Code Jam 2017 - Round 3

Mountain Tour

Problem

You are on top of Mount Everest, and you want to enjoy all the nice hiking trails that are up there. However, you know from past experience that climbing around on Mount Everest alone is bad — you might get lost in the dark! So you want to go on hikes at pre-arranged times with tour guides.

There are **C** camps on the mountain (numbered 1 through **C**), and there are 2 × **C** one-way hiking tours (numbered 1 through 2 × **C**). Each hiking tour starts at one camp and finishes at a different camp, and passes through no other camps in between. Mount Everest is sparsely populated, and business is slow; there are exactly 2 hiking tours departing from each camp, and exactly 2 hiking tours arriving at each camp.

Each hiking tour runs daily. Tours 1 and 2 start at camp 1, tours 3 and 4 start at camp 2, and so on: in general, tour 2 × i - 1 and tour 2 × i start at camp i. The i-th hiking tour ends at camp number $\mathbf{E_i}$, leaves at hour $\mathbf{L_i}$, and has a duration of exactly $\mathbf{D_i}$ hours.

It is currently hour 0; the hours in a day are numbered 0 through 23. You are at camp number 1, and you want to do each of the hiking tours *exactly once* and end up back at camp number 1. You cannot travel between camps except via hiking tours. While you are in a camp, you may wait for any number of hours (including zero) before going on a hiking tour, but you can only start a hiking tour at the instant that it departs.

After looking at the tour schedules, you have determined that it is definitely possible to achieve your goal, but you want to do it as fast as possible. If you plan your route optimally, how many hours will it take you to finish all of the tours?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each begins with one line with an integer **C**: the number of camps. Then, 2 × **C** more lines follow. The i-th of these lines (counting starting from 1) represents one hiking tour starting at camp number floor((i + 1) / 2), and contains three integers $\mathbf{E_i}$, $\mathbf{L_i}$, and $\mathbf{D_i}$, as described above. Note that this format guarantees that exactly two tours start at each camp.

Output

For each test case, output one line containing `Case #x: y`, where `x` is the test case number (starting from 1) and `y` is the minimum number of hours that it will take you to achieve your goal, as described above.

Limits

Time limit: 20 seconds per test set.
Memory limit: 1 GB.
1 ≤ **T** ≤ 100.
1 ≤ $\mathbf{E_i}$ ≤ **C**.
$\mathbf{E_i}$ ≠ ceiling(i / 2), for all i. (No hiking tour starts and ends at the same camp.)
size of {j : $\mathbf{E_j}$ = i} = 2, for all j. (Exactly two tours end at each camp.)

$0 \leq \mathbf{L_i} \leq 23$.
$1 \leq \mathbf{D_i} \leq 1000$.
There is at least one route that starts and ends at camp 1 and includes each hiking tour exactly once.

Small dataset (Test Set 1 - Visible)

$2 \leq \mathbf{C} \leq 15$.

Large dataset (Test Set 2 - Hidden)

$2 \leq \mathbf{C} \leq 1000$.

Sample

Sample Input

```
2
2
2 1 5
2 0 3
1 4 4
1 6 3
4
3 0 24
2 0 24
4 0 24
4 0 24
2 0 24
1 0 24
3 0 24
1 0 24
```

Sample Output

```
Case #1: 32
Case #2: 192
```

In sample case #1, the optimal plan is as follows:

- Wait at camp 1 for an hour, until it becomes hour 1.
- Leave camp 1 at hour 1 to take the 5 hour hiking tour; arrive at camp 2 at hour 6.
- Immediately leave camp 2 at hour 6 to take the 3 hour hiking tour; arrive at camp 1 at hour 9.
- Wait at camp 1 for 15 hours, until it becomes hour 0 of the next day.
- Leave camp 1 at hour 0 to take the 3 hour hiking tour; arrive at camp 2 at hour 3.
- Wait at camp 2 for 1 hour, until it becomes hour 4.
- Leave camp 2 at hour 4 to take the 4 hour hiking tour; arrive at camp 1 at hour 8.

This achieves the goal in 1 day and 8 hours, or 32 hours. Any other plan takes longer.

In sample case #2, all of the tours leave at the same time and are the same duration. After finishing any tour, you can immediately take another tour. If we number the tours from 1 to 8 in the order in which they appear in the test case, one optimal plan is: 1, 5, 4, 7, 6, 2, 3, 8.