5875   Orienteering

Orienteering involves running through unfamiliar territory, using a map and a compass to navigate to various control points marked on the map. In the most common form of the sport, the order in which the control sites must be visited is set in advance by the race organizers, and the winner of a race is the runner who visits all the controls in the prescribed order and arrives at the finish line in the shortest amount of time. Thus the goal is to visit all control points (in order) as fast as possible.

An obscure variant of the sport is *Half-Score Orienteering*, in which the organizers define the order in which the control points must be visited, choose a time limit and assign a score to each control point, generally correlated to its difficulty and distance from the starting point of the race. Each runner begins this kind of race by estimating how far he can run in the time available and then chooses a subset of the control points. The runner's aim is for a high-scoring subsequence that forms a route following the general outline of the organizers' course (and ordering) but skips undesired control points. The total length of the route is close to, but not exceeding, his distance estimate.

The start and finish points for each race are at the origin $(0, 0)$. Assume that the runners navigate perfectly along straight-line paths between control points (and to/from the start and finish). Therefore a viable route for a runner starts at the origin, passes through a subsequence of the control points, in the order of appearance in the input, and then returns to the origin. The total length must not exceed his distance estimate.

After a race, the runners are always curious as to whether they could have done better by aiming for a different subsequence of the available control points. Your task is to write a program to determine the maximum score available for each runner in a race.

Input

Input contains a number of races. The description of each race consists of:

- The first line contains an integer, n, that represents the number of control points in the race $(1 \leq n \leq 30)$.

- Each of the following n lines contains three integers, separated by spaces, that represents data for one control point. The three integers are x, y, and s, where (x, y) are the coordinates of the control point in meters $(-5000 \leq x, y \leq 5000)$ and s is the control point's score $(10 \leq s \leq 200)$.

- A number of lines, with one line for each runner in the race, follows. '# 0' on a line by itself terminates the runners data. Each of these lines contains the runner's name and an integer d, separated by a single space. The name is a single word of up to 60 characters with no blank spaces, and d is the distance in meters that the runner can travel in the time available $(0 \leq d \leq 10000)$.

- The last line of input contains a '0'. This line should not be processed.

Output

Output for each race starts with a line 'Race r' stating the race number, starting with 1. This will be followed by one line for each of the runners in the race, in the order in which they appear in the race input, containing the runner's name and the score of the highest scoring viable route available to that runner, formatted as '*name*: *score*'.

Sample Input

```
5
750 -800 30
1500 0 50
750 750 60
-1250 750 70
-1000 -500 50
Chris 7000
Karl 6500
Tania 5000
# 0
4
500 0 10
0 500 10
-500 0 10
0 -500 10
Hanny 2100
Lizzie 1800
# 0
0
```

Sample Output

```
Race 1
Chris: 230
Karl: 180
Tania: 140
Race 2
Hanny: 20
Lizzie: 20
```