Bus Schedules

`bus.c`, `bus.C`, `bus.java`, `bus.p`

Imagine that you happen to be the one to advance to the World Finals. Sounds good, doesn't it? Then you would be going to travel by various means of transport: airplanes, trains, buses, etc. Are you ready for that? This problem tries to evaluate your orientation skills in bus schedules ("time-tables").

As you know, not all buses operate on every-day basis. In bus schedules, the days when the bus operates are typically given by a list of various specifiers, some of them being simple ("only on Mondays"), some a little bit more confusing ("only on workdays"), and some totally impenetrable ("every day directly preceding Sunday or holiday, unless it is itself a holiday or a day following a holiday inside a leap year").

For the purposes of this problem, the specifier may be one the following:

"`1`"	The bus operates on Mondays.
"`2`" ...	The bus operates on Tuesdays, etc.
"`7`"	The bus operates on Sundays.
"`t`"	The bus operates on Sundays and official holidays, that is January 1, Easter Monday, May 1, May 8, July 5, July 6, September 28, October 28, November 17, and December 24, 25, and 26.
"`w`"	The bus operates on workdays, that is Monday to Friday except for the days specified in `t`.
"`a`"	The bus operates on workdays immediately following the days specified in `t`.

Furthermore, the days may be restricted only to specific dates or a range of dates, given as a comma-separated list of elements. Each element may be either a single date (written in the form of "D`.`M`.`", where D is the day number and M is the month number) or a date range (written as "D_1`.`M_1`.-`D_2`.`M_2`.`", where the first of the dates must precede the second one, and both of these boundary dates are *included* in the range). For example

```
3a 1.1.-30.6.,31.7.,1.9.-31.12.
```

describes a bus that goes only on Wednesdays and on workdays immediately following holidays, but only if one of these occurs on July 31 or between January 1 and July 30, or between September 1 and December 31 (inclusive).

Your task is to count all days that the bus operates in a given year.

Remarks:

- Today is Saturday, October 23, 2010.
- Each year has 365 days, except for leap years, in which an additional day (February 29) is inserted. A year Y is a leap year, if the number Y is evenly divisible by 4. The exception are years divisible by 100, which are leap years only if they are also divisible by 400.

  In all non-leap years, if 29.2. is used as a single date, it is ignored; if it is used as a date-range start, it is interpreted as March 1, and if it is used as a date-range end, it is interpreted as February 28.

- Easter Monday is the Monday immediately following the Easter Sunday. Easter Sunday is the first Sunday *strictly after* the Paschal Full Moon date of the year. The Paschal Full Moon date for year y can be determined by the following algorithm:

  ```
  int golden, solar, lunar, p;

  golden = (y % 19) + 1;
  solar = (y - 1600) / 100 - (y - 1600) / 400 ;
  lunar = (((y - 1400) / 100) * 8) / 25;
  p = (3003 - (11 * golden) + solar - lunar) % 30;
  if (p == 29 || (p == 28 && golden > 11)) p--;
  ```

  Paschal Full Moon date is then p days ($0 \leq p \leq 28$) after March 21.

Input Specification

The input contains several test cases, each case consists of a single line. The line contains two non-empty strings S, R, and an integer Y ($1600 \leq Y \leq 3000$) separated by a space. The string S will contain only characters "`1`", …, "`7`", "`t`", "`w`", "`a`", each appearing at most once (in any order). The string R of length at most 1000 is a comma-separated list of dates and date ranges whose format was specified above. The dates and date ranges in R do not overlap and are always given in an ascending order.

Output Specification

For each input instance, output a single line containing one integer, the number of days in the year Y that belong to the range R and satisfying at least one condition specifier given by S.

Sample Input

```
3a 1.1.-30.6.,31.7.,1.9.-31.12. 2010
```

Output for Sample Input

```
89
```