Chess is a very popular game played by hundreds of millions of people. Nowadays, we have chess engines such as Stockfish and Komodo to help us analyze games. These engines are very powerful pieces of well-developed software that use intelligent ideas and algorithms to analyze positions and sequences of moves, as well as to find tactical ideas. Consider the following simplified version of chess:

- *Board*:
  - It's played on a 4×4 board between two players named *Black* and *White*.
  - Rows are numbered from 1 to 4, where the top row is 4 and the bottom row is 1.
  - Columns are lettered from A to D, where the leftmost column is A and the rightmost column is D.
- *Pieces and Movement*:
  - *White* initially has w pieces and *Black* initially has b pieces.
  - There are no Kings on the board. Each player initially has exactly 1 Queen, *at most* 2 Pawns, *at most* 2 Rooks, and *at most* 2 minor pieces (i.e., a Bishop and/or Knight).
  - *White*'s Pawns move *up* the board, while *Black*'s Pawns move *down* the board.
  - Each move made by any player counts as a single move.
  - Each piece's possible moves are the same as in classical chess, with the following exceptions:
    - Pawns *cannot* move two squares forward.
    - The en passant move is not possible.
  - *Promotion*:
    - Pawns promote to either a Bishop, Knight, or Rook when they reach the back row (promotion to a Queen is not allowed).
    - The players *must* perform promotions whenever possible. This means *White* must promote their Pawns when they reach any cell in the top row, and *Black* must promote their Pawns when they reach any cell in the bottom row.
- *Objective*:
  - The goal of the game is to capture the opponent’s Queen without losing your own.
  - There will never be a draw or tie scenario like you might see in classical chess.

Given m and the layout of pieces for g games, implement a very basic engine for our simplified version of chess that determines whether or not *White* can win in $\leq m$ moves (regardless of how *Black* plays) if *White* always moves first. For each game, print `YES` on a new line if *White* can win in $\leq m$ moves; otherwise, print `NO`.

Input Format

The first line contains an integer, g, denoting the number of games. The subsequent lines describe each game in the following format:

- The first line contains three space-separated integers describing the respective values of w (the number of white pieces), b (the number of black pieces), and m (the maximum number of moves we want to know if *White* can win in).
- The $w + b$ subsequent lines describe each chess piece in the form `t c r`, where t is a character $\in \{Q, N, B, R, P\}$ denoting the type of piece (where Q is Queen, N is Knight, B is Bishop, R is Rook, and P is a Pawn), and c and r denote the respective column and row on the board where the figure is located (where $c \in \{A, B, C, D\}$ and $r \in \{1, 2, 3, 4\}$). These inputs are given as follows:
  - Each of the first w lines describes the type and location of a *White* piece.
  - Each of the subsequent b lines describes the type and location of a *Black* piece.

Constraints

- $1 \leq g \leq 1000$
- $1 \leq w, b \leq 7$
- $1 \leq m \leq 6$
- Each player has exactly 1 Queen, *at most* 2 Pawns, *at most* 2 Rooks, and *at most* 2 minor pieces (i.e., a Bishop and/or Knight).
- It is guaranteed that the initial location of each chess piece is distinct.
- No pawn is initially placed in a row where it would promote.

Output Format

For each of the g games of simplified chess, print whether or not *White* can win in $\leq m$ moves on a new line. If it's possible, print `YES`; otherwise, print `NO` instead.

Sample Input 0

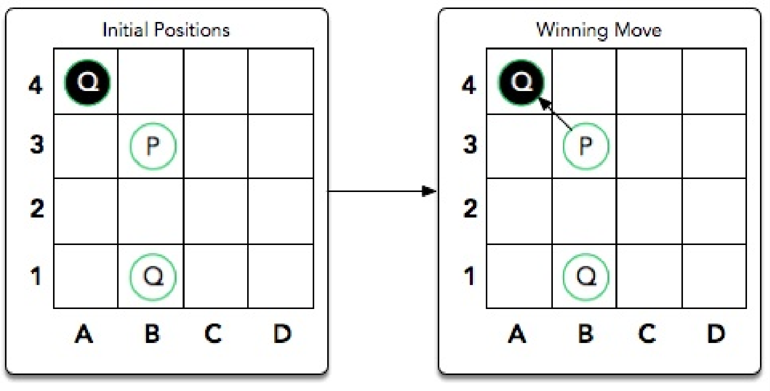
```
1
2 1 1
Q B 1
P B 3
Q A 4
```

Sample Output 0

```
YES
```

Explanation 0

We play the following $g = 1$ game of simplified chess:



White wins by moving their Pawn to $A4$ and capturing *Black*'s Queen, so we print `YES` on a new line.