ᚦᚺᛁᚱᛞ ᚠᛁᚾᚾᚢᛚᚠ ᛜᚢᛁᛚᚠᛋ ᚲᛟᚾᛏᛖᛋᛏ

(Third Annual Juilfs Contest)

May 22, 2011

0001001110011000111110100100110
0100100010000001101000110100001
1010010101110100010110000000100

COMPUTER AND INFORMATION SCIENCE

1101000011100110110010000100100
0001000110100010001000000010011
1010001101010001010100100100011
0010010001001111000011100010101

O | UNIVERSITY OF OREGON

1110001001000011001110011010011
0011100001110100000100101010010

A. Identification 7 Points

In order to maintain stability and integrity in dwarven society, the High Council on Dwarven Identification and Nomenclature's New Naming System Working Group has developed a new system to name dwarves.

According to this system, a name consists of a fixed number of fields with a fixed number of sub-names specified for each field. The sub-committee on Naming System Evaluation has asked you to evaluate several different configurations to determine which one should be adopted.

The chief maildwarf would also like you to determine what the most valuable name in the system is, as he typically reserves the highest-value name to use when mail needs to go to multiple sources simultaneously.

The values for lowercase letters in the Dwarvish alphabet are given in the table on the right. Capital letters are worth double their lower-case value. All other characters are worth zero.

0	1	2	3	4	5	6
a h o v	b i p w	c j q x	d k r y	e l s z	f m t	g n u

The input consists of a line $0 < n < 100$ indicating the number of problems. Each problem begins with a line containing a number $0 < m < 1000$ indicating the number of fields. This is followed by m lines containing names separated by commas. Names may contain any character except commas and line break characters. Names may also be empty.

For each problem you should output a number on a line by itself listing the maximum number of names that can be generated using the system. The dwarves avoid using the same sub-name in two consecutive fields, so no trickery is needed to determine uniqueness. On the next line after the number, output the highest-valued name that can be generated by the system. There are some letter combinations that are considered aesthetically better than others, although the letter score is the same. These are listed first in the list, and should be printed as part of the highest valued name.

Sample Input	Sample Output

`2`
`2`
`Gimli,Gloin,Durin,Dwalin,Mim,Thorin`
`ƀOakenshield,ƀIronfoot,ƀGoblinsbane`
`3`
`Fil,Kil,Or,Nor,Dor,Bi,Bo,Bom,O,Gl`
`in,bur,fur,oin`
`,ƀOakenshield`
	`18`
	`Gimli Goblinsbane`
	`80`
	`Glfur Oakenshield`

Note: ƀ is used to show a space character in the input. This is done for clarity. The actual character in the input is a space.

B. ᚦᚱᚠᚢᛊᛗᛇᚷᚱᚠᚦᛇᛟᛟ ᛟᚠ ᛊᛟᚠᛚᛊ

7 Points

At the beginning of the first age, Aulë fashioned 1000 dwarves from stone. When Dwarves die, their soul goes into the next available dwarf baby. Dwarvish women can only become pregnant (aside from the usual mechanism) if there is a dwarf soul available. At the time of rebirth, dwarves, of course, forget their former life, but dwarf society still recognizes the achievements from previous lives. The problem is that since dwarf births are relatively rare, there may be several years upon the death of a soul before a child body becomes available. For example, during a great battle, many dwarves may fall, but it may take many years for them all to become reborn. During that time, the dwarves become confused about who is being born, and it is particularly important after a great battle to recognize the heroes and the villains.

After a century of peaceful coexistence, the dwarves entered the current period of strife. During that century all dwarves were alive. So, now they have a clean slate, but they would like you to keep track of who is who. For input, you are given the sequence of births and deaths, in chronological order. The souls are reincarnated in the order in which they died. At the end of the death/birth record, the input contains queries about names of currently living dwarves. As output, you should give the names that have been given that dwarf through its lives. Since there are so few dwarves, names are unique.

The input begins with a line with two numbers $0<n<50000$ and $0<m<1000$. The first number, n, indicates the number of lines of death and birth data. The second number, m, indicates number of queries. Each death or birth data line starts with a letter D or B, followed by a space, followed by a name. The name contains only upper and lower alphabetic characters, no spaces or other special characters. Your program should output m lines. For each of the m query names, you should output a line that contains all of the names, in order from earliest to latest, that have been used by that dwarf.

Sample Input	Sample Output
12 3 D Gimly D Alaric D Fregar B Gorem D Durak B Furgil B Trygg D Gorem B Thorin B Morag D Morag B Loki Loki Thorin Trygg	Gimly Gorem Morag Loki Durak Thorin Fregar Trygg

C. ᛞᚹᚠᚱᚠᛗᛏ ᛊᚢᚱᚡᛇ ᛟᚠᚠᛁᚳᛖ

10 Points

You are a member of the dwarven survey office, newly in charge of mapping the nation's tunnel system. Dwarven law requires that each intersection, an endpoint of a tunnel, be given an identifier so that the survey office may record the start and end of each tunnel. However, due to the miracles of technology, computers have now become mainstream and the contents of all the tunnel cards in your file cabinets must now be input into a single database so that a full report of the tunnel system can be compiled. Your task is to write a program that will take as input a system of tunnels and return a list of the intersections and their degree. The degree of an intersection is the number of tunnels that start/end at that intersection.

All records in the file cabinet are in the following form:

`A,B`	A tunnel connecting existing intersection *A* to existing intersection *B*, so the degree of both *A* and *B* are incremented.
`A'C'B,D`	A new tunnel was dug, starting at the newly created intersection *C* between existing intersections *A* and *B*. This tunnel ends at existing intersection *D*. The degree of intersection *C* is set to three and the degree of intersection *D* should be incremented.
`A,C'B'D`	The same as the previous case, except reversed, such that intersection *A* exists and has its degree incremented, and intersection *B* is created with degree 3.
`C'A'D,E'B'F`	A new tunnel was dug, starting at newly created intersection *A* and ending at newly created intersection *B*. Both new intersections are created with degree 3.

Although the examples use capital letters, intersections may be labeled with any combination of letters and numbers. The file cabinet is arranged in historic order, so an intersection listed as new (eg the `B` in `A'B'C`) will not be listed in a preceding card, but may be listed on subsequent cards. The other intersections (eg the `A` and `C` in `A'B'C`)

The input to your program will begin with a line with a single number $0 < n < 10{,}000$, which indicates how many problems follow. Each problem will be on a separate line, enclosed in paratheses. Each problem line will have a list of records from the file cabinet, separated by semicolons. Your output should have one line per intersection with the name of the intersection, a colon and the degree of that intersection. Your output for each problem should have the intersections listed in order from highest degree to lowest degree. The intersections with equal degrees should be listed in normal dwarven order. Normal dwarven order lists the uppercase letters in alphabetic order first, followed by the numberic digits, from lowest to highest, followed by the lowercase alphabetic letters in alphabetic order. (Coincidentally, this is also ASCII order.)

Sample Input

```
3
(A,B;A,C;A,D;B,C;D,C)
(A,B;A,C;A,D;B,C;D,C;A,E;D,E;E,B;E'F'D,A;E'G'F,C;E'H'B,A'I'B;A,I)
(A,B;A,C;A,D;B,C;D,C;A,E;D,E;E,B;E'F'D,A;E'G'F,C;E'H'B,A'I'B;A,I;G,C;D'J'B,A'K'C;F,B;K,I;G,H;C,D;J,B;K,F;G,
B;C'L'K,H;F,D;K'M'A,E'N'A)
```

Sample Output

```
A:3
C:3
B:2
D:2
A:6
B:4
C:4
D:4
I:4
E:3
F:3
G:3
H:3
B:7
A:6
C:6
D:6
G:6
F:6
I:5
H:5
K:5
J:4
E:3
L:3
M:3
N:3
```

Note: The input for the third problem takes up two lines because of type-setting constraints, but it does not contain an internal line-break between parentheses.

D. ᛋᛈᚺᛗᚱᛁᚲᚠᛏ ᚷᛗᛟᚹᛗᛖᛏᚱᛟ

10 Points

Dwarves love gold, which is mined from the planet's crust. Entrances to mines are on the planet's surface, or outermost sphere. When travelling between mine entrances, however, the fastest path is a straight line going through the planet's interior. Fortunately for the dwarves, they have balrogs to make such straight tunnels. Fortunately for you, the dwarves need help calculating the length of these tunnels.

Dwarves record location on and in the planet using the three co-ordinate measurements of arnli, atli, and onli:

Arnli (r) is the point's distance from the planet's center. It is a constant 1 at the surface, so we will exclude it here.

Atli (φ) is the angle from a line between the center and the gold point, directly below the gold star and opposite the copper point (directly below the copper star). Atli is measured in π times radians.

Onli (θ) is the angle around the planet in the positive direction, as observed from the point of view at the gold star. Onli is also measured in π times radians.

The first line of input contains a single number $0 < n \leq 1000$, the number of subsequent lines. Each subsequent line of input has two sets of (atli,onli) ordered pairs, with each in units of π radians. Output should be formatted as a newline-delimited list of distances, as measured in planetary radians, rounded to the nearest $\frac{1}{1000}^{\text{th}}$ (i.e. 3 places after the decimal point).

Sample Input	Sample Output
3 (0, 0) (.5, .5) (0, 0) (1, 0) (0, 0) (.3333, 1)	1.414 2.000 1.000

Recall that spherical co-ordinates can be changed to rectangular co-ordinates sharing the same origin via the following familiar dwarven formulae:

x = arnli * sin(atli) * cos(onli)

y = arnli * sin(atli) * sin(onli)

z = arnli * cos(atli)

E. ᛗᛋᛊᚠᚷᛗ ᛞᛗᚲᛟᛞᛗᚱ

10 Points

During the Dwarf Apocalypse, the evil, dark Dwarves gained control over the world's mines. Recently, a resistance group has been formed, led by the newly crowned king of the good, light Dwarves. The resistance group's mission is simple: retake the world's mines. After several small skirmishes, the resistance group has claimed several victories but they fear retribution. This morning they intercepted an encrypted message that they have enlisted you to decipher!

The dark dwarves' encryption methods are well-known. They write the tuple (N, e) on the walls of every cave juncture. It is well known that $N=pq$ where p and q are primes and that there exists a d such that

$$ed = 1 \bmod \big((p-1)(q-1)\big)$$

To encrypt they use the function:

$$E(M) = M^e \bmod N$$

To decrypt:

$$M = D\big(E(M)\big) = \big(E(M)\big)^d \bmod N$$

```
ExtendedEuclideanAlgorithm(a, b)
    if b == 0
        return (1, 0)
    else
        q = a div b
        r = a mod b
        (s, t) = ExtendedEuclideanAlgorithm(b, r)
        return (t, s – q * t)
```

Unfortunately, there is no known method for deciphering this encryption scheme knowing only (N, e).

But wait! Just a few minutes ago a spy has recovered the primes: p and q. The good dwarf king believes that this should be enough to decipher d! But he thinks you might need to use the Extended Euclidean Algorithm to help (which he was generous enough to provide). This algorithm takes in a, b as input and solves for $ax+by=1$, returning x and y. He also believes that the Python function `pow()` and the Java `BigInteger` class will be useful here as well.

The input will consist of two lines. The first line will have 3 space delimited integers: $0 < p < 10^{20}$, $0 < q < 10^{20}$, $0 < e < 10^{40}$, where p and q are the primes obtained through espionage and e is the publicly known encryption key. The second line will consist of 1 integer: $0<E<10^{40}$, representing the enciphered message.

The output should be a single line containing the original message.

Sample Input	Sample Output
3 7 5 17	5

F. ᚦᚢᛟᛟᛗ ᚲᛟᛟᛗᚲᚦᛟᚱ

10 Points

Dwarves have a good sense of smell when it comes to iron and gold and tin, so they are able to find deposits of precious and useful metals. Unfortunately, they do not have a good sense of smell for other things. In particular, they often encounter natural springs, which then flood the tunnels with water, making them useless. The dwarves close off the flooded tunnels, but this can sometimes leave dwarves trapped in a closed off section.

The Dwarvish Elders have asked you to write a program that takes as input the configuration of tunnels. You then have to print out which locations no longer have a route to the main hall, so rescue operations can be started. The input starts with a line containing a single number $0 < n < 100$, followed by n problems. Each problem starts with a line containing a number $2 \leq m \leq 1000$, the number of locations. This is followed by m lines of tunnel data. Each line of tunnel data starts with the name of a location followed by a space-separated list of other locations to which a tunnel exists. For example, the line

```
Haedelf Moria Nargothrond KhazadDum Belegost
```

says that there are four tunnels from Hædelf to Moria, Hædelf to Nargothrond, Hædelf to Khazad-dûm and Hædelf to Belegost. The tunnels that have been sealed off due to flooding are no longer listed in the data. Each location has a line, but it may only contain the name of the location, if all tunnels coming out of it have been flooded. Each tunnel will be listed twice, on the lines for each endpoint. The names of the locations contain only alphabetic data with no spaces, but may be a mix of upper and lowercase characters. The first location on the first tunnel line for each problem will correspond to the main hall.

Your output for each problem should consist of a single line with a space-separated list of locations that are unreachable from the main hall, in alphabetical order. In each problem, there will be at least one unreachable location.

Sample Input	Sample Output
3 5 Belegost AmonRudh Nogrod Moria KhazadDum Nogrod Belegost AmonRudh AmonRudh Belegost Nogrod KhazadDum Moria 5 Belegost Moria Nogrod AmonRudh Moria AmonRudh Belegost Nogrod Nogrod Belegost Moria AmonRudh AmonRudh Nogrod Moria Belegost KhazadDum 9 Haedelf Freidelf AmonRudh Freidelf Haedelf Weibheim Weibheim AmonRudh Freidelf AmonRudh Haedelf Weibheim Moria Nogrod Belegost KhazadDum EredLuin Nogrod Nogrod KhazadDum Moria EredLuin Belegost KhazadDum Moria KhazadDum Belegost Nogrod Moria	KhazadDum Moria KhazadDum Belegost EredLuin KhazadDum Moria Nogrod

G. ᚦᚢᛟᛟᛗᛚ ᛊᛗᛁᛗᚲᚦᛟᚱ

10 Points

The dwarves have used their sense of smell to identify several natural deposits of ore and water, without excavating them. They used the solution to problem D to determine the distance from each deposit to the ones near it. The dwarves have to build tunnels between the deposits, but the cost in labor is expensive. So, they want to build the least length of tunnels that allows them to travel between deposits. They will give you the distances between deposits, and they want you to output the total minimum distance of tunnels that will connect the deposits.

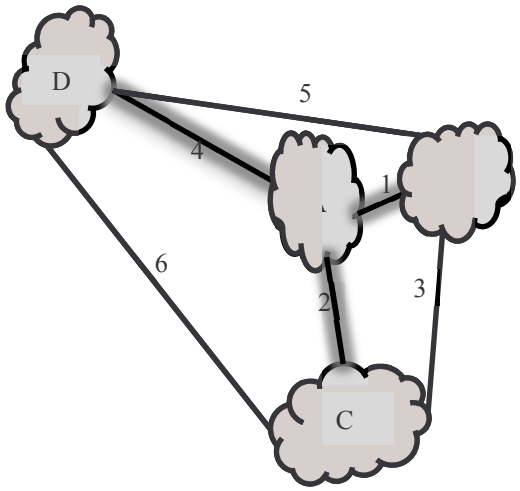
The input consists of a line with a single number $0 < n < 100$ indicating the number of problems. Each problem begins with a line containing a single number $0 < m < 1000$, indicating the number of deposits. This is followed by m–1 lines of data. Each line of data shows the distances between the deposit and the all of the preceding deposits. There is no line for the first deposit. For each distance d, $0 < d \leq 1000$. Dwarvish distances are given in number of armslengths, which are 29 inches.

For example, the data

```
4
1
2 3
4 5 6
```

Represents the distance Matrix

	Deposit A	Deposit B	Deposit C	Deposit D
Deposit A	0	1	2	4
Deposit B	1	0	3	5
Deposit C	2	3	0	6
Deposit D	4	5	6	0



In this case, the shortest tunnel system connects deposits 2, 3 and 4 through deposit 1, so the total distance is 7. The picture is on the right, with the minimum tunnel system highlighted in gray.

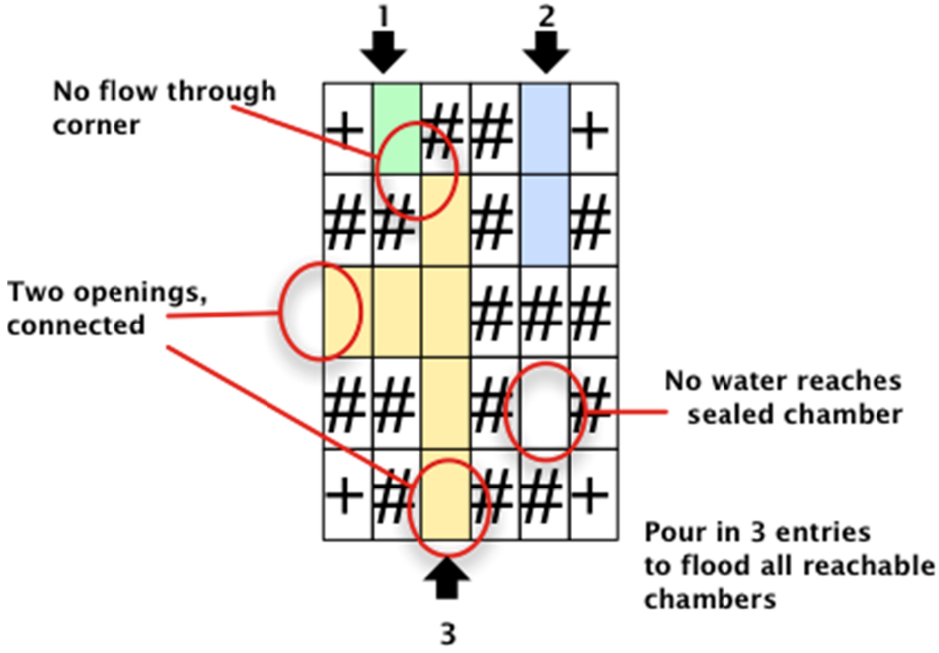
For each problem, you should output a single number on a line by itself which indicates the total distance of the minimal tunnel system connecting all of the deposits.

Sample Input	Sample Output
3 4 2 1 3 3 1 2 5 2 2 2 2 1 2 2 2 1 1 6 10 1 5 5 1 4 2 5 1 4 5 2 4 1 3	4 5 7

H. ᛊᛈᚱᛇᛜᚷ ᚲᛖᛗᚠᛜᛇᛜᚷ

10 Points

Each year in April, the dwarves flood the mines for spring cleaning. Since many of the passageways in the mines are connected, it is enough to pour water in in only a few entries … but how many? Fortunately the dwarves keep extensive maps of each level of the mines, and they need only pour water into the passageways in the top-most level. A map of a level looks like the figure on the right. (Yes, the dwarves still use ASCII art. Their video games are really retro, too.) The maps are always rectangular. The characters "+" and "#" indicate solid rock wall (with "+" in the corners), and blanks indicate open space, including entries to the outside where a blank appears on the edge of the map. If you check carefully, you can see that although there are four entries to the mine system described by this map, it is sufficient to pour water into three of them, because two of them (filled with yellow water below) are connected:



The input begins with an integer describing the number of problems in the remainder of the file. Each problem begins with the dimensions of the map (e.g., 6 rows of 5 characters each, in the example above), followed by an ASCII art map. The corners of the map are indicated with the character "+", and a "+" or "#" character indicates impervious granite, while a blank (" ") indicates an opening or tunnel. Your output for each problem should be a single line saying "*k* entries" where *k* is the minimum number of entries (e.g., *k* would be "3" for the example above).

Sample Input	Sample Output

6
4 9
+####### +
## ## # 	
## ## # 	
+## ##   +
8 8
+# ####+
# # 	
# #### # 	
# # # # 	
# #### # 	
#
# # 	
+#######+
3 6
+ ## +
## # # 	
+ # #+
10 33
+################## ###### #######+
# # # # 	
########### #### ########### ### 	
# # # 	
############# ##### ############# 	
# # # # # 	
            #   #
############ #################### 	
# # 	
+##################################+
10 33
+######################## #######+
# # # # 	
########### #### ########### ### 	
# # # 	
############# ##### ############# 	
# # # # # 	
            #   #   #
############ #################### 	
# # 	
+################## #############+
10 33
+######################## #######+
# # # # 	
########### #### ########### ### 	
# # # 	
###### ####### ##### ############# 	
# # # # # 	
            #   #   #   #
############ ######### ###### ### 	
# # 	
+################## #############+
	2 entries
	1 entries
	4 entries
	3 entries
	4 entries
	1 entries

1. KHAZAD DWARVISH GRAMMAR

12 Points

Khazad is a difficult language for people of Middle Earth, including the Dwarves. They have strict requirements about written language that are so onerous that common Dwarvish, the spoken version, often ignores them. However, sacred texts (bills of lading, for example), must follow these rules of syntax. The High Council of Sacred Scrolls has asked you to write a program that will check written documents to make sure they fit the standards of Khazad. Unlike English, which has seven parts of speech, Khazad has only four: nouns, verbs, adverbs and conjunctions.[1] However, the rules for word order are much more rigid than in English:

1. A sentence consists of a subject and a predicate. The subject must occur first in a sentence, followed by the predicate. The predicate may be intransitive, transitive or ditransitive. The subject is a noun phrase.
2. Nouns phrases (subjects and objects) may be single nouns or compound nouns, meaning that they contain a list of nouns separated by conjunctions. Other things cannot be compound.
3. For intransitive predicates, the verb must occur directly after the subject and may be followed by some adverbs.
4. For transitive predicates, the verb must occur after the subject and the direct object must occur directly after the verb. The direct object may be followed by some adverbs.
5. For ditransitive predicates, the indirect object must occur directly after the subject, followed by the verb, followed by the direct object. The direct object may be followed by some adverbs.
6. Adverbs describing time must occur before other adverbs.

To preserve the confidential nature of the documents, you are not given the originals, but a copy of the document where:

1. Nouns have been replaced by the words "`gimly`", "`tunnel`" or "`gold`" depending on whether they refer to a person, place or thing.
2. Verbs have been replaced with "`counted`" .
3. Adverbs have been replaced with "`now`" or "`glumly`" depending on whether they refer to time or not.
4. Conjunctions have been replaced with "`and`".

The input consists of a line containing a number $0 < n < 1000$ followed by n lines containing one sentence each. The sentences do not contain any punctuation, only a space-separated list of the words given above, each of which appears in all lowercase. Your output for each sentence should be a line containing "`accept`" or "`reject`" depending on whether the sentence is in proper Khazad or not.

Note 1: . Khazad conjunctions are "and", "or", "xor" and "nor". English prepositional phrases are rendered as nouns in Khazad that have diacritcals to make them adjectival or adverbial, so they are treated as verbs or adverbs. As with many human languages, adjectives are considered verbs which are more permanent than active verbs. Verbs are fully inflected as a single word, so they do not use "helping verbs".

Sample Input	Sample Output
17 gimly counted gold glumly glumly gold counted gimly now tunnel now and glumly gimly counted tunnel gimly counted tunnel now glumly gimly and gimly and gimly counted now glumly glumly gold counted gimly gold tunnel counted gimly gold gimly counted gold tunnel gimly and gold counted glumly glumly gimly and gold counted glumly and glumly gimly counted gold and tunnel counted gimly counted and counted and counted glumly gimly counted glumly tunnel here tunnel gold counted gold here tunnel and gold counted gold tunnel gold counted gold tunnel tunnel gold counted gold gold	accept reject reject accept accept accept accept accept accept reject reject reject reject reject reject accept reject

If you prefer, the following is a BNF grammar for this fragment of High Dwarvish:

sentence ::= nouns predicate
predicate ::= verb adverbs | verb nouns adverbs | nouns verb nouns adverbs
nouns ::= noun `and` nouns | noun
noun ::= `gimly` | `tunnel` | `gold`
verb ::= `counted`
adverbs ::= timeadverbs timelessadverbs | timeadverbs | timelessadverbs | ε
timeadverbs ::= `now` timeadverbs | `now`
timelessadverbs ::= `glumly` timelessadverbs | `glumly`