Problem P Unambiguous Dates 50 points

One difficulty with dates is that different countries have different ways of writing them, so a date given in normal NZ form is ambiguous. For example, 4/2/95 means 4th February 1995 in New Zealand, 2nd April 1995 in the US, and 95th February 1904 in Japan. The only dates you can really trust are ones like 9/9/9, which means the 9th September, 1909, in all three countries.

It is possible to eliminate this ambiguity if one is willing to only use dates of the form x/x/x, and accept strange-looking dates. For example, the date 88/88/88 means the 88th day of the 88th month, 1988. The 88th month of 1988 is April, 1995; the 88th day of April in 1995 is the 27th June. So, in all three countries, 88/88/88 means 27th June, 1995 - this day has an unambiguous date using 1900 as the zero point. Every day has an unambiguous date if the zero point year is chosen correctly.

Input for this problem will be from the file PROBLEMP.DAT, and will consist of lines, each containing a date in the form dd mmm yyyy where dd is the day of the month, yyyy the year (including century - yyyy will always be larger than 1000) and mmm one of the strings jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec. The file will be terminated by a line consisting of a single #.

Output must be an unambiguous date for each line of input, in the form nn/nn/nn xxxx where nn is a positive integer and xxxx is the zero point year for this date. The number nn must be the smallest possible number giving an unambiguous date.

Do not worry about non-Gregorian dates - assume the current calendar has always held, which states that the days in the months are respectively 31,28,31,30,31,30,31,31,30,31,30,31, except that the second month has 29 days in every year which is divisible by 4 and is not divisible by 100, unless it is also divisible by 400 (thus feb 1900 had 28 days but feb 2000 will have 29 days). When counting forward a large number of days, you have to know which year each February is in; for example, to count 129 days forward from December 1991, December and January have 31 days, February 29 and March 31, so the date is 7 April, 1992.

EXAMPLE

Input

```
07 jul 1907
2 sep 1995
04 feb 1995
29 feb 1200
#
```

Output

```
7/7/7 1900
125/125/125 1860
188/188/188 1791
305/305/305 869
```

Problem Q **Indoor Cricket** 50 points

Cricket is a wonderful game, and some people want to play it all the time (in fact, potentially, a game can last for ever). To recapture some of the excitement and interest of cricket even in the winter, a game called Indoor Cricket was invented in about 1920 (not related to the bastardised version of cricket that currently has that name). It is played with two octagonal cylinders, which have numbers or words on their 8 faces. The rules are as close as possible to the rules of the game of cricket.

Each team has 11 players - one team is "batting", and uses the "batting" cylinder, and one is "bowling" and uses the "bowling" cylinder. The "batting" team plays first, and has two "batters" in play currently. The first "batter" rolls one cylinder; when it stops rolling, the uppermost face will either display a number from 1 to 6 (which is the number of runs for that play), or the number 0 or the words "Hows that?". If a run count turns up, the "batter" has made that many runs for his team. If this is an even number, he or she continues playing. If it is an odd number, the other "batter" must play (the batters have "changed ends"). When a 0 or "Hows that?" turns up, the score made for that play is 0, and the bowling team must roll their cylinder (it does not matter who plays for them; there are no "overs" in this game). The "bowling" cylinder has 6 of its sides which give ways of going out, and two sides which state "No ball" and "Not out". If the bowling team rolls one of the 6 sides giving an "out", the "batter" currently playing is out, and must be replaced by the next player from the "batting" team, who will make the next play; otherwise (for "No ball" and "Not out") the current "batter" can continue playing. When ten "batters" have gone out, the "batting" team's turn ("innings") has finished - the eleventh "batter" is "Not out". The teams swap cylinders. Each team gets two turns at being the "batting" team, so the game has a total of four "innings"; a total of 40 "batters" have gone out.

We will assume the batting cylinder carries the numbers 0 to 7, with 7 representing "Hows that?". The eight sides of the bowling cylinder are as follows:

0 No Ball	3 LBW	6 Run out
1 Not out	4 Caught	7 Hit wicket with bat
2 Bowled	5 Stumped	

Input will be from the file PROBLEMQ.DAT and will consist of lines containing a list of numbers from 0 to 7, separated by one space. There will be no more than 30 numbers on each line. These are numbers for successive rolls. Using the rules given above, you must print a summary of the game, as shown in the sample output below, giving the batter number. the score each batter makes (the players must be listed in their batting order), how they got out (or "Not out" if they were the eleventh batter at the end of an innings), the total for each innings and which team are the winners (had the highest total). Ignore any part of the input after 40 "batters" have gone out; the file will contain enough data for one game, plus a bit more (the players get carried away by the excitement of the game and forget to stop). Note the layout of the numbers in the output; the units digits of the batter number, and of the scores and total, are in the same column, and there is always at least one blank between fields (you may assume a batter never scores more than 999). If the total of both innings for both teams is the same, the result line should read "The result is a tie".

EXAMPLE

Input

```
1 0 3 6 3 7 5 2 1 4 2 0 1 3 1 7 2 4 1 6 3 2 7 6 1 3 5 1 0 2
0 3 4 7 1 0 2 5 4 1 0 5 2 3 6 0 1 5 2 6 7 2 0 2 1 7 4 0 3 2
5 6 1 7 0 2 1 4 2 6 4 1 7 3 1 0 2 3 0 1 4 5 2 6 3 7 1 4 5 0
1 6 3 2 7 4 1 0 3 6 1 2 1 0 3 6 3 7 5 2 1 4 2 0 1 3 1 7 2 4
1 6 3 2 7 6 1 3 5 1 0 2 0 3 4 1 7 4 0 3 2 5 6 1 7 0 2 1 4 2
6 4 1 7 3 1 0 2 3 0 1 0 1 4 5 2 6 3 7 1 4 5 0 1 6 3 2 7 0 3
6 1 2 1 0 3 6 3 7 5 2 1 4 2 0 1 3 1 7 2 4 1 6 3 2 7 6 1 3 5
1 0 2 0 3 4 1 7 6 1 3 5 1 0 2 0 3 4 7 1 0 2 5 4 1 0 5 2 3 6
0 1 5 2 6 3 7 5 2 1 4 2 0 1 3 1 7 2 4 1 6 3 2 7 6 1 3 5 1 0
2 0 3 4 1 7 4 0 3 2 5 6 1 7 0 2 1 4 2 6 4 1 7 3 1 0 2 3 0 1
0 1 4 5 2 6 3 7 1 4 5 0 1 6 3 2 7 0 3
```

Output

```
Team 1
Batter  1    1 Stumped
Batter  2   0 LBW
Batter  3  18 Bowled
Batter  4  33 Not out
Batter  5   7 Run out
Batter  6   6 Bowled
Batter  7   0 LBW
Batter  8   4 Bowled
Batter  9   5 Stumped
Batter 10  13 Bowled
Batter 11   0 Bowled
Total      87
Team 2
Batter  1  25 Bowled
Batter  2   0 Caught
Batter  3   0 LBW
Batter  4  10 LBW
Batter  5  21 Caught
Batter  6  23 LBW
Batter  7   4 Stumped
Batter  8   7 LBW
Batter  9  18 Bowled
Batter 10  13 Not out
Batter 11   7 Run out
Total     128
Team 1
Batter  1   6 Bowled
Batter  2   4 Caught
Batter  3   0 LBW
Batter  4  29 Bowled
Batter  5   0 LBW
Batter  6  10 LBW
Batter  7  27 Stumped
Batter  8  30 LBW
Batter  9  18 Bowled
Batter 10  13 Not out
Batter 11   7 Run out
Total     144
Team 2
Batter  1   6 Bowled
Batter  2   4 Run out
Batter  3   0 LBW
Batter  4  25 Stumped
Batter  5   6 Bowled
Batter  6   0 LBW
Batter  7   4 Bowled
Batter  8   5 Stumped
Batter  9  25 Bowled
Batter 10  13 Not out
Batter 11   7 Run out
Total      95
Team 1 are the winners
```

Problem R Collisions 50 points

A new fairground game has been invented by the Dodgem Car company, for use at the large amusement park which is going to be set up in orbit sometime next century (the company is very forward-looking). The vehicles will simply be small space-ships, which will travel in straight lines, starting from specified points and stopping at specified points (acceleration and deceleration are practically instantaneous). All the space-ships will start at the same time and all will stop exactly 10 minutes later. The company are planning to use up to 100 space-ships at a time. The excitement will be caused by the fact that sometimes the space-ships will come very close to each other. The company wants a program which will determine when two of their spaceships are going to have a collision.

Input for this problem will be from the file PROBLEMR.DAT and will consist of lines, each containing six integers, separated by blanks. The first 3 numbers give the starting position of a spaceship (in terms of metres, relative to some coordinate system) and the next 3 the stopping position of the spaceship. All numbers are positive and less than 1000. The data set will be terminated by a line containing 6 zeroes; the file will be terminated by a data set with nothing in it (ie two adjacent lines with 6 zeroes). Since acceleration and deceleration take a very short time, the space-ships can be regarded as moving with constant speed; the speed of a space-ship depends on the distance it has to travel, since they all start at the same instant and stop at the same instant.

Output must be either the words "No collision occurs", or a line giving the numbers of the two spaceships which collide first, left-justified and separated by a single space. Spaceships are numbered from 1 within each data set. A collision occurs if, at some time, all the coordinates of the two spaceships are the same when rounded to an integer You may assume there will never be more than two space-ships colliding within the same one-tenth of a second.

EXAMPLE
Input
```
410 522 310 511 330 700
0 0 0 0 0 0
410 522 310 511 330 700
511 330 700 410 522 310
0 0 0 0 0 0
0 0 0 0 0 0
```

Output
```
No collision occurs
1 2
```

Problem T **Cryptic Crosswords** 150 points

Cryptic crossword puzzles often have clues which contain an anagram of the answer as well as a (very cryptic) description of the answer. For example:

"Seeing people eating spaghetti got Bob angry (5)"

(where "(5)" means that the length of the answer is 5 letters) could be a clue for the word "bigot". For this question, an "anagram" of a word is a set of adjacent letters (ignoring spaces, punctuation and capitalisation) which form the word, possibly when rearranged. Here, the "i" from "spaghetti", "got", and "B" from Bob form an anagram of "bigot". The letters must be adjacent (ignoring spaces and punctuation) for the anagram to exist. These crosswords are very difficult to solve, and it would be nice to have some computer assistance.

One way of doing this is to identify possible topics by using words from the clue, and then search a thesaurus for related words which are contained in the clue as anagrams. In the above, either the word "people" or "angry" could eventually lead to "bigot", if the thesaurus search is large enough.

Input will be from a file called PROBLEMT.DAT, and will consist of lines, each containing several words which are separated by one or more blanks. Each line will be less than 80 characters and each word less than 20 characters. The first part of the file will be a thesaurus; the words on each line will all have related meanings. For this preliminary version, there will be no more than 500 lines in the thesaurus, each with no more than 7 words on it, each word consisting entirely of lower-case letters. There will be no more than 1000 different words in the thesaurus; the same word can appear on many different lines, of course. The end of the thesaurus will be marked by a line consisting of a single #. The second part of the file will be a list of cryptic crossword clues, each clue being a line consisting of words separated by spaces as before, and terminated by the word "(n)", where n is a number giving the length of the word sought. The end of the clue list is marked by a line consisting of a single #.

You must give one output line for each clue. For each word W in the clue, you must consult the thesaurus and see if any related word (a word on the same line as W) of length n is present in the clue as an anagram, ignoring spaces and punctuation. Then using each related word instead of W, test for words of length n related to these, and so on until every word in the thesaurus which is related to W, however distantly, has been examined. All the different words which are found must then be written out in alphabetic order, as shown below. The list must not include words which were present in the clue in their normal form - only words which were present as anagrams. If no words are found, print a blank line.

EXAMPLE

Input

```
angry furious cross wild hate
men women people love hate
native free wild growing weed
racist hate bigot nazi
love marry elope kiss
weed toadstool mushroom fungus slime
excrement vomit slime pimples
sick ache vomit pangs
#
No mew people - men!(5)
No mew people - women!(5)
Seeing people eating spaghetti! got Bob angry (5)
#
```

Output

```
women

bigot elope pangs
```

Problem U **Digit Sums** 150 points

Given 3 positive integers A, B and C, find how many positive integers less than or equal to A, when expressed in base B, have digits which sum to C.

Input will be from the file PROBLEMU.DAT and will consist of a series of lines, each containing the numbers A, B and C. 0 < A <= 1,000,000,000, B <= 100, C <= 1,000,000,000. The numbers A, B and C are given in base 10 and are separated by one or more blanks. The input is terminated by a line containing three zeros.

Output will be the number of numbers, for each input line (it must be given in base 10).

EXAMPLE
Input
```
100 10 9
100 10 1
750000 2 2
1000000000 10 40
100000000 100 200
0 0 0
```

Output
```
10
3
189
45433800
666303
```

Problem V **Fitting Rectangles** 150 points

A set of rectangles can possibly be fitted together into a square. For example, the 9 rectangles 5x6, 3x3, 4x4, 2x8, 6x9, 4x4, 3x3, 3x8, 5x6 can be fitted into a 14x14 square as follows:

```
***************
*     *   * * *
*     *   * * *
*     *   * * *
*     ***** * *
*******   * * *
*     *   * * *
*     *   * * *
*     *********
*     *    *  *
*     *    *  *
*     *    ****
*     *    *  *
*     *    *  *
***************
```

Input will be from the file PROBLEMV.DAT and will consist of several sets of rectangles, each set starting with a single line giving the number N of rectangles (N <= 20), followed by N lines giving the rectangles, each described by 2 numbers separated a blank. The size of the resulting square will be no bigger than 24x24. The file will be terminated by a line consisting of a zero value for N.

Output will be a figure consisting of * and blanks as shown above. You may assume that an answer is always possible. You may arrange the rectangles inside the square in any way you like, including rotating them, as long as every rectangle is used and the square is filled completely. Each square shown should be separated from the next by a single blank line (any blank lines at the end of the output will be ignored).

EXAMPLE

Input
```
2
2 4
4 2
2
6 4
2 6
0
```

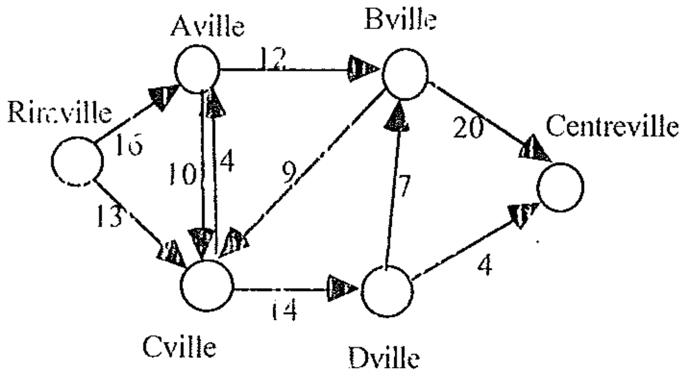
Output
```
*****
*   *
*****
*   *
*****

*******
*     *
*     *
*     *
*******
*     *
*******
```

Problem W **Factory Size** 150 points

The Blue Blahs company wants to start up a new Blah manufacturing plant at Rimville. The central warehouse (where all the Blahs are stored) is in Centreville, and the Blahs have to be shipped from Rimville to Centreville. To do this, the company can hire space on various shipping firms which make various runs, via various towns (Aville, Bville etc). These routes and capacities are fixed, and the company needs to know how much of its product it can ship each day, so it knows what size factory to make.

Input will be from the file PROBLFMW.DAT and will be a set of scenarios, each a series of lines which give two town names (each a single word - there are no more than 70 towns) and the maximum amount (an integer, up to 100) that can be shipped between them (from the first town to the second town). Note that not all towns have a shipping connection. Each series will be terminated by a line consisting of a single #. The file will be terminated by a line consisting of two ##.



Output must be a single number for each scenario giving the maximum amount that can be shipped between Rimville and Centreville, which will always be two of the towns mentioned. Rimville will always appear as the first word on at least one line and Centreville will be the second word on at least one line.

EXAMPLE
Input
```
Rimville Aville 16
Aville Bville 12
Bville Centreville 20
Rimville Cville 13
Cville Dville 14
Dville Centreville 4
Aville Cville 10
Cville Aville 4
Bville Cville 9
Dville Bville 7
#
##
```

Output
```
23
```