Section 1: Background

Under the Julian Calendar, named for Julius Caesar, a year is a leap year if it is divisible by four. This rule is simple, but it results in too many leap days. When the Julian Calendar was initially adopted, the solstices and equinoxes occurred on the 25th of the month (March, June, September, and December). By 325, the calendar had advanced relative to the Earth's orbit, and the solstices/equinoxes fell on the 21st of the month. By the late 1500s, the calendar had advanced even farther, and the equinoxes/solstices were falling on the 11th. It was determined that the calendar needed to be fixed.

In 1582, Pope Gregory XIII approved a new calendar designed primarily by Father Christopher Clavius, S.J., who incorporated a formula suggested by Vatican librarian Aloysius Giglio. Giglio's formula, still in use today, has a year being a leap year if it is divisible by four, except when it is divisible by 100 and not by 400. So 1900 was not a leap year, 2100 will not be a leap year, but 2000 was a leap year and 2400 will be a leap year. This means that in the Gregorian calendar there are 97 leap days every 400 years, instead of the Julian calendar's 100 leap days every 400 years. This new formula corrected the calendar's drift, but did nothing to undo the shift which had already happened. To address that problem, October 1582 had 10 days chopped out of it: Thursday, October 4 was followed immediately by Friday, October 15. This put the equinoxes and solstices back on the 21st, where they will probably remain for the forseeable future.

However, the sixteenth century was a time of political upheaval, and so the new calendar was not universally adopted. England and her colonies adopted the calendar in 1752, by which point the calendar had wandered even farther from solar events. So in September 1752, the calendar in British possessions had 11 days removed, giving this:

```
   September 1752
 S  M Tu  W Th  F  S
       1  2 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

People who lived during this time had important dates (such as birthdays) rearranged as a result. George Washington was born in 1732, and his birthday was 11 February 'Old Style', which we now refer to as '22 February'.

It has been proposed that an additional refinement be made to Giglio's formula, so that every year divisible by 4000 would not be a leap year. As it is difficult to predict precisely what political conditions will pertain over the next 1999 years, we do not know whether this refinement will be formally in place when it becomes important. As a result, we will not consider dates after Monday, 28 February 4000. Additionally, to keep the problem domain small enough to manage, we will not consider any date prior to the adoption of the Gregorian Calendar, on 15 October 1582.

Within those restrictions, we are interested in two questions about a given date:

1. how many days ago was (or days from now is) it?
2. what day of the week was (or is) it?

Section 2: Problem Specification

Your program must accept three integers as input, which will represent a day, month, and year. (For C++ students, accept the three numbers on a line; VB students can use three text boxes.) For each date, you must display one of the following messages as a result:

1. The date, using the name (instead of a number) for the month, the day of the week (as a word), and the number of days either in the past or future.
2. The numbers entered, a warning, and a statement of why the date is not valid. If the date is invalid for more than one reason, you must print them all (you may print them in any order).

You may remember (or you may not 8-)) that April, June, September, and November have 30 days, February has 28 or 29 days depending on whether it is a leap year, and the other months have 31 days. Do not allow negative numbers or zero for any value. In the event a month number is invalid (as in '18' or '-6'), use 31 as a default upper bound for 'valid' day numbers (since no month in the current calendar has more than 31 days).

Note: You should reject a date of '33 5 1999' with a specific error, advising the user that May 1999 had only 31 days. But you can't give a specific error for the 33 in '33 22 1992', because '22' doesn't refer to any actual month. Therefore, in addition to saying that 22 is an invalid month, you should say that 33 is an invalid day.

Assume that for valid months, they will never have more days than the maximum they can have in the current system. So 29 February 4000 should give only one error, that it is out of range. But 30 February 4000 should give two errors: it is out of range, and February cannot have more than 29 days.

Your program should exit if a zero is entered for all three values. In all other cases, it should be able to accept further input and continue processing.

As New Jersey was a British colony, we will follow British usage. While our start date is the adoption of the Gregorian Calendar on 15 October 1582, Britain was still on the Julian Calendar until 1752. From 15 October 1582 through 2 September 1752, use the Julian leap year formula (in which every year divisible by four is a leap year). Starting with 14 September 1752, and continuing through 28 February 4000, use the Gregorian leap year formula (which follows the Julian formula with the modification that years divisible by 100 are not leap years unless they are also divisible by 400). Remember that September 1752 had only 19 days, and that 2 September 1752 was followed immediately by 14 September 1752.

Notes on Formatting and Output

Your output **does not** have to conform to the formatting which appears in Section 3, so long as the output is clear, understandable, and correct. You should submit a screenshot showing your program's output for the test data which appears in Section 4. Your program must allow you to run multiple dates without restarting, and have the results of multiple dates displayed on the screen at once.

You should submit sufficient screenshots to include all of the test data. If your output format requires more than one screenshot to display all the results, you may submit more than one screenshot along with your code.

Your program will also be tested using additional test data known only to the judges.

Section 3: Sample Data

```
Input:      Output:

1 4 2001    1 April 2001: Sunday, 1 day from now

30 1 1982   30 January 1982: Saturday, 7000 days ago

13 9 1752   13 9 1752 is invalid: September 1752 was truncated

14 9 1752   14 September 1752: Thursday, 90778 days ago

2 9 1752    2 September 1752: Wednesday, 90779 days ago

1 1 1581    1 1 1581 is invalid: no dates before 15 October 1582

30 23 2345  30 23 2345 is invalid: 23 is an invalid month

34 56 7890  34 56 7890 is invalid: no dates after 28 February 4000
                                   56 is an invalid month
                                   34 is an invalid day

29 2 4000   29 2 4000 is invalid: no dates after 28 February 4000

31 3 2101   31 March 2101: Thursday, 36524 days from now

32 15 3001  32 15 3001 is invalid: 15 is an invalid month
                                   32 is an invalid day

30 2 2000   30 2 2000 is invalid: no days > 29 in February 2000

29 2 2000   29 February 2000: Tuesday, 396 days ago

15 15 2001  15 15 2001 is invalid: 15 is an invalid month

-3 13 322   -3 13 322 is invalid: no dates before 15 October 1582
                                  -3 is an invalid day
                                  13 is an invalid month

0 4 1999    0 4 1999 is invalid: 0 is an invalid day

3 -3 2000   3 -3 2000 is invalid: -3 is an invalid month

31 3 2001   31 March 2001: Saturday, 0 days ago
```

(**Note:** The last one could also say ‘0 days from now’.)

Section 4: Test Data

Run your program on these dates, and print out a screenshot showing that your output is correct.

```
29 2 1600

30 2 1700

29 2 1800

29 2 2400

0 5 1996

8 0 1997

3 9 1752

32 1 2004

18 13 2002

99 9 9999

111 111 111

23 4 1564      (Birth of William Shakespeare)

15 7 1606      (Birth of Rembrandt van Rijn)

21 3 1685      (Birth of Johann Sebastian Bach)

4 7 1776       (Declaration of Independence)

10 8 1787      (Mozart composes 'Eine Kleine Nachtmusik')

10 12 1815     (Birth of Augusta Ada Byron King, Countess of Lovelace)

14 3 1879      (Birth of Albert Einstein)

23 6 1912      (Birth of Alan Turing)

20 7 1969      (Neil Armstrong and Edwin Aldrin land on the moon)

31 3 3001      (1000 years from today)

0 0 0
```