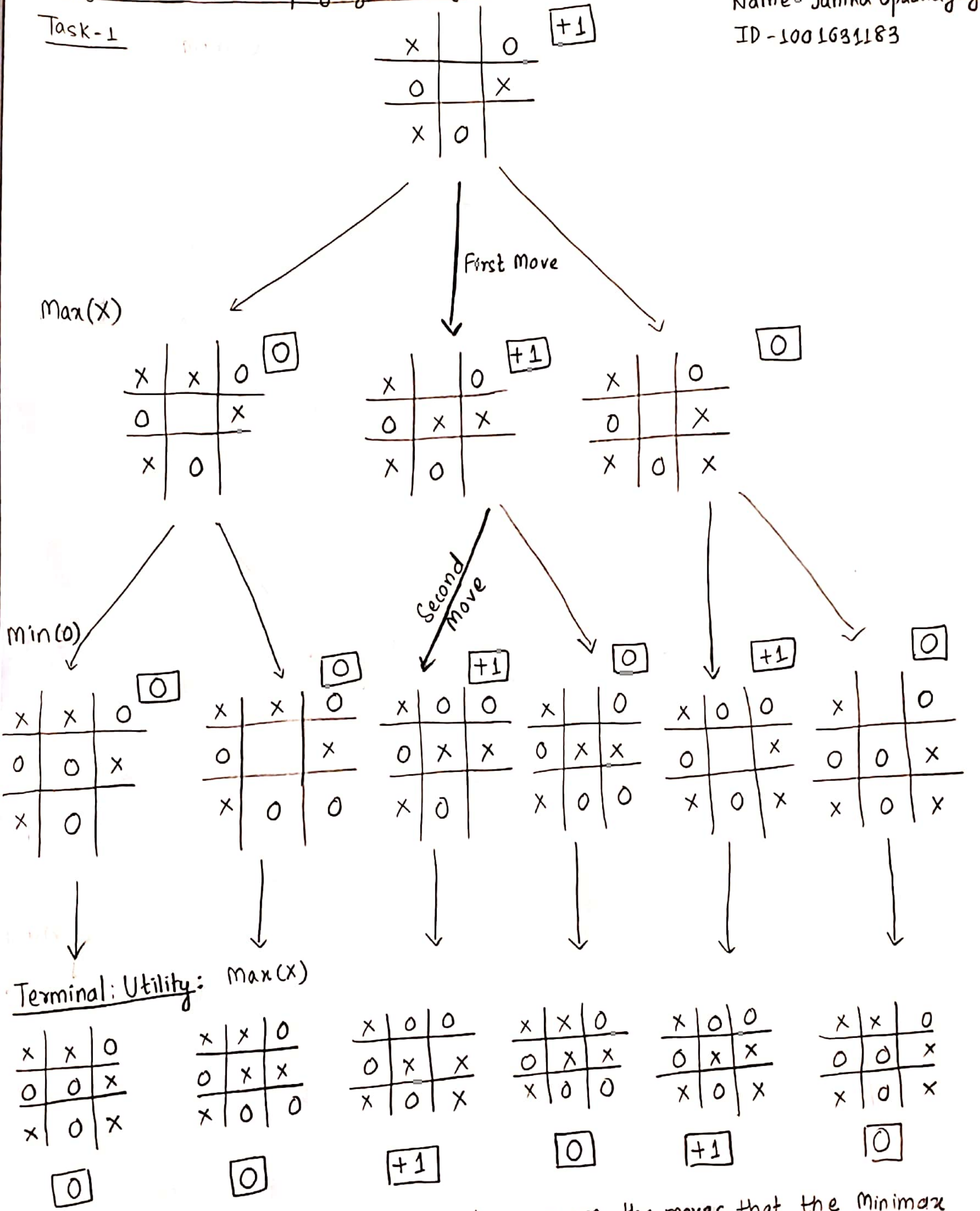
Assignment-3 (Game playing and Logic)

Task-1

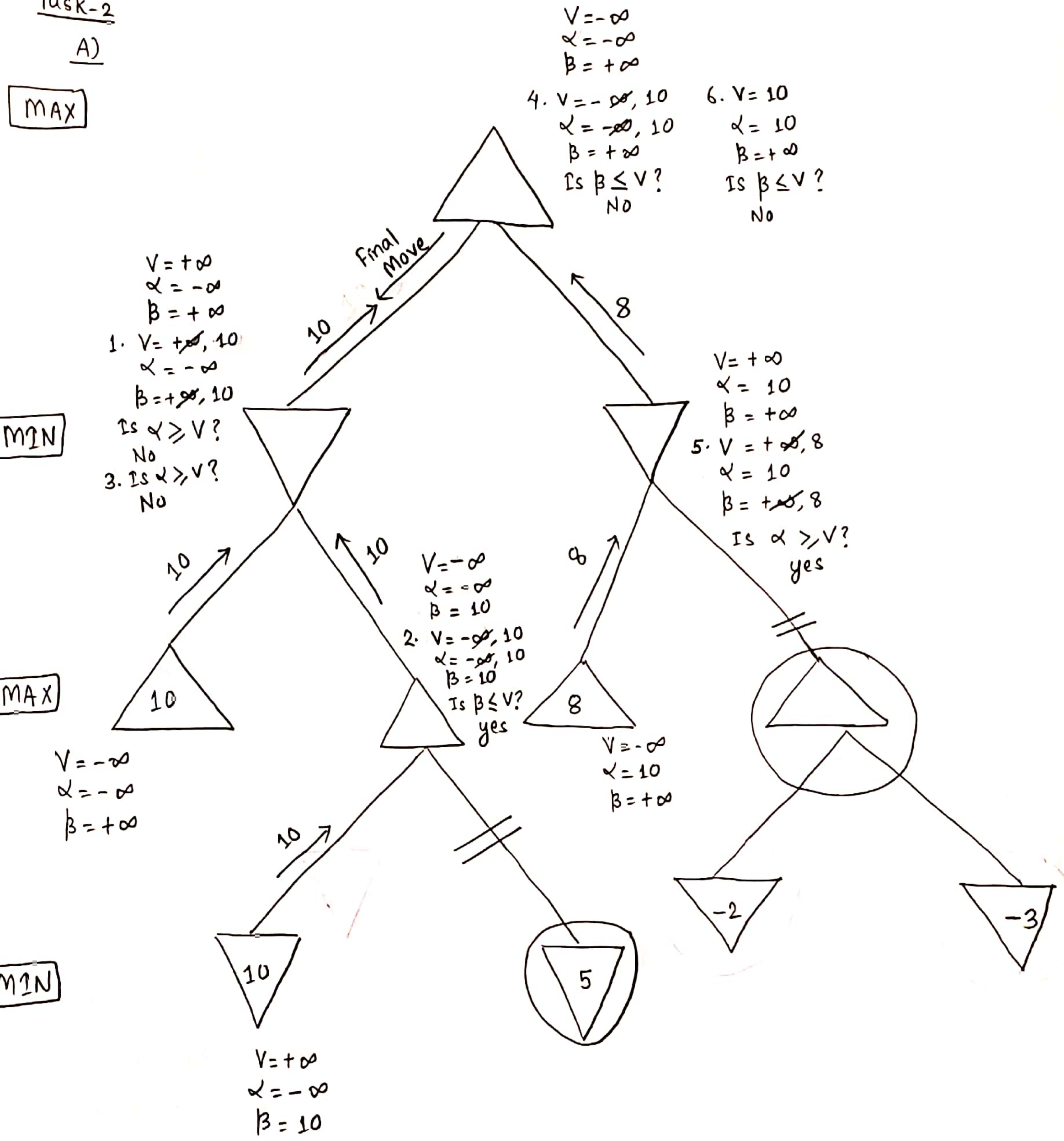
Name - Yunika Upadhayaya
ID - 1001631183



Here, Marked first and second moves are the moves that the Minimax algorithm decides to play for X.

Task-2

A)

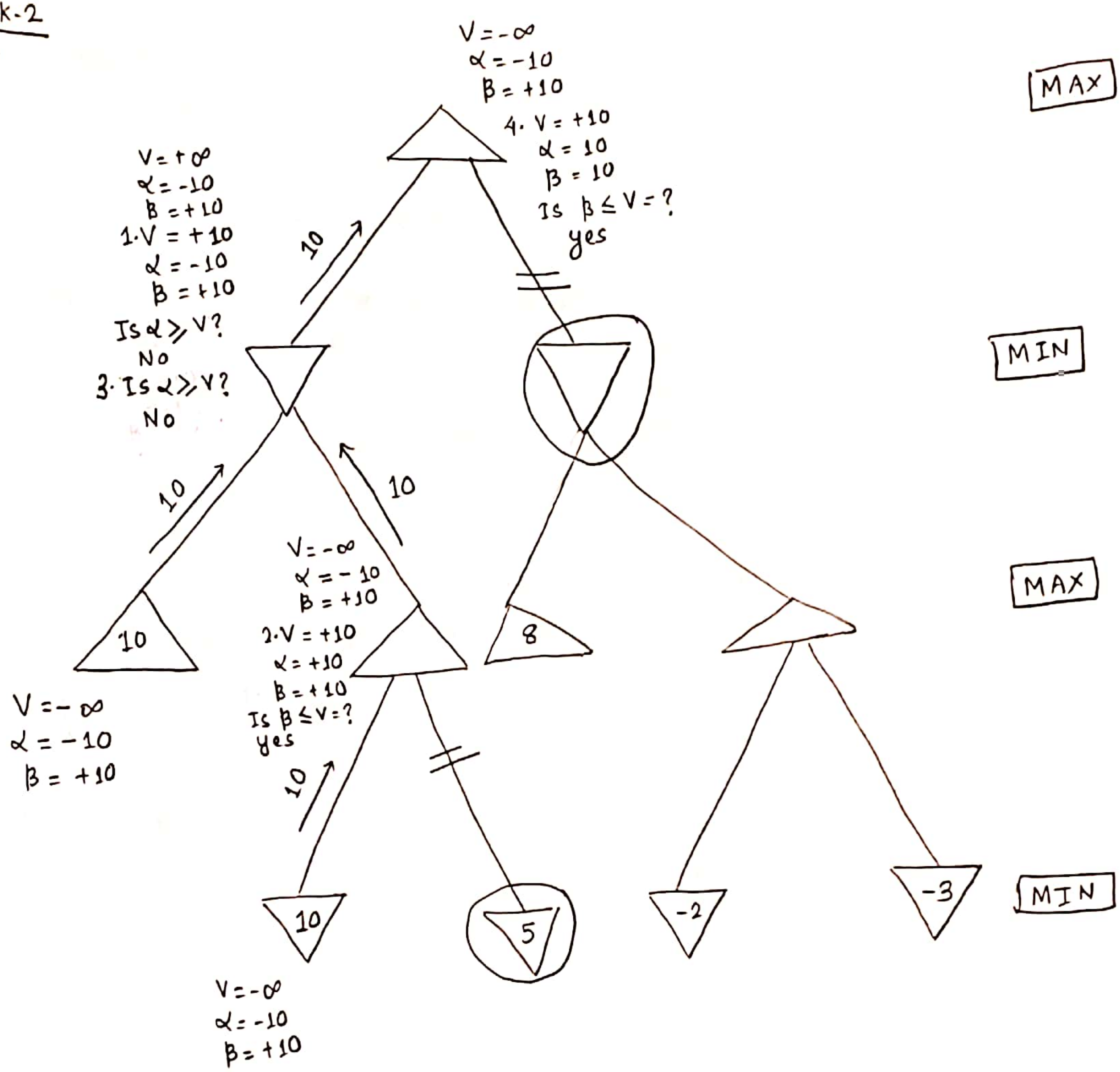


Notes:

1) Circle nodes are pruned notes.
2) The steps are numbered as 1, 2, 3 ..... so on.

Task-2

B)



Note:

1. Circled nodes are pruned nodes.
2. The steps are numbered as 1, 2, 3, and so on.

Task-3

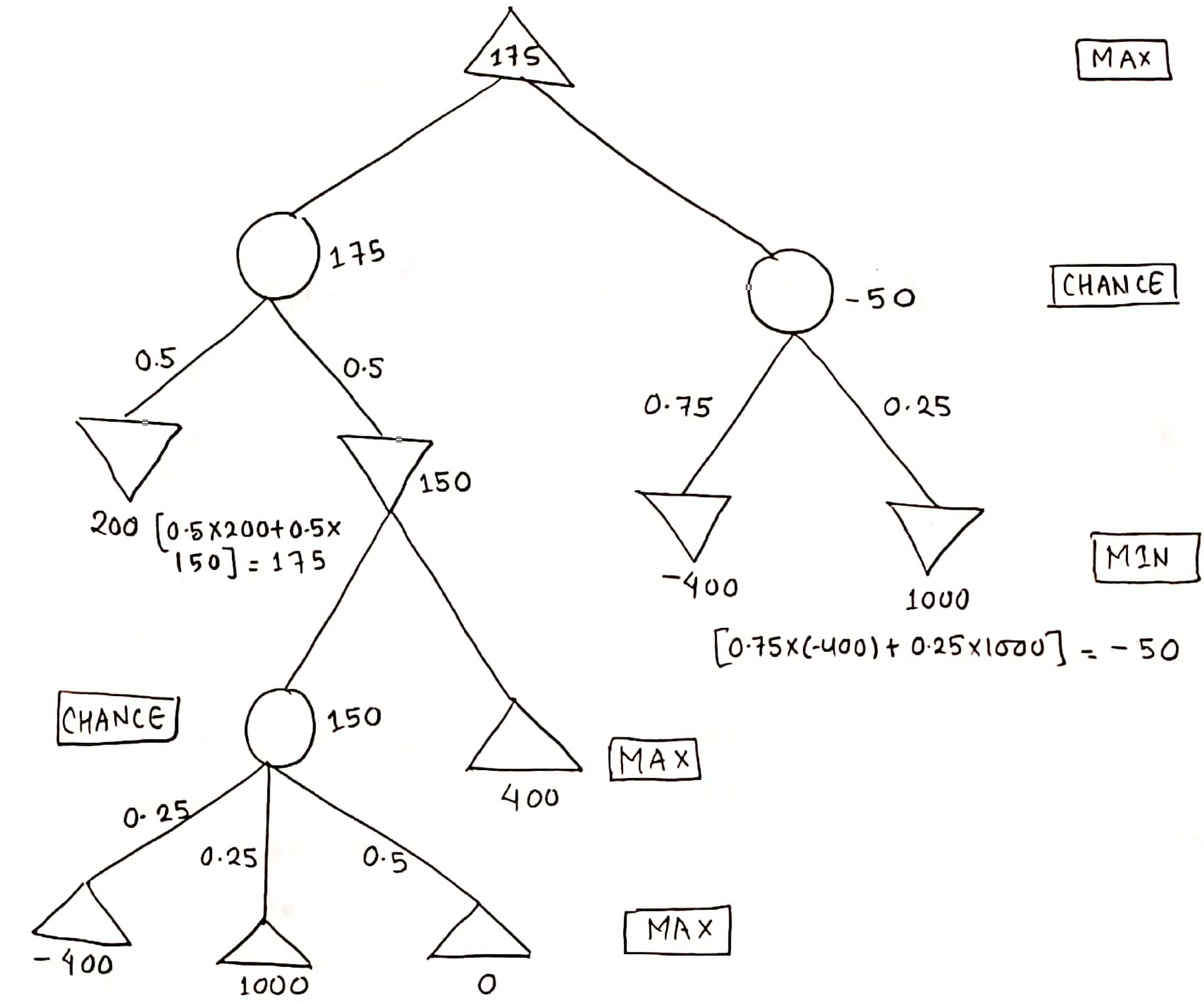
function MINIMAX_DECISION (state) returns an action
inputs: state, current state in game
return the a in ACTIONS (state) maximizing MIN_VALUE (RESULT (a, state))

function MAX-VALUE (state) returns a utility value
if TERMINAL-TEST (state) then return UTILITY (state)
$v \leftarrow -\infty$
for a, x in SUCCESSORS (state) do v ← MAX (V, MIN-VALUE (X))
return V

function MIN-VALUE (state) returns a utility value
if TERMINAL-TEST (state) then return UTILITY (state)
V ← MAX-VALUE (DeepGreenMove (state))
return V

The advantage of this algorithm is that we will know opponent's moves in all the state but Minimax assumes that the opponent's move is always optimal.

Task-4

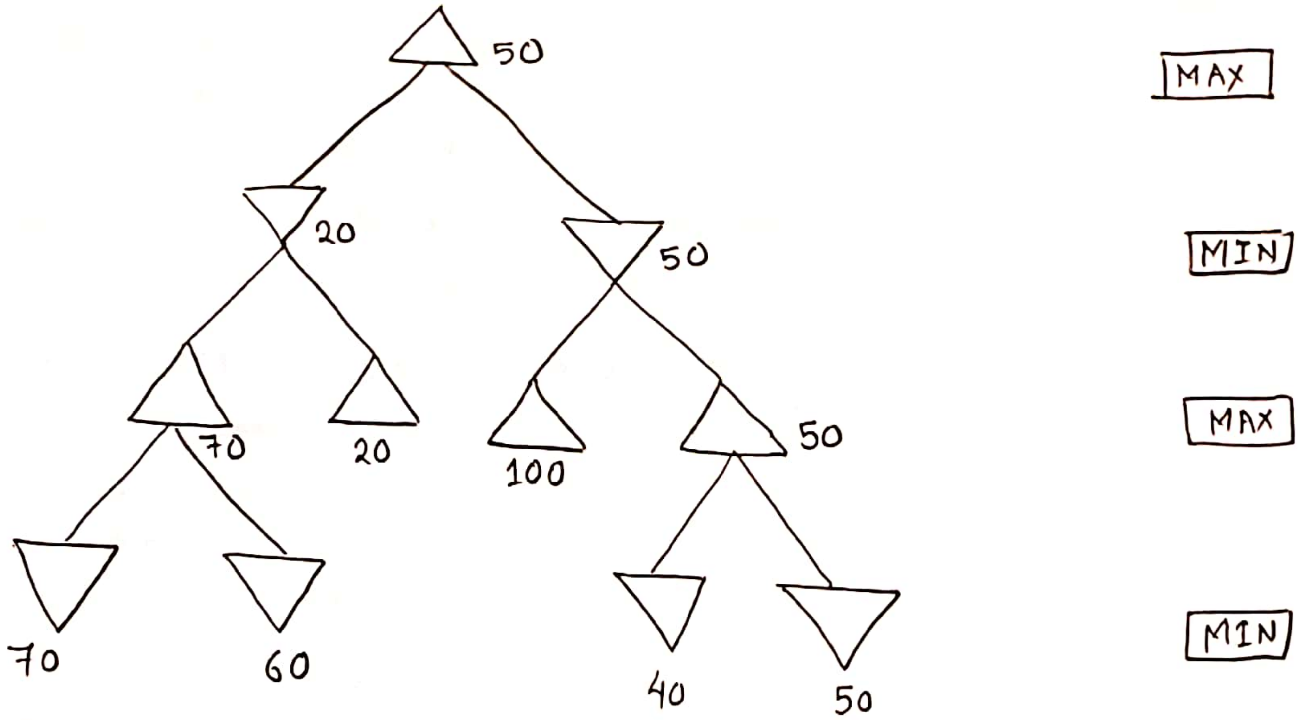


$[0.25\times(-400) + 0.25\times1000 + 0.5\times0] = 150$

Highest possible outcome = 1000
Lowest possible outcome = -400

The algorithm will perform the action that provides the most expected payoff for the player. Thus, the expected payoff is 175.

Task-5



Since, we are the MAX player and we are following the MINIMAX algorithm to play a full game against an opponent. But, we do not know what algorithm the opponent uses.

Our goal is to maximize the value. The best possible outcome of playing the full game for the MAX player is 100. MAX player will definitely chose the highest value (i.e. right branch), but MIN player can also chose the left branch (with utility 100), since we do not what algorithm MIN is using. The worst possible outcome of playing the full game for the MIN player is 50. This will happen if min player also uses MINMAX algorithm and tries to minimize our value.

Task-6

```
function  CHECK_EQUIVALENCE (KB1, KB2) returns true or false
    inputs: KB1, first Knowledge base in propotional logic
            KB2, second Knowledge base in propotional logic

    symbols ← symbols in KB1 and KB2 are proposition symbols
    return MATCH (KB1, KB2, symbols, {})

function MATCH (KB1,KB2, symbols,model) return true or false
    if EMPTY? (symbols) then
          if Plogic (KB1,model)
          then
             return Plogic (KB2,model)
          else
              return true
    else
        do
        P ← FIRST (symbols)
        rest ← REST (symbols)
        return (MATCH (KB1,KB2,rest,model U {P= true})
                and MATCH (KB1, KB2,rest,model U{P= false})
                and MATCH (KB2, KB1, rest,model U {P= true})
                and MATCH (KB2, KB1, rest, model U {P= false}))
```

Note:

If a sentence is within a model. then Plogic () will return true. A partial model (some symbol assignments) is represented by variable model.

Task-7

a) We know that, KB entails S1 only if all the models of KB generates true value for S1. There are three models for KB that is D(A = True, B = True, C = True), D(A = True, B = False, C = True), and D(A = False, B = False, C = True). For these same data models, S1 is also True. Therefore, we can say that KB entails S1.

b) First, lets figure out the values of NOT (KB) and NOT (S1)

KB	NOT(KB)	S1	NOT (S1)
True	False	True	False
False	True	True	False
True	False	True	False
False	True	True	False
False	True	False	True
False	True	False	True
True	False	True	False
False	True	False	True

NOT (KB) will entail NOT(S1) only when True values of NOT(KB) gives True values of NOT(S1). From the above table, we can see that there are some true values of NOT(KB) for which NOT (S1) are false. Thus, NOT (KB) does not entail NOT(S1).

Task-8

From the given information, we construct the truth table as follows:

A	B	C	D	KB
True	True	True	True	True
True	True	True	False	True
True	True	False	True	True
True	False	True	True	False
False	True	True	True	True
False	False	True	True	True
False	True	False	True	True
False	True	True	False	True
True	False	False	True	True
True	True	False	False	True
True	False	True	False	True
True	False	False	False	True
False	False	False	False	True
False	True	False	False	True
False	False	False	True	True
False	False	True	False	False

First case: when A is true, B is false, C is true, D is true, knowledge base is false. Thus, Knowledge base is true, when A is false, B is true, C is false, and D is false. Then, CNF is:

$$(\neg A \vee B \vee \neg C \vee \neg D)$$

Second case: When A is false, B is false, C is true, D is false, Knowledge base is false. Thus, knowledge base is true, when A is true, B is true, C is false, and D is true. Then, CNF is:

$$(A \vee B \vee \neg C \vee D)$$

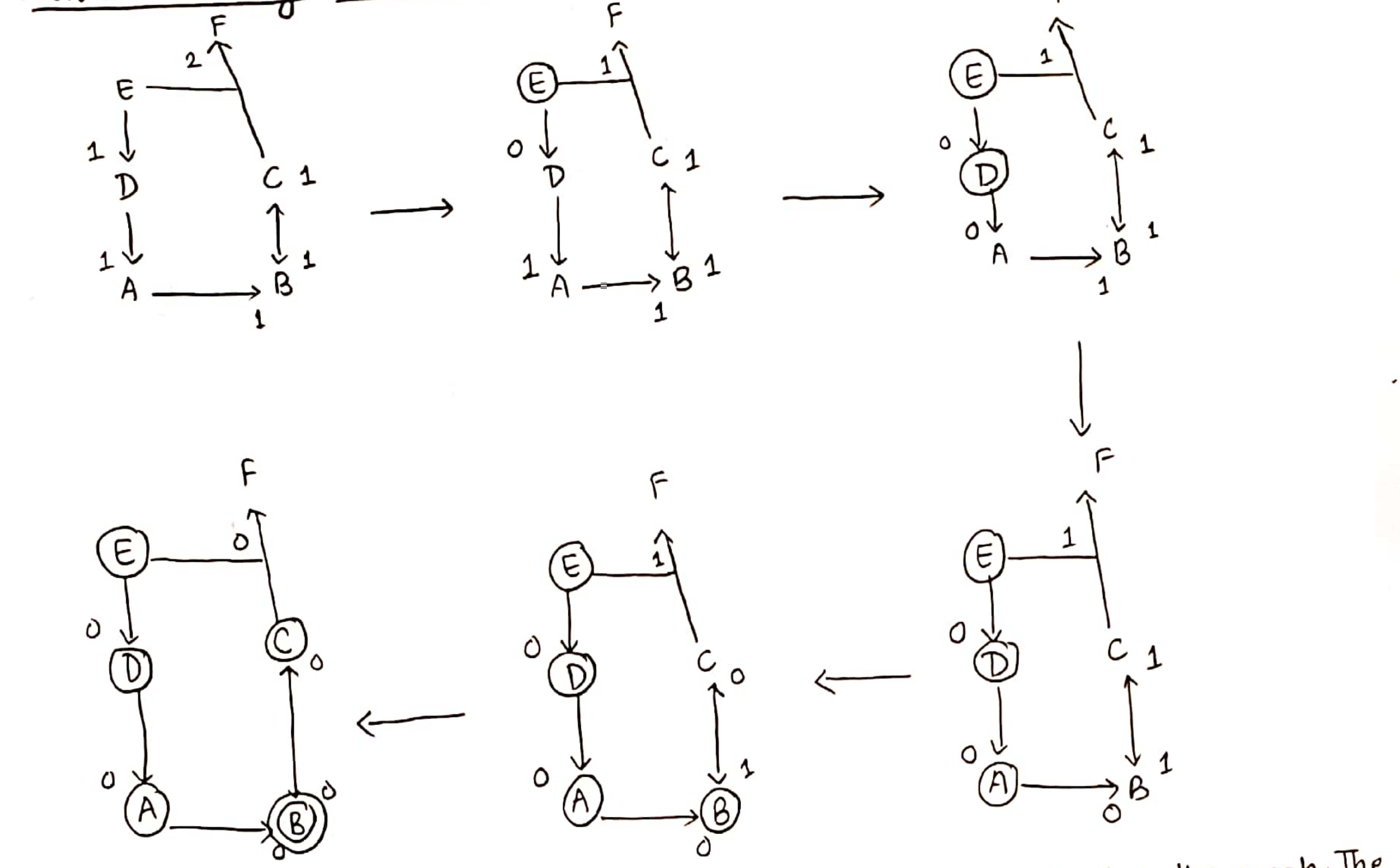
Combining case(I) & (II):

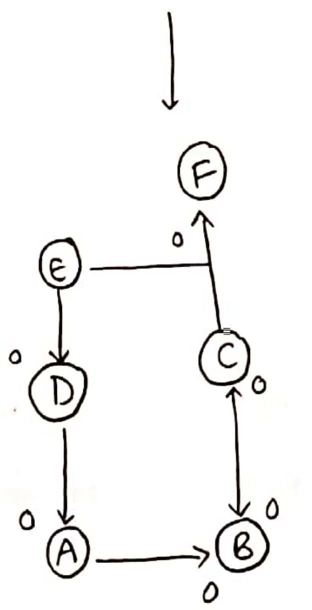
$$\Rightarrow (\neg A \vee B \vee \neg C \vee \neg D) \wedge (A \vee B \vee \neg C \vee D)$$

Scanned with CamSca

Task-9

i) Forward chaining: (1st Method)





Here, At first all the Knowledge based is plotted in the graph. The values are assigned for each arrow based on given knowledge base. Then, values that are true are circled and the connections from that value are reduced by 1. Here, connection to 'E' is reduced by 1. Now, D is true and it is circled and the connection 'A' from 'D' is reduced by 1. Then, 'A' is circled and it is true. The connection from 'A' which is 'B' is reduced by 1. B is circled and true now. The connection from 'B' which is 'C' is reduced by 1 which is then circled and is true. Then, the connections from 'C' which are 'B' and 'F' are reduced by 1 and becomes '0'. At this point, B is visited again, though it was already visited before. F is circled and true which is our final state.

Therefore, using forward chaining we can say that the above Knowledge base entails 'F'.

Forward Chaining (2nd method):

KB

$A \Rightarrow B$

$B \Leftrightarrow C$

$D \Rightarrow A$

$E \Rightarrow D$

C AND $E \Rightarrow F$

E

Using logical equivalence:

$A \Rightarrow B$

$(B \Rightarrow C) \wedge (C \Rightarrow B)$

$D \Rightarrow A$

$E \Rightarrow D$

$(C \wedge E) \Rightarrow F$

Using forward chaining:

$$\frac{E \quad E \Rightarrow D}{D}$$

$$\frac{D \quad D \Rightarrow A}{A}$$

$$\frac{A \quad A \Rightarrow B}{B}$$

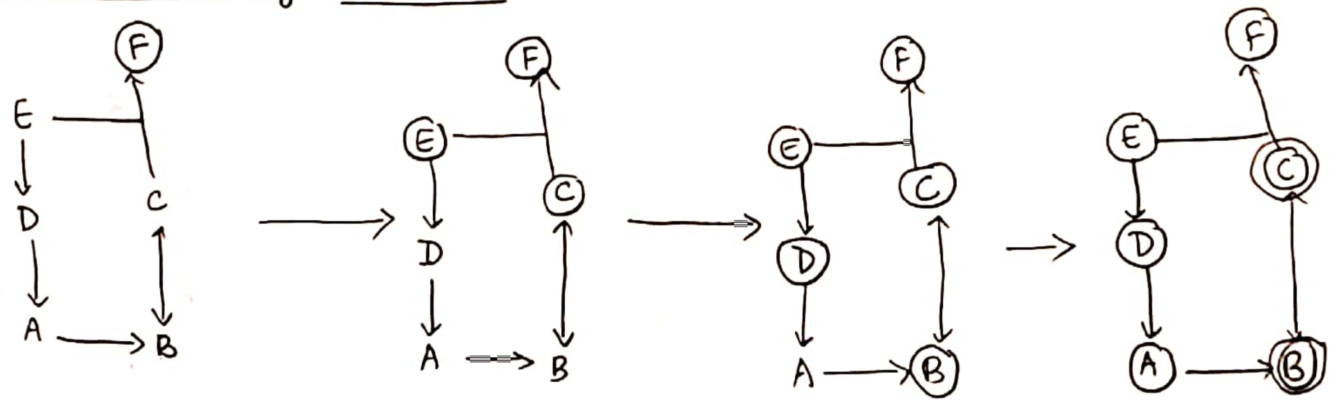
$$\frac{B \quad B \Rightarrow C}{C}$$

$$\frac{C \wedge E \quad (C \wedge E) \Rightarrow F}{F}$$

Here, using modus Ponens, we were able to get to F. Thus, using forward chaining we can say that KB entails F.

proved //

ii) Backward chaining: (1st method)



Here, we begin from our final state 'F'. F is connected to E and C. 'C' is added to stack. Also, C is connected B, so B is added to our goal stack. A is connected B, D is connected to A, and finally E is connected to D. A, D and E are also pushed to our stack and the checking stops at E, since it's true. Our stack currently has [F, C, B, A, D, E]. Now, since E is true state, it is popped out. Since, E is connected to D, D is also popped out. Similary, D is connected to A, A is connected to B, B is connected to C. A, B, and C are popped out respectively and marked true. Now, the only state left is 'F', which is marked true and therefore, using backward chaining we can say that Knowledge base entails 'F'.

2nd Method:

$A \Rightarrow B$

$B \Leftrightarrow C$ i.e. $B \Rightarrow C$ and $C \Rightarrow B$

$D \Rightarrow A$

$E \Rightarrow D$

$(C \land E) \Rightarrow F$

Our goal stack:

E
D
A
B
C
E
F

$\underbrace{E, \quad E \Rightarrow D}_{D}$

$\underbrace{D, D \Rightarrow A,}_{A}$

$\underbrace{A, A \Rightarrow B,}_{B}$

$\underbrace{B, B \Rightarrow C,}_{C}$

$\underbrace{E, C \quad C \land E \Rightarrow F}_{F}$

Therefore, using backward chaining we can say that Knowledge base entails 'F'.

iii) Resolution:

The given KB should be reduced to CNF form:

Equation	Equivalent CNF
$A \Rightarrow B$	$= (\neg A \cup B)$
$B \Leftrightarrow C$	$= (B \Rightarrow C) \cap (C \Rightarrow B)$ $= (\neg B \cup C) \cap (\neg C \cup B)$
$D \Rightarrow A$	$= (\neg D \cup A)$
$E \Rightarrow D$	$= (\neg E \cup D)$
$C \cap E \Rightarrow F$	$= (\neg (C \cap E) \cup F)$ $= (\neg C \cup \neg E \cup F)$
E	$= E$

Entails $= F$

Starts with $= \neg F$

$(\neg F)$ $\quad (\neg C \cup \neg E \cup F)$

$(\neg C \cup \neg E)$ $\quad (E)$

$(\neg C)$ $\quad (\neg B \cup C)$

$(\neg B)$ $\quad (\neg A \cup B)$

$(\neg A)$ $\quad (\neg D \cup A)$

$(\neg D)$ $\quad (\neg E \cup D)$

$(\neg E)$ $\quad (E)$

Here, F is true, since we achieve null at the end. $\quad \phi$

Task-10

a) Constants:

May,
John, Mary

Predicates:

Rains(May) → It rains in May
Gave (John, Mary) → John gave \$1000 cheque to Mary.
Mow (Mary) → Mary mows the lawn.

First order logic statement:

1) $(\exists May)\ Rains(May) \Rightarrow (\exists John\ (\exists Mary)\ Gave(John, Mary))$

2) $(\exists John)(\exists Mary)\ Gave(John, Mary) \Rightarrow (\exists Mary)\ Mows(Mary)$

b) Logical statements:

$\neg Rains(May) \Rightarrow \neg Gave(John, Mary)$
$\neg Gave(John, Mary) \Rightarrow \neg Mows(Mary)$

c) Symbols are:

Rains, Gave, Mows

d) $\neg Rains \Rightarrow \neg Gave$
$\neg Gave \Rightarrow \neg Mows$

e) The contract was not violated because the statement says John must give cheque if it rains but its upto John if doesn't rain. Once John gives cheque to Mary, Mary mows the lawn as per contract.