**Name: Muzamil Mushtaq**

**Roll No: SU92-BSAIM-F24-037**

**Section: BSAI-3A**

**Submitted To: Sir Rasikh Ali**

**1. Introduction**

The Minimax Algorithm is a backtracking algorithm used in decision making and game theory.  
It provides the best possible move for a player assuming that the opponent also plays optimally.

It is mainly used in two-player, turn-based games such as:

Tic-Tac-Toe

Chess

Checkers

Connect Four

**2. Concept**

The algorithm simulates all possible moves of both players and builds a game tree.  
Each node in this tree represents a game state, and the leaf nodes represent the final score or outcome of that game.

It alternates between:

Maximizer: tries to get the highest score possible.

Minimizer: tries to minimize the opponent’s score get the lowest value possible.

The algorithm recursively chooses the best move for the maximizing player, assuming that the minimizing player also plays optimally.

**3.How It Works Step-by-Step**

1. Generate all possible moves for the current player.
2. Simulate each move, and recursively call the minimax function for the next player.
3. Assign a value to each terminal leaf state:

Win = positive score

Lose = negative score

Draw = zero

1. Propagate scores upward:

If it’s the maximizing player’s turn → choose the maximum value.

If it’s the minimizing player’s turn → choose the minimum value.

1. The final value at the root node is the best achievable outcome for the starting player.

**4.Advantages Minimax Algorithm**

* **Optimal Decision Making**

Always chooses the best possible move assuming the opponent plays perfectly.

* **Simple and Logical Approach**

Easy to understand and implement using recursion.

* **Applicable to Many Games**

Works for all two-player, turn-based, zero-sum games like Chess, Tic-Tac-Toe, and Checkers.

* **Foundation for AI Techniques**

Forms the base for advanced algorithms such as Alpha-Beta Pruning and Expect minimax.

* **Predictable and Deterministic**

Produces consistent results since there’s no randomness in its decision process.

**6.Disadvantages of Minimax Algorithm**

* **High Time Complexity**

Explores every possible move, leading to exponential growth (O(b^d)).

* **High Memory Usage**

Requires large recursion stacks and space to store game trees.

* **Slow for Complex Games**

Becomes impractical for games with deep or large move trees (like chess).

* **Assumes Perfect Play**

Works best only when both players play optimally — unrealistic in real scenarios.

* **No Learning Ability**

Does not improve or adapt over time; purely rule-based decision making.