Problem Set for the 3rd Day of Summer Camp 2007

Japanese Alumni Group

Contest Held on 24 Sep 2007

Status of Problems

All problems were newly created by the members of Japanese Alumni Group.

Some portion of the problem statement was modified for clarifications.

The sample input in Problem A was found to contain an illegal dataset. It is corrected in this PDF file.

Terms of Use

Problem A
Election

Summer Camp (Day 3), Tokyo
24 Sep 2007

Input: A.txt

Giselle has just made a vote for a national election. In her country, members of the legislature are elected by a system called mixed member proportional representation (MMP). Basically, half the members are elected from constituencies, and the other half are elected from party lists by proportional representation. Each voter has two votes, one for a constituency representative and one for a party.

In each constituency, the representative is chosen by a single-winner voting system called the first-past-the-post. This system is very simple: the candidate who earns the highest number of votes wins the seat. There are constituencies equal to half the number of seats, and they are determined in accordance with geographical areas.

Each party is allocated the seats according to the percentage of votes cast for that party. Only parties that have either at least five percent of the total party votes or at least three constituency seats are eligible for the seats; the parties that satisfy neither of these prerequisites are excluded on the following procedure. The number of seats for each eligible party is determined based on the value given by:

$$\text{(the number of overall seats)} \times \frac{\text{(the number of votes cast for that party)}}{\text{(the total number of votes cast for all eligible parties)}}.$$

Note that the multiplier in the above formula is the number of *overall* seats, not party-list seats (i.e. not half the members). Each party receives the seats equal to the integer part of this value. There usually remain some seats, and they are allocated to the parties in decreasing order of the fraction parts, where each party receive at most one extra seat. If two or more parties have the same fraction parts, the party that gained a greater number of votes gets higher preference.

The number of seats allocated by the above procedure counts both the constituency seats and the party-list seats. Each party is therefore entitled to add members from the list just as many as the number of its allocated seats minus the number of its constituency seats. Those members are chosen in the order predetermined by the party. If some candidates in the party list already have the seats for constituency representatives (this happens because each constituency candidate is allowed to also be included in the list), they are not counted and the next candidates down are added instead.

The candidates who won in constituencies never forfeit their seats. It sometimes happens that the number of constituencies where a party won exceeds the number of seats allocated for the party vote. In this case, *all* winners in constituencies receive the seats in the legislature, although no more members will be elected from the party list. The same still applies to the candidates in the parties ineligible to be allocated the seats. Note that this raises the total number of seats. The seats added for this reason are called *overhang seats*.

Now, let us take an example. Suppose three parties A, B, and C are competing for eight seats, where the party A has earned one constituency seat and 9,000 party votes, the party B one and 8,000, and the party C two and 3,000. The total number of party votes is $9000 + 8000 + 3000 = 20000$, thus the five-percent threshold is $20000 \times (5/100) = 1000$. From this threshold, all parties are eligible to be allocated the seats. The formula gives $(8 \times 9000)/20000 = 3.6$, $(8 \times 8000)/20000 = 3.2$, and $(8 \times 3000)/20000 = 1.2$, so the parties A, B, and C receive three seats, three, and one respectively. There is one remaining seat, and it goes to the party A for the largest fraction part 0.6 ($= 3.6 - 3$). In conclusion, the party A gains four seats in total, and since this party won one constituency seat, there are three more members to be chosen from the party A's list. Similarly, there are two more members from the party B's list. On the other hand, the party C receives only one seat despite winning in two constituencies. So no members will be chosen from the party C's list and one overhang seat occurs. The total number of elected members therefore will be nine. This example corresponds to the first case of the sample input and output.

You are required to write a program that determines which candidates win the seats.

► Input

The input consists of multiple data sets. Each data set has the following format:

> N M
> $Party_1$
> $Party_2$
> ...
> $Party_M$
> $Constituency_1$
> $Constituency_2$
> ...
> $Constituency_{N/2}$

N is a positive even integer that represents the number of seats. M is a positive integer that represents the number of parties. $Party_i$ is the description of the i-th party. $Constituency_i$ is the description of the i-th constituency.

Each party description is given in the following format:

> $PartyName$ C V
> $Name_1$
> $Name_2$
> ...
> $Name_C$

$PartyName$ is the name of the party. C is a positive integer that represents the number of candidates in the party list. V is a non-negative integer that represents the number of votes cast for that party. $Name_i$ is the name of the candidate with the i-th highest priority in the party list.

Each constituency description is given in the following format:

> C
> $Name_1$ $Party_1$ V_1
> $Name_2$ $Party_2$ V_2
> ...
> $Name_C$ $Party_C$ V_C

C is a positive integer, equal to or greater than two, that represents the number of candidates in the constituency. $Name_i$ is the name of the i-th candidate in the constituency. $Party_i$ is the name of the party that the i-th candidate belongs. V_i is a non-negative integer that represents the number of votes cast for the i-th candidate.

The input is terminated with a line that contains two zeros. This line should not be processed.

You may assume all the followings:

- The name of each party is a string up to ten characters that begins with an uppercase character and consists of only uppercase and numeric characters. The name of each candidate is a string up to twenty characters that begins with a lowercase character and consists of only lowercase and numeric characters. No multiple parties or candidates have the same name.
- The number of parties, the number of seats, and the total number of different candidates do not exceed 20, 200, and 1,000 respectively. Neither the total number of party votes nor the total number of votes in each constituency exceeds 10,000,000.

- No two or more parties receive the same number of party votes. Also, in each constituency, no two or more candidates receive the same number of constituency votes.
- Each party list contains enough candidates, that is, the party can always choose the required number of candidates from the list.
- Every candidate belongs to just one of the parties. No candidate is allowed to compete in more than one constituency. Note that, however, each candidate may appear up to twice in a data set, one in a party list and one in a constituency description.
- The number of data sets in the input does not exceed fifty.

► Output

For each data set, print names of all elected persons, one name per line, in lexicographical order according to the ASCII code. Print an empty line between two consecutive data sets.

► Sample Input

```
8 3
A 6 9000
a1
a2
a3
a4
a5
a6
B 6 8000
b1
b2
b3
b4
b5
b6
C 4 3000
c1
c2
c3
c4
2
a7 A 2000
b2 B 4000
3
a8 A 1500
c3 C 500
b1 B 1000
2
c2 C 2328
a3 A 2327
2
b5 B 2345
c5 C 4000
4 3
A 3 2500
a1
a2
```

```
a3
B 3 1500
b1
b2
b3
C 1 150
c1
2
a4 A 1500
b4 B 1000
2
a5 A 700
b5 B 800
0 0
```

► Output for the Sample Input

```
a1
a2
a3
a8
b1
b2
b3
c2
c5

a1
a2
a4
b5
```

Problem B
Jaggie Spheres

Input: B.txt

Summer Camp (Day 3), Tokyo
24 Sep 2007

Let $J(n)$ be a three-dimensional body that

- is a union of unit cubes whose all vertices lie on integer coordinates,
- contains all points that are closer than the distance of $\sqrt{n}$ to the origin, and
- is the smallest of all such bodies.

The figure below shows how $J(1)$, $J(2)$, and $J(3)$ look.

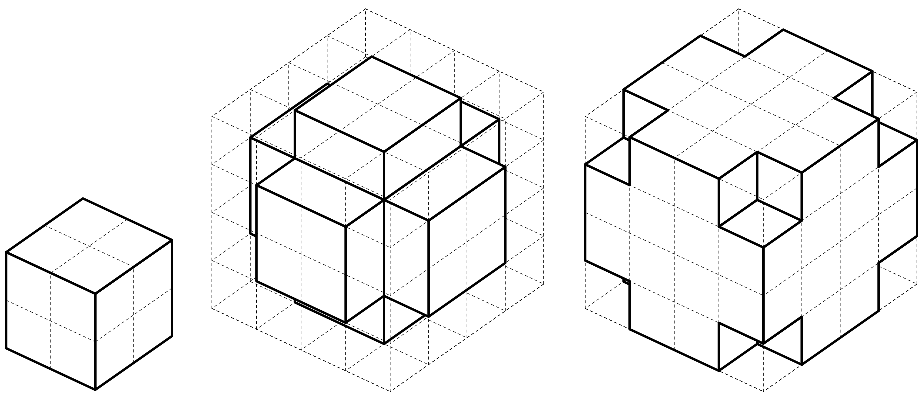

Figure 1: Jaggie Spheres for $n = 1, 2, 3$

Your task is to calculate how many faces $J(n)$ have. Here, we define two square belong to the same face if they are parallel and share an edge, but don't if they share just a vertex.

► Input

The input consists of multiple data sets, each of which comes with a single line containing an integer n ($1 \le n \le 1000000$). The end of input is indicated by $n = 0$.

► Output

For each data set, print the number of faces $J(n)$ have.

► Sample Input

```
1
2
3
4
0
```

► Output for the Sample Input

```
6
30
30
6
```

Problem C
Nagashi Soumen

Input: C.txt

Summer Camp (Day 3), Tokyo
24 Sep 2007

Mr. Natsume, the captain of a baseball club, decided to hold a *nagashi-soumen* party. He first planned to put a flume at the arena, and let members stand by the side of the flume to eat soumen. But they are so weird that they adhere to each special position, and refused to move to the side of the flume. They requested Mr. Natsume to have the flumes go through their special positions. As they never changed their minds easily, Mr. Natsume tried to rearrange flumes in order to fulfill their requests.

As a caretaker of the baseball club, you are to help Mr. Natsume arrange the flumes. Here are the rules on the arrangement:

- each flume can begin and end at arbitrary points. Also it can be curved at any point in any direction, but cannot have any branches nor merge with another flume;
- each flume needs to be placed so that its height strictly decreases as it goes, otherwise soumen does not flow;
- Mr. Natsume cannot make more than K flumes since he has only K machines for water-sliders; and
- needless to say, flumes must go through all the special positions so that every member can eat soumen.

In addition, to save the cost, you want to arrange flumes so that the total length is as small as possible. What is the minimum total length of all flumes?

▶ Input

Input consists of multiple data sets. Each data set follows the format below:

N K
x_1 y_1 z_1
$\cdots$
x_N y_N z_N

N ($1 \le N \le 100$) represents the number of attendants, K ($1 \le K \le 4$) represents the number of available flumes, and x_i, y_i, z_i ($-100 \le x_i, y_i, z_i \le 100$) represent the i-th attendant's coordinates. All numbers are integers. The attendants' coordinates are all different.

A line with two zeros represents the end of input.

▶ Output

For each data set, output in one line the minimum total length of all flumes in Euclidean distance. No absolute error in your answer may exceed 10^{-9}. Output -1 if it is impossible to arrange flumes.

► Sample Input

```
1 1
0 0 0
4 4
0 0 0
1 1 1
2 2 2
3 3 3
3 4
0 0 0
1 1 1
2 2 2
4 1
1 0 1
0 0 0
1 1 0
0 1 -1
5 2
0 0 100
0 0  99
1 0  99
1 0  98
1 0 -100
7 4
71 55 -77
-43 -49 50
73 -22 -89
32 99 -33
64 -22 -25
-76 -1 6
39 4 23
0 0
```

► Output for the Sample Input

```
0
0
0
-1
200
197.671366737338417
```

Problem D
Private Teacher

Input: D.txt

Summer Camp (Day 3), Tokyo
24 Sep 2007

You are working as a private teacher. Since you are giving lessons to many pupils, you are very busy, especially during examination seasons. This season is no exception in that regard.

You know days of the week convenient for each pupil, and also know how many lessons you have to give to him or her. You can give just one lesson only to one pupil on each day. Now, there are only a limited number of weeks left until the end of the examination. Can you finish all needed lessons?

► Input

The input consists of multiple data sets. Each data set is given in the following format:

N W
t_1 c_1
list-of-days-of-the-week
t_2 c_2
list-of-days-of-the-week
...
t_N c_N
list-of-days-of-the-week

A data set begins with a line containing two integers N ($0 < N \leq 100$) and W ($0 < W \leq 10^{10}$). N is the number of pupils. W is the number of remaining weeks. The following $2N$ lines describe information about the pupils. The information for each pupil is given in two lines. The first line contains two integers t_i and c_i. t_i ($0 < t_i \leq 10^{10}$) is the number of lessons that the i-th pupil needs. c_i ($0 < c_i \leq 7$) is the number of days of the week convenient for the i-th pupil. The second line is the list of c_i convenient days of the week (Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, and Saturday).

The end of input is indicated by a line that contains two zeros. This line should not be processed.

► Output

For each data set, print "`Yes`" if you can finish all lessons, or "`No`" otherwise.

► Sample Input

```
2 2
6 3
Monday Tuesday Wednesday
8 4
Thursday Friday Saturday Sunday
2 2
7 3
Monday Tuesday Wednesday
9 4
Thursday Friday Saturday Sunday
0 0
```

► Output for the Sample Input

```
Yes
No
```

Problem E
Triangles

Input: E.txt

Summer Camp (Day 3), Tokyo
24 Sep 2007

There is a group that paints an emblem on the ground to invite aliens every year. You are a member of this group and you have to paint the emblem this year.

The shape of the emblem is described as follows. It is made of n regular triangles whose sides are equally one unit length long. These triangles are placed so that their centroids coincide, and each of them is rotated counterclockwise by $360/n$ degrees with respect to the one over it around its centroid. The direction of the top triangle is not taken into account.

It is believed that emblems have more impact as their n are taken larger. So you want to paint the emblem of n as large as possible, but you don't have so much chemicals to paint. Your task is to write a program which outputs the area of the emblem for given n so that you can estimate the amount of the needed chemicals.

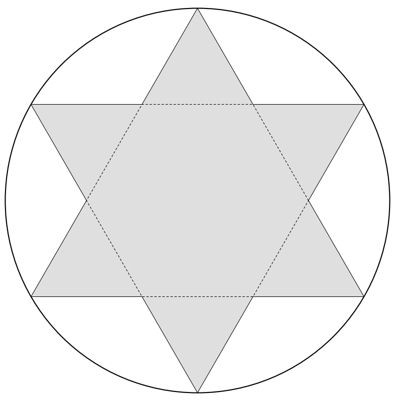
Figure 2: The Emblem for $n = 2$

► Input

The input data is made of a number of data sets. Each data set is made of one line which contains an integer n between 1 and 1000 inclusive.

The input is terminated by a line with $n = 0$. This line should not be processed.

► Output

For each data set, output the area of the emblem on one line. Your program may output an arbitrary number of digits after the decimal point. However, the error should be 10^{-6}(= 0.000001) or less.

► Sample Input

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

► Output for the Sample Input

```
0.433013
0.577350
0.433013
0.732051
0.776798
0.577350
```

Problem F
Two-finger Programming

Input: F.txt

Summer Camp (Day 3), Tokyo
24 Sep 2007

Franklin Jenkins is a programmer, but he isn't good at typing keyboards. So he always uses only '`f`' and '`j`' (keys on the home positions of index fingers) for variable names. He insists two characters are enough to write programs, but naturally his friends don't agree with him: they say it makes programs too long.

He asked you to show any program can be converted into not so long one with the *f-j names*. Your task is to answer the shortest length of the program after all variables are renamed into the f-j names, for each given program.

Given programs are written in the language specified below.

A block (a region enclosed by curly braces: '`{`' and '`}`') forms a scope for variables. Each scope can have zero or more scopes inside. The whole program also forms the global scope, which contains all scopes on the top level (i.e., out of any blocks).

Each variable belongs to the innermost scope which includes its declaration, and is valid from the declaration to the end of the scope. Variables are never redeclared while they are valid, but may be declared again once they have become invalid. The variables that belong to different scopes are regarded as different even if they have the same name. Undeclared or other invalid names never appear in expressions.

The tokens are defined by the rules shown below. All whitespace characters (spaces, tabs, linefeeds, and carrige-returns) before, between, and after tokens are ignored.

token:
  identifier | *keyword* | *number* | *operator* | *punctuation*

identifier:
  `[a-z]+` (one or more lowercase letters)

keyword:
  `[A-Z]+` (one or more uppercase letters)

number:
  `[0-9]+` (one or more digits)

operator:
  '`=`' | '`+`' | '`-`' | '`*`' | '`/`' | '`&`' | '`|`' | '`^`' | '`<`' | '`>`'

punctuation:
  '`{`' | '`}`' | '`(`' | '`)`' | '`;`'

The syntax of the language is defined as follows.

program:
  statement-list

statement-list:
  statement
  statement-list statement

statement:
  variable-declaration
  if-statement
  while-statement
  expression-statement

block-statement

variable-declaration:
  "`VAR`" *identifier* '`;`'

if-statement:
  "`IF`" '`(`' *expression* '`)`' *block-statement*

while-statement:
  "`WHILE`" '`(`' *expression* '`)`' *block-statement*

expression-statement:
  expression '`;`'

block-statement:
  '`{`' *statement-list*$_{optional}$ '`}`'

expression:
  identifier
  number
  expression operator expression

► Input

The input consists of multiple data sets.

Each data set gives a well-formed program according to the specification above. The first line contains a positive integer N, which indicates the number of lines in the program. The following N lines contain the program body.

There are no extra characters between two adjacent data sets.

A single line containing a zero indicates the end of the input, and you must not process this line as part of any data set.

Each line contains up to 255 characters, and all variable names are equal to or less than 16 characters long. No program contains more than 1000 variables nor 100 scopes (including the global scope).

► Output

For each data set, output in one line the minimum program length after all the variables are renamed into the f-j name. Whitespace characters must not counted for the program length. You are not allowed to perform any operations on the programs other than renaming the variables.

► Sample Input

```
14
VAR ga;
VAR gb;
IF(ga > gb) {
  VAR saa;
  VAR sab;
  IF(saa > ga) {
    saa = sab;
  }
}
WHILE(gb > ga) {
  VAR sba;
  sba = ga;
  ga = gb;
}
7
VAR thisisthelongest;
IF(thisisthelongest / 0) {
  VARone;
  one = thisisthelongest + 1234567890123456789012345;
}
VAR another;
32 = another;
0
```

► Output for the Sample Input

```
73
59
```

The two programs should be translated as follows:

```
VAR f;
VAR ff;
IF(f > ff) {
  VAR j;
  VAR fj;
  IF(j > f) {
    j = fj;
  }
}
WHILE(ff > f) {
  VAR j;
  j = f;
  f = ff;
}

VAR j;
IF(j / 0) {
  VARf;
  f = j + 1234567890123456789012345;
}
VAR f;
32 = f;
```

Problem G
Walk under a Scorching Sun

Input: G.txt

Summer Camp (Day 3), Tokyo
24 Sep 2007

Edward R. Nelson is visiting a strange town for some task today. This city is built on a two-dimensional flat ground with several vertical buildings each of which forms a convex polygon when seen from above on it (that is to say, the buidlings are convex polygon columns). Edward needs to move from the point S to the point T by walking along the roads on the ground. Since today the sun is scorching and it is very hot, he wants to walk in the sunshine as less as possible.

Your task is to write program that outputs the route whose length in the sunshine is the shortest, given the information of buildings, roads and the direction of the sun.

► Input

The input consists of multiple data sets. Each data set is given in the following format:

N M
NV_1 H_1 $X_{1,1}$ $Y_{1,1}$ $X_{1,2}$ $Y_{1,2}$ $\ldots$ X_{1,NV_1} Y_{1,NV_1}
$\ldots$
NV_N H_N $X_{N,1}$ $Y_{N,1}$ $X_{N,2}$ $Y_{N,2}$ $\ldots$ X_{N,NV_N} Y_{N,NV_N}
$RX_{1,1}$ $RY_{1,1}$ $RX_{1,2}$ $RY_{1,2}$
$\ldots$
$RX_{M,1}$ $RY_{M,1}$ $RX_{M,2}$ $RY_{M,2}$
θ φ
SX SY TX TY

All numbers are integers. N is the number of the buildings ($1 \le N \le 15$). M is the number of the roads ($1 \le M \le 15$). NV_i is the number of the vertices of the base of the i-th building ($3 \le NV_i \le 10$). H_i is the height of the i-th building. $X_{i,j}$ and $Y_{i,j}$ are the (x, y)-coordinates of the j-th vertex of the base of the i-th building. $RX_{k,\cdot}$ and $RY_{k,\cdot}$ are the coordinates of the endpoints of the k-th roads. θ is the direction of the sun, which is given by a counterclockwise degree which starts in the positive direction of x ($0 \le \theta < 360$). φ is the elevation angle of the sun ($0 < \varphi < 90$). (SX, SY) and (TX, TY) are the coordinates of the points S and T, respectively. All coordinates range from 0 to 1000.

The end of the input is represented by a line with $N = M = 0$. This line should not be processed.

It is guaranteed that both the points S and T are on the roads. No pair of an edge of a shade and a road is parallel. You should assume that the sun is located infinitely far away and that it doesn't move while Edward is walking.

► Output

For each data set, output in one line the length for which he has to walk in the sunshine at least. Your program may output an arbitrary number of digits after the decimal point. However, the error should be 0.001 or less.

► Sample Input

```
1 1
4 10 0 0 10 0 10 10 0 10
-5 -20 30 15
135 45
0 -15 15 0
0 0
```

► Output for the Sample Input

```
11.213
```