Schedule

Run-time Limit: 1 second
Memory Limit: 32 MB

DESCRIPTION

Petruk lives on year 3000 in Namex planet. He is a very busy person. He always take a note of every schedule he has on his notebook. If he has a new schedule, Petruk will check his notes for his schedule. One schedule set item has a pair of hour, minute, and second, which states the starting event time and ending event time. Today, Petruk has already had three schedule written on his notebook. And then, Petruk has a new important meeting. Help Petruk to determine if the new schedule is collided with the schedule that already in the notebook.

Because Petruk lives in Namex planet, there is a time difference between Namex and Earth. A year on Namex is equal to 20 month on Earth. And every month on Namex is equal to 50 days on Earth.

INPUT FORMAT

The first line of input contains integer N ($1 \le N \le 100$) ,denoting number of cases. Each case contains 4 lines. Every line consists of pair of date time range D_1, Mo_1, Y_1, H_1, Mi_1, S_1, D_2, Mo_2, Y_2, H_2, Mi_2, and S_2. It's guaranteed that the second time is greater than the first time and the schedules that is written on the notebook don't collide with each other.

The 1st, 2nd, and 3rd pair of date time range considered to be the schedules that is written in Petruk's notebook. The 4th set of date time range considered to be the new schedule that should be checked.

CONSTRAINTS

- $1 \le D_1, D_2 \le 50$
- $1 \le Mo_1, Mo_2 \le 20$
- $3000 \le Y_1, Y_2 \le 3050$
- $0 \le H_1, H_2 \le 23$
- $0 \le Mi_1, Mi_2 \le 59$
- $0 \le S_1,S_2 \le 59$

OUTPUT FORMAT

For each case, output “Case #X: Y” where X is the case number starts from 1, and Y is “COLLIDE” (without quotes) if the new schedule is colliding with any of the three schedule written in the notebook, otherwise, Y will be “OK” (without quotes).

1st INPUT EXAMPLE

```
1
2 20 3001 15 50 0 3 20 3001 16 0 0
4 20 3001 16 50 0 5 20 3001 17 0 0
1 20 3001 17 30 0 1 20 3001 18 0 0
2 20 3001 18 0 0 2 20 3001 21 0 0
```

1st OUTPUT EXAMPLE

```
Case #1: COLLIDE
```

2nd INPUT EXAMPLE

```
2
30 15 3002 12 0 0 40 20 3003 16 0 0
35 17 3005 11 0 0 40 20 3006 13 0 0
25 15 3001 12 0 0 15 14 3002 16 0 0
25 15 3001 1 0 0 25 15 3001 11 0 0
30 15 3013 12 0 0 40 20 3023 15 0 0
35 17 3025 11 0 0 45 20 3030 17 0 0
25 15 3031 12 0 0 15 14 3035 15 0 0
25 12 3035 1 0 0 25 15 3040 11 0 0
```

2nd OUTPUT EXAMPLE

```
Case #1: OK
Case #2: COLLIDE
```