DISCRETE MATH PROJECT 2

SHRESHT BHOWMICK

Contents

1 Part A

Invariant Strategy:

Let the number of squares in each row be $n_1, n_2, \ldots, n_k$. Define the *XOR-sum* as:

$$S = n_1 \oplus n_2 \oplus \cdots \oplus n_k$$

where $\oplus$ is the bitwise XOR operation.

Claim: A position is winning if $S \neq 0$. The strategy is to move in such a way that $S = 0$ after our turn.

1.1 Proof:

If $S = 0$, any move results in $S \neq 0$, leaving the opponent in a winning position. If $S \neq 0$, the player can always adjust one row, say n_i, to $n_i \oplus S$, making the new *XOR-sum* zero:

$$S' = n_1 \oplus n_2 \oplus \cdots \oplus (n_i \oplus S) \oplus \cdots \oplus n_k = 0$$

Thus, the optimal strategy is to always move to ensure $S = 0$ after our turn.

2 Part B

- **Input:** The current state of the board, with $n_1, n_2, \ldots, n_k$ squares in each row.
- **Step 1:** Calculate the XOR-sum:

$$S = n_1 \oplus n_2 \oplus \cdots \oplus n_k$$

- **Step 2:** If $S = 0$, the opponent made a mistake; wait for their next move.
- **Step 3:** If $S \neq 0$, find the row r_i such that $n_i \oplus S < n_i$.
- **Step 4:** Set $n_i = n_i \oplus S$.
- **Step 5:** Remove the corresponding number of squares from row r_i.
- **Step 6:** Update the board state and pass the turn to the opponent.

3 Part C

Written in Python file.