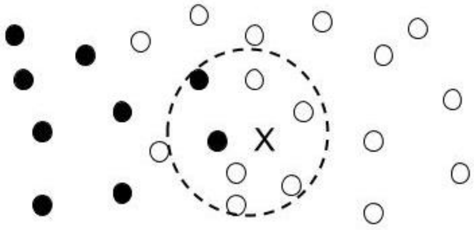
k-NN

Author: ***Shahril***

In *Artificial Intelligence*, there is a subset field called ***machine learning*** which provides the computer the ability to automatically learn and improve from experience without being explicitly programmed.

One of such simple and very popular technique is ***k*-NN (K-Nearest Neighbours)**.

k-NN works by searching the ***k*** nearest points (neighbours) to the desired point. By getting the nearest points, *k*-NN will try to make an assumption based on the nearest points' classes and use that to predict the class of the desired point. In A.I. this is also called classifying problem as we want to predict the class of the desired point.



k = 7

For example, above is set of points with two classes (***A = black circle***, ***B = transparent circle***). ***X*** is a point that we desired to predict. *k* is number of nearest points that we want to find (which in this case is **7**). By searching the distances between X and all points, we then sort out the result. Based on the *nearest seven points* to the X, we found out that there are **2 and in class A and 5 are in the class B.** By getting this information, k-NN will choose the highest frequency among all the classes and uses that determine class of X (which in case **X = class B**).

Input

First line of input has ***$P\ (5 \leq P \leq 20)$,*** which represent number of points in the cartesian plane. For each P lines, there will contain three input X, Y, C where the first two input represent the coordinate of the point and C represent *its class*.
$X\ (0 \leq X \leq 10)$, $Y\ (0 \leq Y \leq 10)$, $C\ (1 \leq C \leq 2)$.

Next line will contain ***$T\ (1 \leq P \leq 5)$***, where it represents the number of test case(s). For each next T lines, there will be three value I, J and K, where I and J represent location of point where we want to predict, and K represent how much nearest nodes we want to compare.
$I\ (0 \leq I \leq 10)$, $J\ (0 \leq J \leq 10)$, $K\ (1 \leq K \leq P)$ and K is odd number.

Output

Print the the case number 'Case #n:'; n is test case number followed by class predicted by your k-NN algorithm. Please note that there will be only two classes, 1 and 2.

Sample Input	**Sample Output**

5
4 2 1
7 2 2
3 6 1
7 4 2
2 3 1
3
3 4 1
5 4 3
4 4 5
	Case #1: 1
	Case #2: 2
	Case #3: 1

Hint:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$