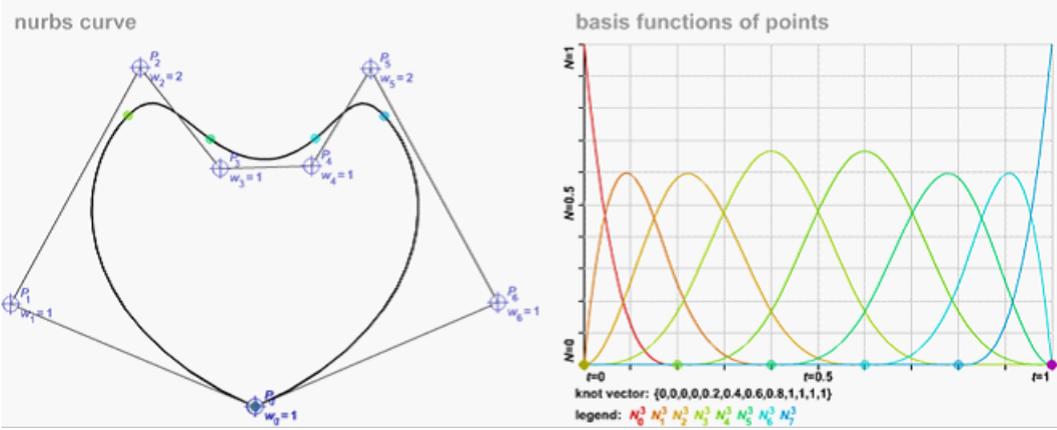


# 12565 Lovely Magical Curves

NURBS Curves are lovely and magical, because you can make a lot of interesting shapes from it:



Given two NURBS curves, your task is the find all their intersection points.

If you're not familiar with NURBS curves, here we go:  
 NURBS is a parametric curve which takes the following form:

$$C(u) = \frac{\sum_{i=1}^n w_i N_{i,k}(u) P_i}{\sum_{i=1}^n w_i N_{i,k}(u)}$$

Where  $u$  is the parameter,  $n$  is the number of control points,  $k$  is the degree of the curve,  $P_i$  and  $w_i$  are the location and weight of the  $i$ -th control point.

The basis function  $N_{i,k}$  is defined recursively below:

$$N_{i,k}(u) = \frac{u - t_i}{t_{i+k} - t_i} N_{i,k-1}(u) + \frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+1}} N_{i+1,k-1}(u)$$

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } t_i \leq u < t_{i+1} \\ 0 & \text{else} \end{cases}$$

Where  $t_i$  is the  $i$ -th knot value. **In the formula above, 0/0 is deemed to zero.**

To understand the formulae above, here are some brief explanations of the parameters:

**Degree.** The *degree* is a positive integer. NURBS lines and polylines are usually degree 1 (linear curve), NURBS circles are degree 2 (quadratic curve), and most free-form curves are degree 3 or 5.

**Control Points.** The control points are a list of at least *degree* + 1 points. One of easiest ways to change the shape of a NURBS curve is to move its control points (You can try it out: <http://geometrie.foretnik.net/files/NURBS-en.swf>).

Each control point has an associated number called weight. In this problem, weights are positive numbers. If you increase the weight of a control point, the curve is pulled toward that control point and away from other control points.

**Knots.** The knot vector is defined as  $U = [t_1, t_2, \dots, t_m]$ . The relation between the number of knots  $m$ , the degree  $k$ , and the number of control points  $n$  is given by  $m = n + k + 1$  (In OpenNURBS/Rhinoceros website,  $m = n + k - 1$ . The algorithm presented here is referred as “some older algorithms”. When solving this problem, please stick to this problem description).

The sequence of knots in the knot vector  $U$  is assumed to be non-decreasing, i.e.  $t_i \leq t_{i+1}$ . Each successive pair of knots represents an interval  $[t_i, t_{i+1})$  for the parameter values to calculate a segment of a shape. **Thus, the whole NURBS curve is defined within  $[t_1, t_m)$ .** The number of times a knot value is duplicated is called the knot’s *multiplicity*, which should be no more than the *degree*. Duplicate knot values in the middle of the knot list make a NURBS curve less smooth.

If you’re still puzzled after reading all the information above, suppose we’re moving  $u$  from  $t_1$  towards  $t_m$  (but never reach  $t_m$ ), then the point  $C(u)$  will move long the NURBS curve we define.

## Input

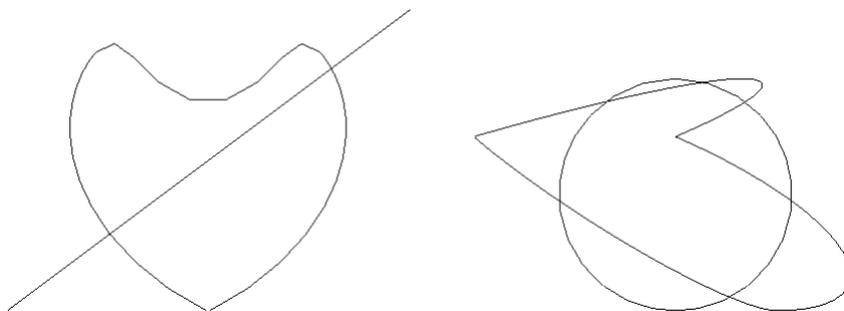
The first line contains the number of test cases  $T$  ( $T \leq 25$ ). Each test case contains two parts, one for each NURBS curve. Each curve begins with two integers  $n$  and  $m$  ( $2 \leq n \leq 20$ ), the number of control points and the number of knots. Each of the next  $n$  lines contains three real numbers  $x, y, w$  ( $0 \leq x, y \leq 10, 0 < w \leq 10$ ), describing a control point  $(x, y)$  with weight  $w$ . The next line contains  $m$  real numbers, describing the knot vector. The first knot value is always 0 and the last one is always 1. The degree of both NURBS curves will be 1, 2, 3 or 5.

## Output

For each test case, print the number of intersection points in the first line, then each point is printed in a following line. The coordinates should be rounded to three decimal places, and points should be sorted lexicographically (i.e. points with smaller  $x$ -coordinate comes earlier). Inputs are carefully designed so that the minimal difference of  $x$ -coordinate between any two intersection points will be at least 0.005 (otherwise the sorting result might be affected by numerical stability).

Print a blank line after each test case.

**Note:** The pictures of the samples are shown below:



## Sample Input

```
2
8 12
2 0 1
0 1 1
1 3 2
1.5 2 1
2.5 2 1
3 3 2
```

```
4 1 1
2 0 1
0 0 0 0 0.2 0.4 0.6 0.8 1 1 1 1
2 4
0 0 1
4 3 1
0 0 1 1
7 10
1 1.732 1
0 0 0.5
2 0 1
4 0 0.5
3 1.732 1
2 3.464 0.5
1 1.732 1
0 0 0 0.333 0.333 0.667 0.667 1 1 1
7 10
0 1.732 1
2 0 0.5
3 0 1
6 0 0.5
2 1.732 1
6 3.464 0.5
0 1.732 1
0 0 0 0.333 0.333 0.667 0.667 1 1 1
```

### Sample Output

Case 1: 2

(1.029, 0.772)

(3.221, 2.416)

Case 2: 6

(0.847, 1.092)

(1.307, 2.078)

(2.283, 2.274)

(2.538, 0.133)

(2.693, 2.078)

(3.153, 1.092)