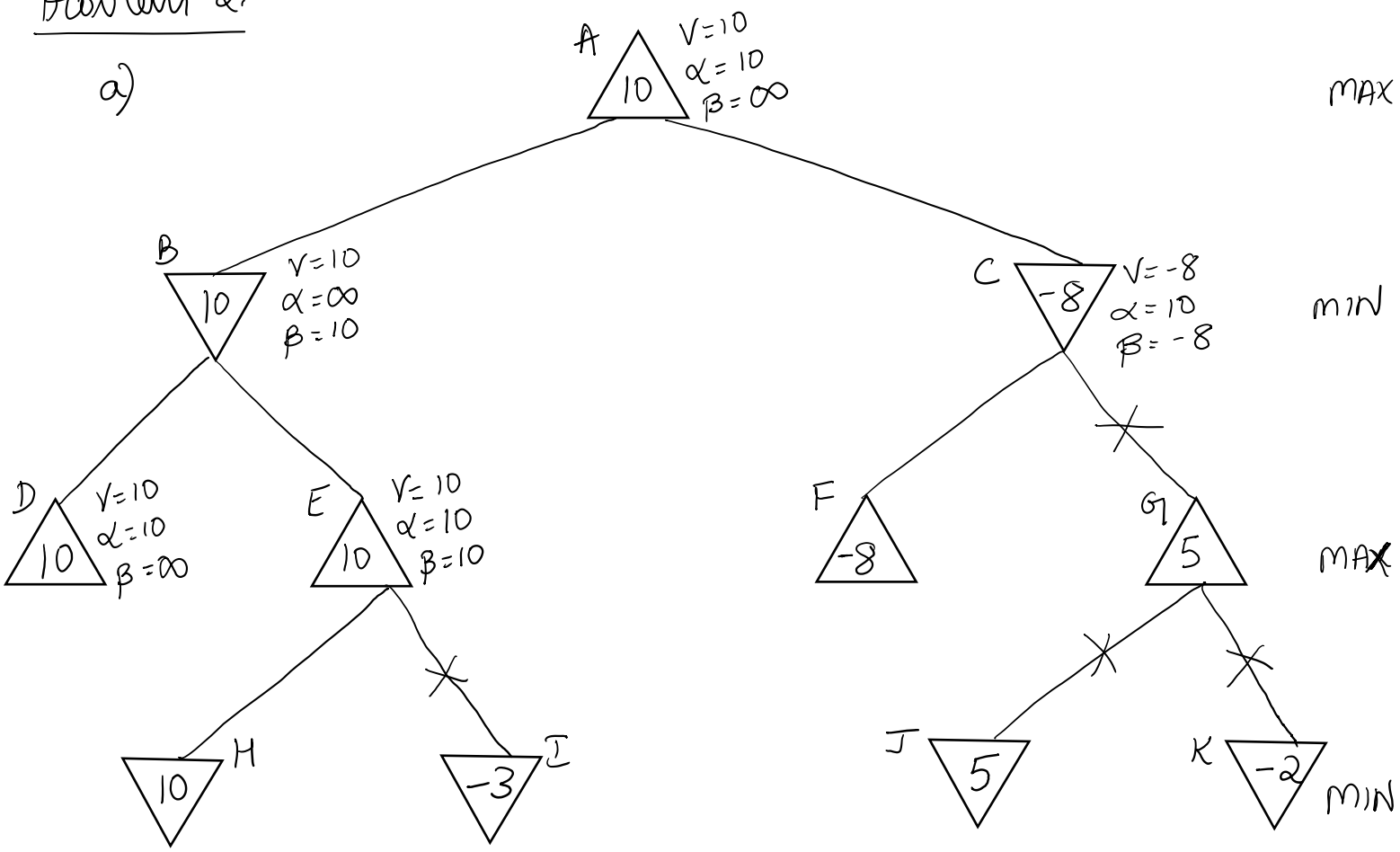
SUNDESH RAJ
1001633297

Problem 1:-



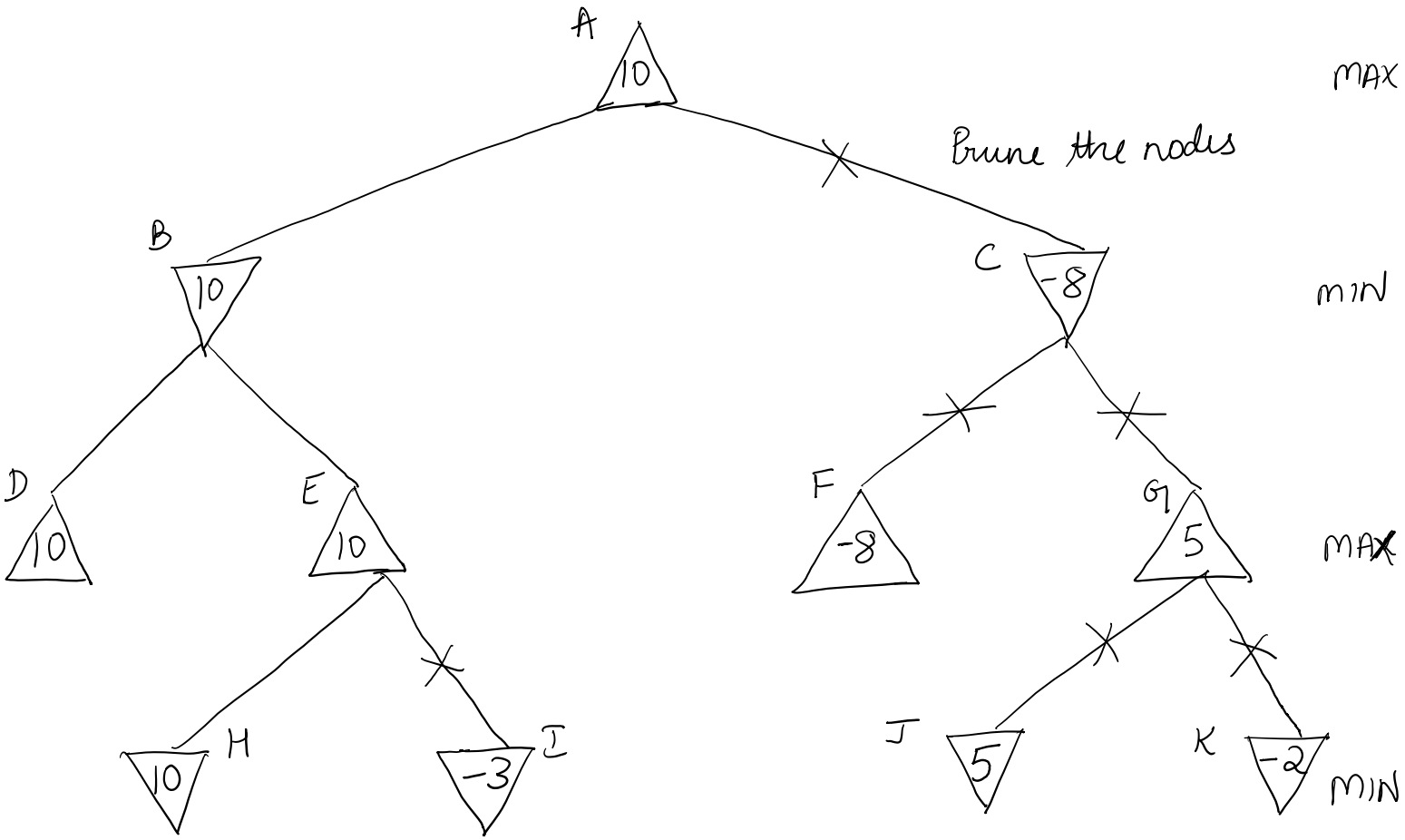
Problem 2:-

a)



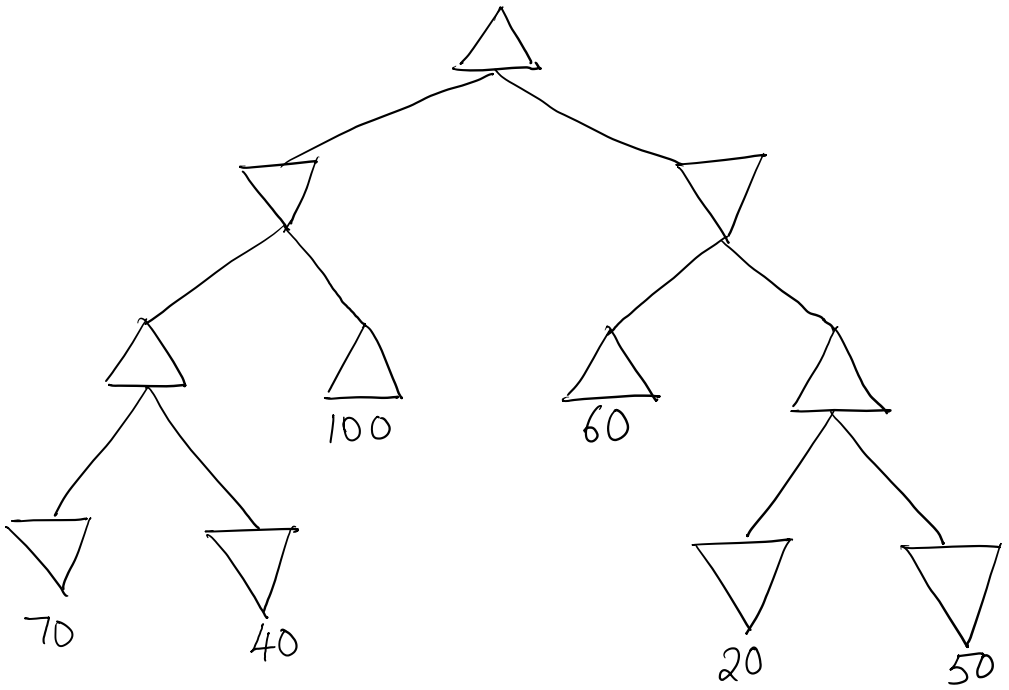
Node I will be pruned since its value is less than 10. Similarly nodes J, K and G would also be pruned.

(b) Additional Information of minimum utility = 10, which means it is not possible for max player to get an outcome greater than 10.



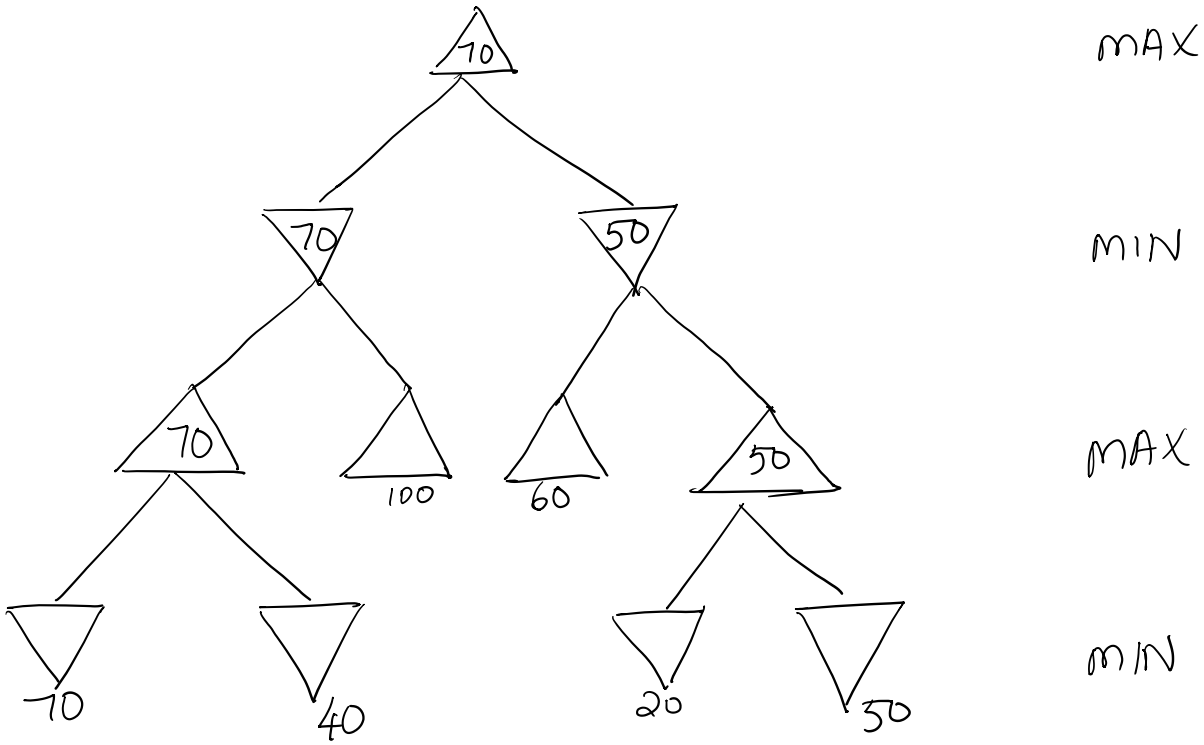
We can improve the efficiency of alpha-beta search by using the additional information of maximum utility 10. By looking for utility values of MAX player until we get utility 10. Once we get the utility value 10, i.e. 10 is the maximum utility value and so there cannot be utility value higher than this & so we prune the rest of the node as shown in the above figure as this is the maximum utility value.

Problem 3:-



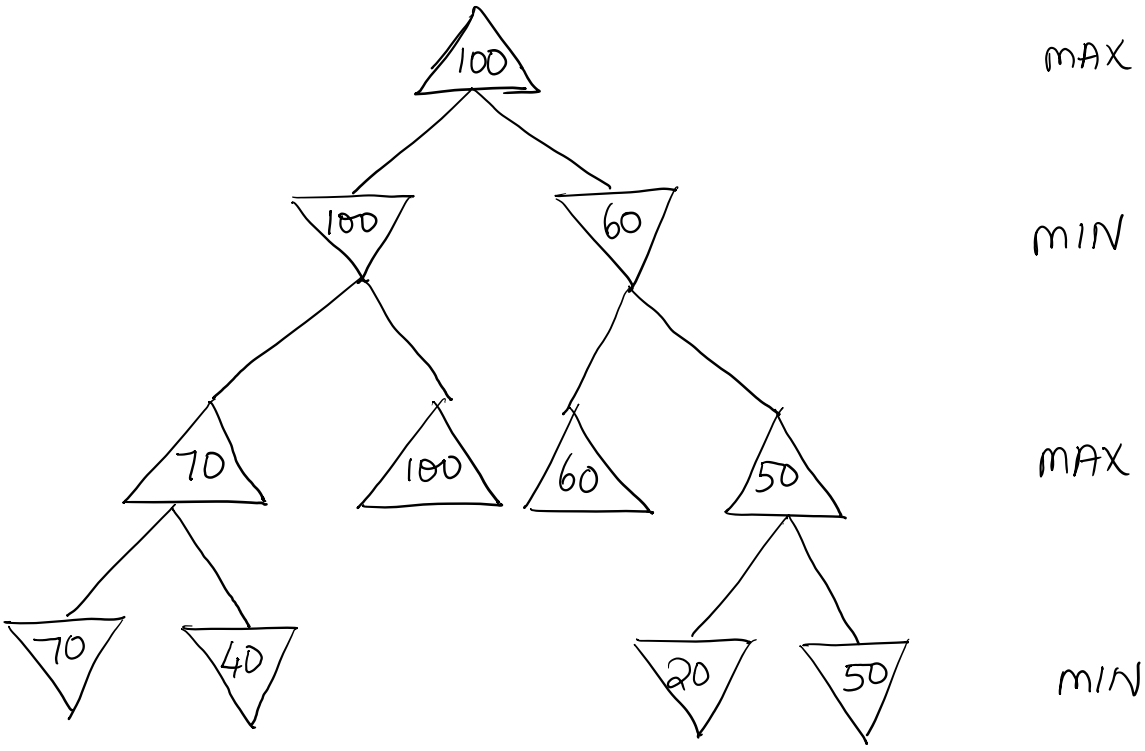
Solution:- We are the MAX player and we follow the MINIMAX algorith against the opponent.

(i) Worst possible outcome for MAX player will be when the MIN player (opponent) will also play optimally



The MIN player will not allow MAX player to gain more points as both follow MINIMAX and hence the worst possible outcome for MAX player will be 70.

(ii) Now, the best possible outcome for a MAX player will be when MIN will not play optimally.

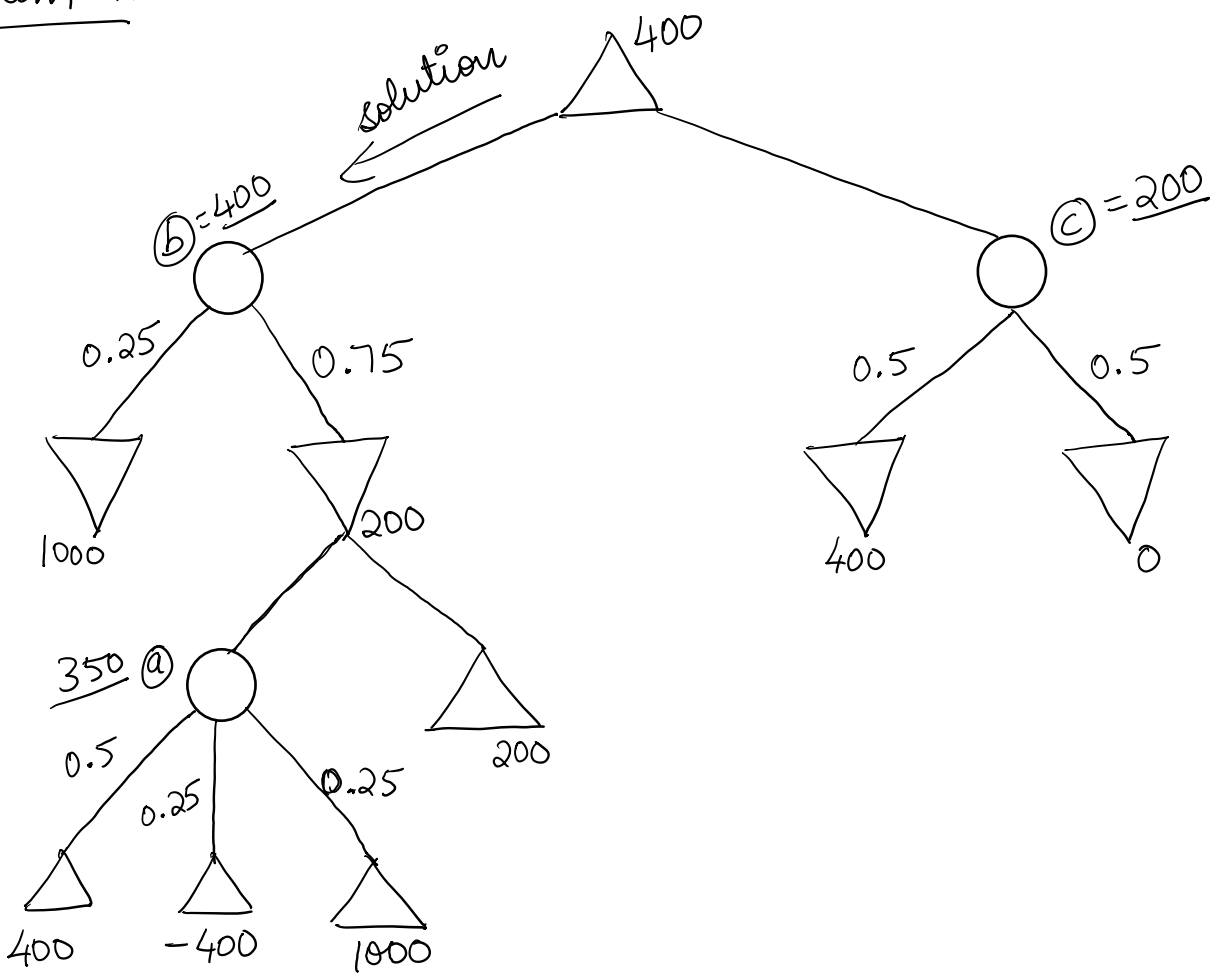


The best possible outcome for playing a full game for MAX will be <u>100</u>.

In the above search tree, MIN player does not play optimally, as a result MIN player selects a node with utility value [100] instead of node with utility value [70], as MIN is not playing optimally.

Similarly, it chooses node with utility value [60] instead of node with utility value [50].

Problem 4:-



At node ⓐ → $(0.5)(400) + (0.25)(-400) + (0.25)(1000)$

$\Rightarrow 200 - 100 + 250 \Rightarrow \underline{350}$

At node ⓑ → $(0.25)(1000) + (0.75)(200)$

$\Rightarrow 250 + 150 \Rightarrow \underline{400}$

At node ⓒ → $(0.5)(400) + (0.5)(0)$

$\Rightarrow 200 + 0 \Rightarrow \underline{200}$

The algorithm chooses the path with node ⓑ

Problem 5 :-

The opponent is a supercomputer called DeepGreen.
DeepGreen does not use MINIMAX
Library function DeepGreenMove(s), takes any state s as argument and returns the move that DeepGreen will choose for that state s (i.e it returns the state resulting from opponents move.

As minimax algorithm is a 2 player deterministic game of perfect information we will use minimax algorithm to compete. We will be the MAX player while the opponent will be the MIN player

Pseudocode :-

function MINIMAX_DECISION(state) returns an action
input → state, current state in the game
return a in Actions(state) maximizing DeepGreenMove (Result(a, state))

function MAX_VALUE(state) returns a utility value
if TERMINAL_TEST(state) then return Utility[state]
$V \leftarrow -\infty$
for a, s in SUCCESSORS(state)
do $V \leftarrow MAX(V, DeepGreenMove(s))$
return V;