**Java program:** Prob03.java

**Input File:** Prob03.in.txt

**Output:** Your output needs to be directed to stdout (i.e., using System.out.println())

**Introduction**

Not all years are created equal. Every now and then (approximately every four years), we get an extra day to enjoy because of a leap year. Leap years occur because we need to keep the calendar year in sync with the astronomical year, so we insert an extra day (February 29th) to correct the difference between the calendars.

Most of the world uses the Gregorian calendar to keep track of the calendar year. The Gregorian calendar was first used in the year 1582, so there were no leap years before that time. Under this calendar system, years that are not leap years are called common years. Here is the pseudo code for telling the difference between common years and leap years:

* If the year is prior to 1582, then it is a common year
* Else if the year is not divisible by 4 then it is a common year
* Else if the year is not divisible by 100 then it is a leap year
* Else if the year is not divisible by 400 then it is a common year
* Else the year is a leap year

Your task is to write a program that will tell whether or not a given year was a leap year.

**Program Input**

The first line of the file Prob03.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case will have the following input:

* The first line of each test case will contain a positive number N denoting the number of years that follow
* The next N lines will contain a single year per line

**Example Input:**

2

1

1984

3

1999

2001

4000

**Program Output**

Your program should simply print the word Yes if the year was (or will be) a leap year, and No if it was not (or will not be).

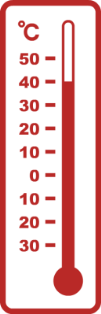
**Example Output:**

Yes

No

No

Yes

**Java program:** Prob04.java

**Input File:** Prob04.in.txt

**Output:** Your output needs to be directed to stdout (i.e., using System.out.println())

**Introduction**

Talking about the weather has been one of the most common small-talk subjects since the beginning of time. “Whew, can you believe this heat wave? It’s been 38 degrees for days!” Wait, what? Anyone who lives in Texas should know better than to complain when the temperature gets above 38 degrees. Or should they? It all depends on the temperature scale!

Here in the United States, we still use the Fahrenheit temperature scale, while most of the rest of the world uses the Celsius scale. Just like talking to people who speak different languages, if we want to be able to talk about the weather with someone who uses a different scale, we need a translator. That’s where you come in!

Your job is to write a temperature translator program. Here is a formula that relates a temperature measured in Fahrenheit (F) to a temperature measured in Celsius (C):

**Program Input**

The first line of the file Prob04.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case will have the following input:

* The first line of each test case will contain a positive integer N denoting the number of temperature conversions that follow.
* The next N lines will contain a single temperature value in the following format:

<Number><Scale>

The number and the scale will be separated by a single space. The number could have a decimal point, but will have no more than one digit after the decimal. The scale will either be C or F.

**Example Input:**

2

3

0 C

212 F

50.0 C

4

98.6 F

-6 C

40.1 C

123.4 F

**Program Output**

Your program should convert the temperature to the other scale and print it out for easy reading. Your output lines should follow the following format:

<OriginalNumber><OriginalScale> = <ConvertedNumber><NewScale>

Your original number should appear just as it is from the input file, but your converted number should always be printed to the nearest tenth. See Appendix A for more information on how you should round.

**Example Output:**

0 C = 32.0 F

212 F = 100.0 C

50.0 C = 122.0 F

98.6 F = 37.0 C

-6 C = 21.2 F

40.1 C = 104.2 F

123.4 F = 50.8 C

**Java program:** Prob05.java

**Input File:** Prob05.in.txt

**Output:** Your output needs to be directed to stdout (i.e., using System.out.println())

**Introduction**

If a city or a country wants to measure how much income its residents have, one such measure is called

“Per Capita Income”. Per capita income, also known as income per person, is the average income of the people in an economic unit such as a country or city. It is calculated by taking a measure of all sources of income in the aggregate (such as GDP or Gross National Income) and dividing it by the total population.

Besides just knowing the per capita income for a given point in time, it is useful to look at the trend of per capita income over a period of time to diagnose the economic health of an area. This is where you come in. You have been asked to create a program that will graph the per capita income of a region over time.

**Program Input**

The first line of the file Prob05.in.txt will contain a positive integer T denoting the number of test cases that follow. Each test case will have the following input:

* The first line of each test case will contain the name of the region
* The second line of each test case will contain a positive number N denoting the number of data points for the region
* The next N lines will contain two space separated numbers representing a single data point for the region in the following form:

<PER\_CAPITA\_INCOME><YEAR>

Where PER\_CAPITA\_INCOME is a dollar amount that will have two decimal places and year is a four digit year. Each data point will have a distinct year.

**Example Input:**

1

USA

5

15777.0 1993

28829.0 2013

4141.0 1973

23276.0 2003

9494.0 1983

**Program Output**

Your program should output a horizontal bar graph of the data for each region. The first line of each region’s output should be the name of the region followed by a colon. The next N lines should be the data points for that region sorted ascending by their year. Each line should contain the year followed by a space followed by some number of asterisks. Each asterisk should correspond to $1000, which means you will need to round the per capita income to the nearest thousand dollars. See Appendix A for more information on how you should round.

**Example Output:**

USA:

1973 \*\*\*\*

1983 \*\*\*\*\*\*\*\*\*

1993 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2003 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2013 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*