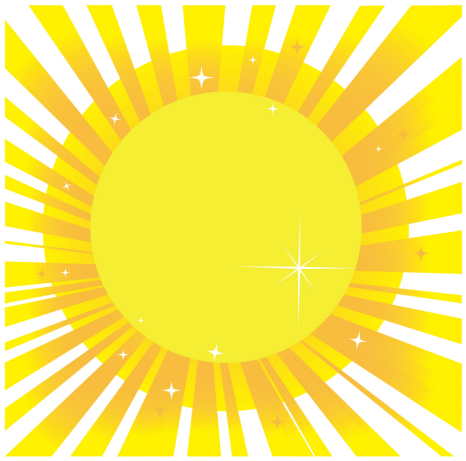
COMPUTER AND INFORMATION SCIENCE

UNIVERSITY OF OREGON

Nineteenth Annual
University of Oregon
Eugene Luks Programming Competition

2015 May 02, Saturday
am10:00 – pm2:00



Problem Contributors
Daniel Lowd
Eugene Luks
Chris Wilson

Technical Assistance
Paul Bloch

A: HAIKU CAN BE FOUND ALMOST ANYWHERE IF YOU KNOW TO LOOK FOR THEM.

Haiku is a type of Japanese poetry. Each haiku poem consists of three lines of 5, 7, and 5 syllables, respectively. It is easy to create poems of this form, even by accident. For example, this sentence can be formatted into a haiku.

```
For example, this
sentence can be formatted
into a haiku.
```

You have recently been hired by the Totally Random Internet Poetry Project (TRIPP) to automatically discover haiku in web pages and other text sources. A sentence is considered a haiku if it contains exactly 17 syllables and it can be broken into lines of 5, 7, and 5 syllables without splitting any words across lines or changing the word order. A group of several sentences is also considered a haiku if they satisfy the same criteria. A sentence may be split across several lines, but partial sentences are not allowed. For example:

```
Programming on a
Saturday is great fun. Will
your team win first place?
```

You will work with a coded representation where each word is represented by an integer specifying its syllable count and the end of each sentence is represented by a zero. For example, the previous two examples would be coded as follows:

```
1 3 1 2 1 1 3 2 1 2 0
3 1 1 3 1 1 1 0 1 1 1 1 1 1 0
```

You must write a program that will determine if a haiku can be constructed out of a sequence of sentences, and if so, provide the index of the first sentence that begins a haiku. The first line of the input contains a single integer K<100 specifying the number of sequences to process. Each of the following K lines consists of an integer sequence representing the syllable counts and sentence breaks. Each line contains no more than 200 integers and ends with a sentence break ("0"). For each sequence, you should output the number of the first sentence where a haiku can be found or "No haiku" if none is present.

Sample Input

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

Sample Output

```
1
No haiku
No haiku
3
2
```

B: THIS PATH IS JUST RIGHT

Once this busy term is over, you are wishing to go for a nice long hike in the Oregon Coast Range. You have been looking at maps, and see that there are many possible trails between Siuslaw and Tillamook. However, you find yourself in a quandary, since the shortest path between these cities is too short and boring, while the longest path might take far too much time. What you would like to know is what is the average length of such a path?

The average length of a path from S to T in this sense is defined to be the sum of the length of all distinct paths from S to T divided by the number of such distinct paths. For this to be well defined and computationally simpler, the graph representing the paths is an acyclic directed graph.

The input will start with an integer K, the number of graphs to be processed. The description of a graph will start with a pair N M, where N in the number of nodes in the graph and M is the number of edges. This is followed by M lines of the form I J W, where W>1 is the integer length of the edge from I to J. It will always be true that $1 \le I < J \le N$, which guarantees that there are no cycles. Furthermore, you may assume that there is always at least one path from S=1 to T=N.

The output should be one line for each graph. It should say "graph c: ", where $1 \le c \le K$ is the graph number, followed by the number of distinct paths from S to T, a space, and the average length of a path. This average should be the sum of the lengths of all paths divided by the number of distinct paths, rounded down to the nearest integer. For example, in graph number 1 below, there are 4 distinct paths from node 1 to node 5, and the sum of the lengths is 70. Thus the average is 70/4, which rounds down to 17.

Sample Input

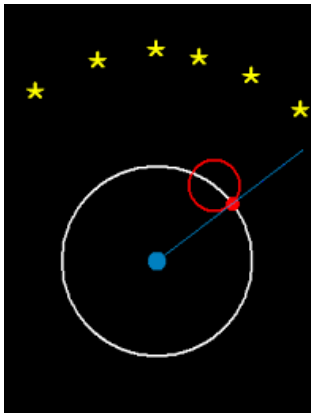
```
2
5 7
1 2 7
1 3 10
2 4 8
2 5 3
3 4 5
3 5 10
4 5 5
10 12
1 2 1
1 3 1
2 4 1
3 4 1
4 5 1
4 6 1
5 7 1
6 7 1
7 8 1
7 9 1
8 10 1
9 10 1
```

Sample Output

```
graph 1: 4 17
graph 2: 8 6
```

Comment:
On the test graphs, the number of paths will fit into an `int`, but the sum of all lengths may require a `long`.

C: PLANET TENALP



"Planettanelp" is a weak approximation to the name of a planet whose inhabitants sing their words, the 10 letters of the language bringing forth uniquely mellifluous tones. Fortunately, since the Planettenalp alphabet is also used as the digits in a decimal system, we can transliterate to {0,1,2,3,4,5,6,7,8,9} for the purposes of this problem.

Conversation is a joy to the ear on Planettenalp, but the most pleasant of all sounds are the words/numbers that appear as palindromes. As a result, parents are inclined to use these musical snippets in naming their children. There are, however, some legal naming rules. First names must be multiples of the family name, cannot be longer than 100 letters, with feminine names having even length, and masculine names having odd length. But the most burdensome restriction is that names are confined to use a selected set of digits: since it has a ritualistic significance, "0" cannot be used in a first name, and the digits must have appeared on the birth certificate, e.g., in the standard representation (hour/day/month) of the time the child was born, though they can each be used multiple times.

In the setting of this problem, you will be well-paid by parents if you can find names for their babies. You are given a family name (a positive integer), the set of non-zero digits appearing on the birth certificate, and the gender of the child. You are to determine the *length* of the least legal palindromic name, if there is one.

The first line of the input will indicate the number N of children to be considered. The first line of data for each child indicates the family name F (with F< 10000) followed by the gender (Girl or Boy). The second line gives the set D of usable digits.

The output, for each child, should be the number of digits in the least legal palindromic name/number constructible using digits in D and divisible by F, or the response "Dammit, I'm mad!" if there is no legal choice.

Sample Input

```
7
9 Girl
3
10 Boy
6
8 Girl
3 7
7 Boy
2 3 5
11 Girl
3
11 Boy
1 2 3 4 5
19 Girl
2 5
```

Sample Output

```
6
Dammit, I'm mad!
Dammit, I'm mad!
3
2
3
4
```

D: WHO BROKE MY PALINDROME?

Everybody likes palindromes, especially the curators and patrons of the Palindrome Museum. Thousands of visitors every day visit the museum to see the famous classics, such as ‘a man, a plan, a canal, panama’ and ‘never odd or even’.

You have just taken a programming job at the museum’s palindrome repair shop, because accidents do happen. In one case, an untended visitor was allowed to touch the classic ACROBATSSTABORCA and absent-mindedly pulled out some of the letters while talking on a cell phone. Fortunately a horrified guard was able to recover the letters in order, OBSARCA, and they were sent along with what remained, ACRATSTBO, to you in the repair shop. Your job is to write a program to determine if all the letters in the two strings of characters, OBSARCA and ACRATSTBO, can be merged into a palindrome.

Note that a proper merging of two strings into a palindrome must use all the characters of the two strings. The merging must maintain the left-to-right order of the characters from each string.

Specifically, you must provide a program to determine if a series of pairs of strings can be merged into a palindrome. It should take input consisting of several lines – the first line contains a single integer K<20, which is the number of pairs you need to process. After this first line are 2K lines, representing the K pairs of strings. Each string consists of at most 150 characters, all capitalized letters A-Z. There is no punctuation, no spaces, and every string has at least one character. You should output for each pair YES if the two strings can be merged into a palindrome and NO otherwise.

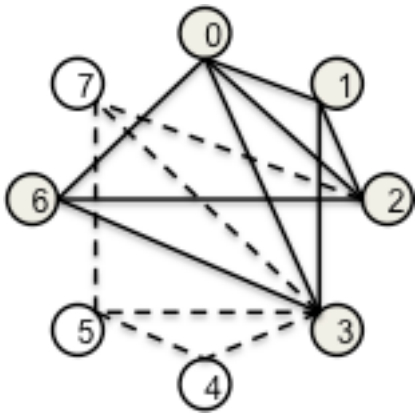
Sample Input

```
4
AA
AAAPAAA
AA
AAAPAA
AAA
AAAPAA
ACRATSTBO
OBSARCA
```

Sample Output

```
YES
NO
YES
YES
```

E: MYFACE.COM, INC.



MyFace is the hottest new social network startup. Like many other social network sites, each user has a list of friends with whom they share status updates and other information. These friendships are always symmetric, so if user A is friends with user B then user B is also friends with user A. MyFace is adding a new "MyFace Groups" feature, which allows groups of friends to create shared pages for organizing events and discussions. In order to promote this new feature, MyFace would like to suggest possible groups to its users. Your job is to write a program to find these potential friend groups.

A *potential friend group (PFG)* is a set of users where every user is friends with at least half of the other users, rounded up. Larger friend groups are more interesting, so you must find the largest PFG in the graph. You will be given several graphs, each representing a small portion of MyFace's total social network.

See the figure above for a simple example. Users 0, 1, 2, 3, and 6 define a potential friend group of size 5 because each of the 5 users is friends with at least half of the other 4 users.

Your program must accept input according to the following format. The first line contains a single integer K<25 specifying the number of social network graphs to process. This is followed by K graph descriptions. The first line of each graph description is a single integer N<20 specifying the number of users in the graph. The next N lines specify the friends for each of the users in order, with one friends list on each line. The friends list refers to friends by their zero-based index. You may assume that each friends list is sorted numerically and that all friendship relations are symmetric. For each graph, your program must print the size of the largest PFG.

This portion of the sample input defines an 8-node graph with the same structure as the example above.

Sample Input

```
2
4
1 2 3
0 2
0 1 3
0 2
8
1 2 3 6
0 2 3
0 1 6 7
0 1 4 5 6 7
3 5
3 4 7
0 2 3
2 3 5
```

Sample Output

```
4
5
```