# B • The Bank of Kalii

Customers of the *Bank of Kalii* handle their banking transactions similar to the way they handle their taxes: be as terse as possible. As a result, when a customer writes a check or fills out a deposit or withdrawal form, they leave off the year on any date they write down. So, instead of writing: 09/20/2005, they would write: 9/20 and be done with it. In general, the year can be inferred since it will be relatively close to the date the transaction is actually processed by the bank.

Without going into the intricate details of how the *Bank of Kalii* calculates interest and banking fees (that is a problem for another time...), suffice to say the bank must determine the actual date the customer wrote on the check or form, and calculate the number of days prior (or in the future) the document is dated. You see, *Kaliian* bankers, like their government officials, are overworked, so they may not get around to processing transactions for up to a week. The customers know this, so they often date their checks and forms a several days in the future –this complicates the bankers' duties as well.

Your job is to write a program to compare a date written on a check or form with the date the transaction is being processed, and, determine the full date the customer meant as well as how many days prior (or in the future) the document is dated.

## Input

The first line of input contains an integer $N$ which is the number of datasets that follow ($1 \le N \le 1000$). Each dataset consists of a single line containing two dates: the *transaction* date and the *document* date; there is a single space between them. The *transaction* date is of the form **M/D/Y** where **M** is the month number ($1 \le M \le 12$), **D** is the day of month ($1 \le D \le md_1$) and **Y** is the year ($2000 \le Y \le 2200$). The *document* date is of the form **m/d** where **m** is the month number ($1 \le m \le 12$) and **d** is the day of month ($1 \le d \le md_2$). The values of $md_1$ and $md_2$ will not exceed the number of days in the respective months **M** and **m**.

## Output

For each dataset print out the dataset number followed by a space followed by the result of the date comparison as shown in the table below:

| Result to print | Criteria |
|---|---|
| **m/d/y IS *n* DAY(S) PRIOR** | If the document date occurs before the transaction date and is within 7 days in the past. |
| **m/d/y IS *n* DAY(S) AFTER** | If the document date occurs after the transaction date and is within 7 days in the future. |
| **SAME DAY** | If the dates are the same |
| **OUT OF RANGE** | All other results not with +/- 7 days. |

Notes: When printing the result date, **`m/d/y,`** you will have to determine the year value **`y`** (`1999 <= y <= 2201`). This is not necessarily the same as the transaction date's year value **`Y`**. Since the *Kalii* taxation fiasco a couple of years back, the *Kaliian* government decided to switch to the standard Gregorian calendar. As such, Gregorian leap year rules apply. A year is a leap year (February has 29 days instead of 28) if the year if evenly divisible by 4, except for century years (those ending in 00), which are leap years only if they are evenly divisible by 400. 2000 and 2004 are leap years, but 2100 and 2101 are not. For those who do not know, the months of January, March, May, July, August, October and December all have 31 days in them. February has 28 days (unless in a leap year, then it has 29). The remainder of the months has 30 days.

| Sample Input | Sample Output |
|---|---|
| 7<br>11/20/2005 11/21<br>11/20/2005 11/17<br>11/20/2005 11/20<br>11/20/2005 11/13<br>11/20/2005 11/28<br>1/2/2005 12/30<br>12/31/2100 1/3 | 1 11/21/2005 IS 1 DAY AFTER<br>2 11/17/2005 IS 3 DAYS PRIOR<br>3 SAME DAY<br>4 11/13/2005 IS 7 DAYS PRIOR<br>5 OUT OF RANGE<br>6 12/30/2004 IS 3 DAYS PRIOR<br>7 1/3/2101 IS 3 DAYS AFTER |