Czech ACM Student Chapter
Charles University in Prague
Slovak University of Technology
University of Žilina

Czech Technical University in Prague
Technical University of Ostrava
Masaryk University
Pavol Jozef Šafárik University in Košice

CTU Open Contest 2010

Bus Clock Display

`clock.c`, `clock.C`, `clock.java`, `clock.p`

When you travel to Egypt, you will probably want to enjoy the famous pyramids and possibly also to take a bus to Luxor to visit the Valley of the Kings. To overcome boredom during such a long trip, imagine you are trying to keep yourself occupied by observing a large clock above the driver's seat. The clock is a digital one and uses the classical 7-segment display. Since the Egyptian buses are crowded and most local people use to carry strange things with them, your view may be obstructed and some segments possibly hidden.

If you think about such situations, you will quickly notice that it is sometimes possible to read the time reliably even if some of the segments are not visible. For example, in Figure 1, the clock cannot show anything else than 12:04.

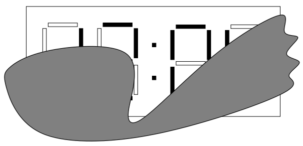
Figure 1

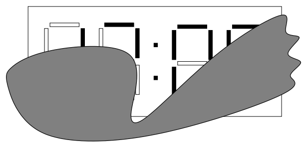
Figure 2

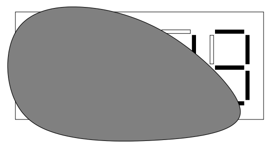
Figure 3

In some other cases, there may be several possibilities and you cannot tell for sure what time is displayed. Figure 2 is a good example – the last digit may be 0, 5, 6, 8, or 9. But even if you cannot tell the time immediately, you do not lose hope. After a while, you take another look at the clock and see another "sample" of it. Meanwhile, the people moved a little bit, so you can see a different portion of the display now, as shown in Figure 3. The new picture is even more ambiguous if seen alone. But you are quite sure that no more than four minutes passed between the two samples. With this knowledge, you may not only say what time it is now, but also retroactively determine what time it was before.

Your task is to write a program that automates this process. Given samples of the display (with some segments potentially hidden) and the estimate of the time elapsed between every two consecutive samples, you have to compute what time was displayed at the clock when the samples were taken.

Input Specification

The input contains several bus descriptions. Each description starts with a positive integer number S, which denotes the number of samples taken, $1 \leq S \leq 100$. Then the individual samples follow. Each sample is represented by 28 characters that give states of all segments, in the order shown in Figure 4. The characters are separated by at least one space or new-line. Additional spaces may be used for better presentation in printed materials.

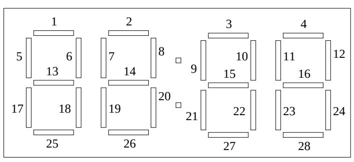

Figure 4

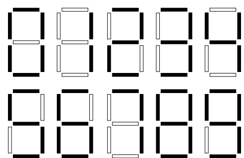
Figure 5

Each character is either a dash “-” when you see that a horizontal segment is lit, a pipe character “|” for a vertical segment lit, a dot “.” for a horizontal or vertical segment that you know is not lit, or a question mark “?” if the corresponding segment cannot be seen at all.

Shapes of individual digits used by the display are shown in Figure 5. The clock uses the 24-hour format with values between 0:00 and 23:59. The first of four digit positions never displays zero, it will be blank if the number of hours is less than 10.

Between each two consecutive samples, there are two integer numbers separated by a space, T_{min} and T_{max}, $0 \leq T_{min} \leq T_{max} \leq 120$, denoting the minimal and maximal number of minutes that may have elapsed between those two samples. The last bus description is followed by a line containing zero.

Output Specification

For each bus description, print one line for every sample taken in that bus. If the time shown by the clock can be determined uniquely, print the time as three or four digits with a colon between hours and minutes. If the time is ambiguous, print the sentence “`ambiguous,` P_i `possibilities`”, where P_i is the total number of different times that might have been displayed at that moment. Remember to always consider all those (and only those) possibilities that are consistent with *all* samples. Print one empty line after each bus.

Sample Input

```
3
 -    -      .    -
. |  . |    | |  | |
 -    -      -    .
| .  . |    . |  | |
 -    -      .    -
16 30
 ?    ?      ?    -
? ?  ? ?    ? ?  | .
 ?    ?      ?    -
? ?  ? ?    ? ?  . |
 ?    ?      ?    -
15 30
 ?    ?      ?    ?
? ?  ? ?    ? ?  ? |
 ?    ?      ?    ?
? ?  ? ?    ? ?  ? ?
 ?    ?      ?    ?
1
 .    -      -    .
. |  . |    | |  | ?
 ?    ?      .    ?
? ?  ? .    | ?  ? ?
 ?    ?      ?    ?
2
 .    -      -    -
. |  . |    | |  | ?
 ?    ?      .    ?
? ?  ? .    | ?  ? ?
 ?    ?      ?    ?
0 4
 ?    ?      .    -
? ?  ? ?    ? |  . |
 ?    ?      ?    -
? ?  ? ?    ? ?  ? |
 ?    ?      ?    -
0
```

Output for Sample Input

```
23:40
0:05
ambiguous, 13 possibilities

12:04

12:09
12:13
```