3439 Bingo!

Bingo is a game in which players try to form patterns on a 5×5 grid (or *card*). Each column on the card is represented by a letter in the game's namesake: B, I, N, G, or O. Each square on the grid contains a number. Players mark numbers as they are chosen randomly until a person has a card with a winning pattern marked (or *bingo*). An exception to this is the center square in the grid, which is a *free spot* and is already marked for all players at the beginning of each game. The possible numbers called are 1..75, inclusive, with each set of fifteen numbers corresponding to a letter: B for 1..15, I for 16..30, N for 31..45, G for 46..60, and O for 61..75.

Given the amount of numbers for each letter already called and information used to determine the set of winning patterns, write a program to determine the fewest amount of numbers that still need to be called for a possible bingo.

Input

Input to this problem will begin with a line containing a single integer n indicating the number of data sets. The first line in each data set will be in the format $B\ I\ N\ G\ O\ X\ Y$ where:

1. B is the amount of numbers in the B category that have already been called;
2. I is the amount of numbers in the I category that have already been called;
3. N is the amount of numbers in the N category that have already been called;
4. G is the amount of numbers in the G category that have already been called;
5. O is the amount of numbers in the O category that have already been called;
6. X (where $1 \leq X \leq 19$) is the number of input patterns (the winning patterns are described through combinations of the input patterns);
7. and Y (where $1 \leq Y \leq \min(5, X)$) is the minimum number of input patterns that must be combined to form a winning pattern.

The next 5 lines in each data set will be a series of 5×5 grids of the input patterns in a format where '`X`' represents a square that must be marked and '`O`' represents a square that does not have to be marked. Using the input patterns and Y given above, the entire set of winning patterns can be determined.

For example, given an X of 4, a Y of 2, and a set of input patterns as follows:

```
XXOOO OOOXX OOOOO OOOOO
XXOOO OOOXX OOOOO OOOOO
OOOOO OOOOO OOOOO OOOOO
OOOOO OOOOO XXOOO OOOXX
OOOOO OOOOO XXOOO OOOXX
```

the set of winning patterns (of which only one must be marked to have a bingo) is:

```
XXOXX XXOOO XXOOO OOOXX OOOXX OOOOO
XXOXX XXOOO XXOOO OOOXX OOOXX OOOOO
OOOOO OOOOO OOOOO OOOOO OOOOO OOOOO
OOOOO XXOOO OOOXX XXOOO OOOXX XXOXX
OOOOO XXOOO OOOXX XXOOO OOOXX XXOXX
```

Output

For each data set, output a single line containing the fewest amount of numbers that still need to be called to form a bingo.

Sample Input

```
3
0 1 0 2 1 4 2
XXOOO OOOXX OOOOO OOOOO
XXOOO OOOXX OOOOO OOOOO
OOOOO OOOOO OOOOO OOOOO
OOOOO OOOOO XXOOO OOOXX
OOOOO OOOOO XXOOO OOOXX
1 1 0 1 1 5 1
XXXXX OOOOO OOOOO OOOOO OOOOO
OOOOO XXXXX OOOOO OOOOO OOOOO
OOOOO OOOOO XXXXX OOOOO OOOOO
OOOOO OOOOO OOOOO XXXXX OOOOO
OOOOO OOOOO OOOOO OOOOO XXXXX
15 15 15 15 4 1 1
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
```

Sample Output

```
4
0
1
```