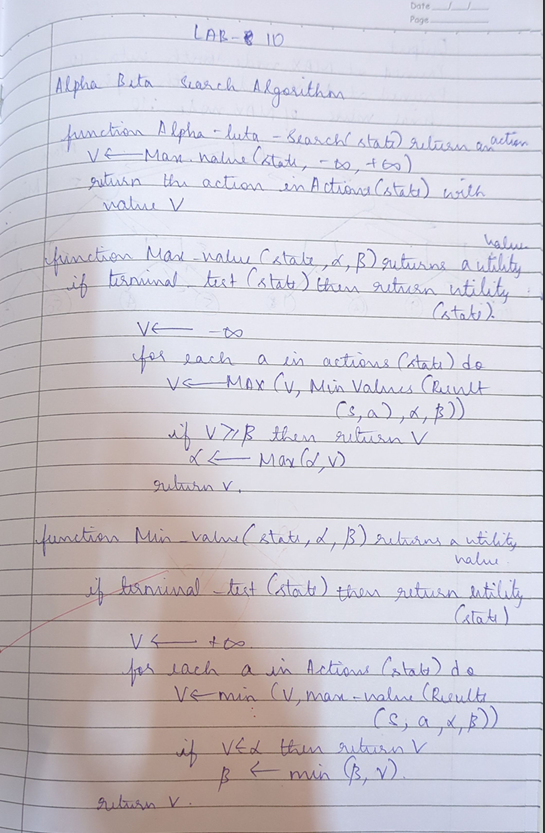
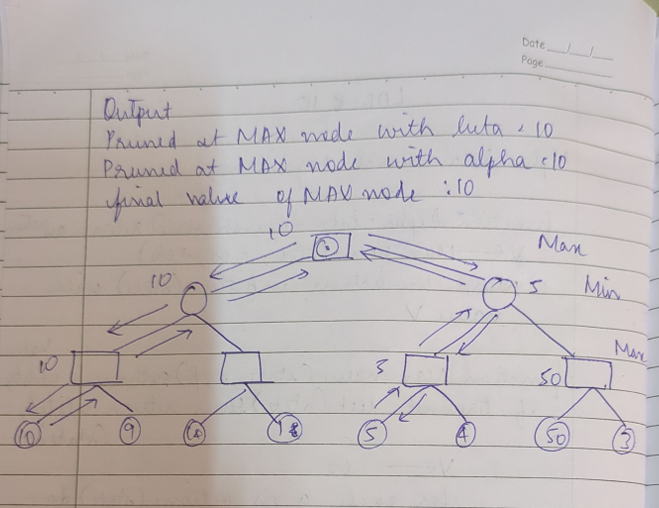
**Alpha-Beta Pruning**

**Algorithm:**





**Code:**

import math

def alpha\_beta\_search(tree, depth, alpha, beta, maximizing\_player):

    if depth == 0 or isinstance(tree, int):

        return tree

    if maximizing\_player:

        max\_eval = -math.inf

        for child in tree:

            eval = alpha\_beta\_search(child, depth - 1, alpha, beta, False)

            max\_eval = max(max\_eval, eval)

            alpha = max(alpha, eval)

            if beta <= alpha:

                print("Pruned at MAX node with beta =", beta)

                break

        return max\_eval

    else:

        min\_eval = math.inf

        for child in tree:

            eval = alpha\_beta\_search(child, depth - 1, alpha, beta, True)

            min\_eval = min(min\_eval, eval)

            beta = min(beta, eval)

            if beta <= alpha:

                print("Pruned at MIN node with alpha =", alpha)

                break

        return min\_eval

game\_tree = [

    [[10, 9], [14, 18]],

    [[5, 4], [50, 3]]

]

final\_value = alpha\_beta\_search(game\_tree, 3, -math.inf, math.inf, True)

print("Final Value of the MAX Node:", final\_value)

**Output:**

