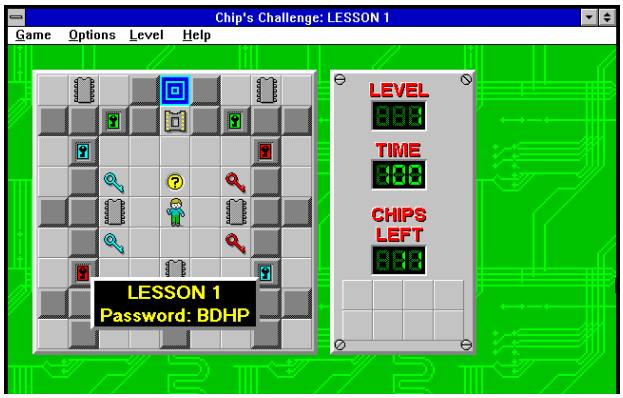
Chips Challenge

Chip's Challenge is a classic top down puzzle game played on a **N x M** grid. Released in 1989, the protagonist of the game is the high-school nerd Chip McCallahan, who has met Melinda The Mental Marvel in the school science laboratory and fell in love at first sight! Chip must navigate through Melinda's clubhouse, consisting of several increasingly difficult puzzles, in order to prove himself and acquire Melinda's heart and also gain access to the very exclusive Bit Busters Club. Such is the premise of this vintage game, cliched, but who cares about the plot when the gameplay is so cool?

The game consists of more than one hundred two-dimensional levels. Gameplay involves using the arrow keys of the numeric keyboard or mouse to move Chip around each of the levels in turn. To progress through each level, Chip needs to interact with various game elements like computer chips, buttons, locked doors, water and lethal monsters. There are also some levels containing block-pushing puzzles and dodging enemies. Below is an example of a level from the actual game.



In this problem, we will attempt to solve a simplified variation of the game Chip's Challenge. You will be given a **N x M** grid with Chip's starting position and also the position of Melinda The Mental Marvel. The grid will consist of empty cells, water, lethal monsters and locked doors only. The rules of this game are a bit different than the original version, so pay close attention to the following details. Chip starts from his initial position and wants to travel to Melinda's location. To do so, he can navigate to any four of the adjacent cells from his current position using the arrow keys up, down, left and right. Provided the cell he wants to move to is empty and inside the grid, he will move to the cell and the whole process takes **exactly 1 second**. Otherwise, Chip is not allowed to make a move.

He can also click on any cell that is not empty and attempt to destroy it. Any cell that is not empty can be destroyed and once destroyed, the cell becomes empty for the rest of the game. Note that the cell does not necessarily have to be adjacent to Chip's current position while destroying it. Initially empty cells are marked using the character '.' (dot) and both Chip's starting position and Melinda's positions are initially empty cells and are different. Every non-empty cell has a power value and destroying a cell with power value **X** takes exactly **X** clicks, hence **X** seconds in total. That is, per click it takes 1 second. Keep in mind that the non empty cells can either consist of water, monsters or locked doors.

Cells containing water are marked using lowercase letters, therefore there can be **26** different

types of water. The power value of a cell containing water is equivalent to its position in the alphabet. That means if a cell contains the character **'a'**, it's power value is **1**, if a cell contains the character **'b'** it's power value is **2**, and similarly for **'z'** it's power value is **26**. Monsters are marked with uppercase letters, and their power values are calculated by adding **26** with its position in the alphabet. So if a cell contains **'A'**, the power value of it will be **27**, for **'B'** it will be **28** and similarly for **'Z'** it will be **52**. Lastly, cells containing locked doors will be represented by the digits **0-9**. The power value of a cell containing a locked door will be the value of the digit plus **53**. If a cell contains a locked door labeled **'0'**, it's value will be **53**, for a cell having locked door labeled '1', the power value will be **54** and for the locked door **'9'**, the power value will be **62**.

Chip can't wait to rejoin the love of his life and join the Bit Busters club with her and he wants to reach her in the shortest time possible. Given the description of the maze, the initial position of Chip McCallahan and Melinda The Mental Marvel, can you suggest how to play Chip's Challenge optimally so that Chip can reach Melinda as fast as possible?

Input

The first line will contain a single integer **T** and then **T** test cases follow. Every test case contains two lines. The first line contains **6** integers - **N, M, Cx, Cy, Mx, My**. The grid dimensions are denoted by **N** and **M**, **(Cx, Cy)** indicates the starting position of Chip and **(Mx, My)** the starting position of Melinda. The maze will be generated pseudorandomly. The second line of every case contains one integer called seed, and the maze is generated by following the C++ code snippet described below. Note that **seed** is a global variable that is initialized once at the start of every test case with the value given, and gets updated after each **update_seed()** call.

```
long long seed;

char maze[2021][2021];

void update_seed(){
    seed = (seed * 1000003 + 10007) % 1000000009;
}

void gen_maze(int N, int M, int Cx, int Cy, int Mx, int My){
    for (int i = 1; i <= N; ++i){
        for (int j = 1; j <= M; ++j){
            if ((i==Cx && j==Cy) || (i==Mx && j==My)){
                maze[i][j] = '.';
                continue;
            }

            update_seed();
            int power = seed % 63;

            if (power == 0)
                maze[i][j] = '.';

            else if (power <= 26)
                maze[i][j] = power - 1 + 'a';

            else if (power <= 52)
```

```
                maze[i][j] = power - 27 + 'A';

            else
                maze[i][j] = power - 53 + '0';
        }
    }
}
```

Constraints

- **1 ≤ T ≤ 10**
- **1 ≤ N, M ≤ 2000**
- **1 ≤ Cx, Mx ≤ N**
- **1 ≤ Cy, My ≤ M**
- **(Cx, Cy) ≠ (Mx, My)**
- **0 ≤ seed ≤ 1000003**

Output

Print the case number followed by the shortest time to complete Chip's Challenge if played optimally. Check the sample for more details.

Sample Input

```
2
3 3 1 1 3 3
3
1 3 1 1 1 3
1
```

Sample Output

```
Case 1: 29
Case 2: 59
```

Explanaton

For the first test case, the maze looks like:

```
.bl
JaS
Gv.
```

The shortest path from (1,1) -> (3,3) is the following:

- Destroy cell (1, 2) in 2 seconds
- Destroy cell (2, 2) in 1 second
- Move right from cell (1, 1) to cell (1, 2) in 1 second
- Move down from cell (1, 2) to cell (2, 2) in 1 second
- Destroy cell (3, 2) in 22 seconds
- Move down from cell (2, 2) to cell (3, 2) in 1 second
- Move right from cell (3, 2) to cell (3, 3) in 1 second

Therefore the overall journey takes **2 + 1 + 1 + 1 + 22 + 1 + 1 = 29** seconds