Problem Set for the 4th Day of Summer Camp 2008

Japanese Alumni Group

Contest Held on 15 Sep 2008

Status of Problems

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

Some portion of the problem statement may modified from the actual practice contest for correction and/or clarification.

Terms of Use

Problem A
Girls' Party

Input: A.txt

Summer Camp (Day 4), Tokyo
15 Sep 2008

Issac H. Ives hosted a party for girls. He had some nice goods and wanted to distribute them to the girls as the presents. However, there were not enough number of presents and thus he needed to decide who would take them. He held a game for that purpose.

Before the game, Issac had all the girls divided into two teams: he named his close friends Bella and Gabriella as two leaders and asked other girls to join either Bella or Gabriella. After the two teams were formed, Issac asked the girls to form one big circle.

The rule of the game was as follows. The game consisted of a number of rounds. In each round, the girls called numbers from 1 to N in clockwise order (N was a number fixed before the game started). The girl calling the number N was told to get out of the circle and excluded from the rest of the game. Then the next round was started from the next girl, that is, the girls called the numbers again, and the one calling N left the circle. This was repeated until only the members of either team remained. The remaining team won the game.

As the game went on, Bella found far more girls of her team excluded than those of Gabriella's team. Bella complained it, and requested Issac to make the next round begin with a call of zero instead of one. Issac liked her idea as he was a computer scientist, and accepted her request. After that round, many girls of Gabriella's team came to leave the circle, and eventually Bella's team won the game and got the presents from Issac.

Now let's consider the following situation. There are two teams led by Bella and Gabriella respectively, where they does not necessarily have the same numbers of members. Bella is allowed to change the starting number from one to zero at up to one round (regardless the starting girl belongs to her team or not). We want to know how many girls of Bella's team at most can remain in the circle. You are requested to write a program for it.

► Input

The input is a sequence of datasets. The first line of the input contains the number of datasets. The number of datasets does not exceed 200.

Each dataset consists of a line with a positive integer N ($1 \leq N \leq 2^{30}$) and a string that specifies the clockwise order of the girls. Each character of the string is either '`B`' (that denotes a member of Bella's team) or '`G`' (Gabriella's team). The first round begins with the girl indicated by the first character of the string. The length of the string is between 2 and 200 inclusive.

► Output

For each dataset, print in a line the maximum possible number of the girls of Bella's team remaining in the circle, or "`0`" (without quotes) if there are no ways for Bella's team to win the game.

► Sample Input

```
6
1 GB
3 GBGBBB
9 BBBGBBBGGG
9 GGBGBBGBBB
7 GBGGGGBGGG
3 BBBBBGBBBB
```

► Output for the Sample Input

```
1
4
4
1
0
8
```

Problem B Bitwise Kingdom Input: B.txt	*Summer Camp (Day 4), Tokyo* *15 Sep 2008*

In the Bitwise Kingdom, located somewhere in the universe, there are exactly 2^N citizens living and each of them has a unique identification string that represents his or her class in the society. An identification string is a binary string of length N which consists of characters '0' or '1'. The order of classes is defined among the citizens by the following criteria:

1. Citizens identified by a string containing a greater number of ones are ranked higher. For example, "011" indicates a higher class than "100".
2. Among those who have identification strings with the same number of ones, citizens identified by a lexicographically greater identification string are ranked higher. For example, "110" indicates a higher class than "101".

For example, if $N = 3$, there are 8 ($= 2^3$) people in the country, and their identification strings are "000", "001", "010", "100", "011", "101", "110", and "111" (from the lowest class to the highest).

You are given two numbers N ($1 \le N \le 60$) and M ($1 \le M \le 2^N$), and you want to resolve the identification string of the person of the M-th lowest class among 2^N citizens. Can you write a program to solve this problem?

► Input

The input consists of multiple datasets.

Each dataset consists of a line which contains two integers N and M in this order, separated with a single space. The input does not contain any other extra characters such as leading or trailing spaces.

The end of input is indicated by a line with two zeros. This line is not part of any datasets.

► Output

For each dataset, print the identification string of the person of the M-th lowest class in one line. Your program may not omit any leading zeros in the answer.

► Sample Input

```
3 3
3 5
0 0
```

► Output for the Sample Input

```
010
011
```

Problem C
Adaptive Time Slicing Quantization

Summer Camp (Day 4), Tokyo
15 Sep 2008

Input: C.txt

Nathan O. Davis is a student at the department of integrated systems. Today he learned digital quantization in a class. It is a process that approximates analog data (e.g. electrical pressure) by a finite set of discrete values or integers.

He had an assignment to write a program that quantizes the sequence of real numbers each representing the voltage measured at a time step by the voltmeter. Since it was not fun for him to implement normal quantizer, he invented a new quantization method named Adaptive Time Slicing Quantization. This quantization is done by the following steps:

1. Divide the given sequence of real numbers into arbitrary M consecutive subsequences called frames. They do not have to be of the same size, but each frame must contain at least two elements. The later steps are performed independently for each frame.
2. Find the maximum value $V_{\max}$ and the minimum value $V_{\min}$ of the frame.
3. Define the set of quantized values. The set contains 2^L equally spaced values of the interval $[V_{\min}, V_{\max}]$ including the both boundaries. Here, L is a given parameter called a *quantization level*. In other words, the i-th quantized value q_i ($1 \le i \le 2^L$) is given by:
$$q_i = V_{\min} + (i-1) \cdot \frac{V_{\max} - V_{\min}}{2^L - 1}.$$
4. Round the value of each element of the frame to the closest quantized value.

The key of this method is that we can obtain a better result as choosing more appropriate set of frames in the step 1. The quality of a quantization is measured by the sum of the squares of the quantization errors over all elements of the sequence: the less is the better. The quantization error of each element is the absolute difference between the original and quantized values.

Unfortunately, Nathan caught a bad cold before he started writing the program and he is still down in his bed. So he needs you help. Your task is to implement Adaptive Time Slicing Quantization instead. In your program, the quantization should be performed with the best quality, that is, in such a way the sum of square quantization errors is minimized.

► Input

The input consists of multiple datasets. Each dataset contains two lines. The first line contains three integers N ($2 \le N \le 256$), M ($1 \le M \le N/2$), and L ($1 \le L \le 8$), which represent the number of elements in the sequence, the number of frames, and the quantization level. The second line contains N real numbers ranging in $[0, 1]$. The input is terminated by the dataset with $N = M = L = 0$, which must not be processed.

► Output

For each dataset, output the minimum sum of square quantization errors in one line. The answer with an absolute error of less than or equal to 10^{-6} is considered to be correct.

► Sample Input

```
5 2 1
0.1 0.2 0.3 0.4 0.5
6 2 2
0.1 0.2 0.3 0.4 0.5 0.6
0 0 0
```

► Output for the Sample Input

```
0.01
0.00
```

Problem D
Reaction

Input: D.txt

Summer Camp (Day 4), Tokyo
15 Sep 2008

You are a hero of a role playing game, asked by the king to defeat monsters threating people's life.

You have been making a long journey with your colleagues, and you are now in the town closest to the final dungeon where the head of monsters dwells. You have heard from people that the head monster hits with his strong arms, casts powerful spells, and has many special abilities, and your party would be easily killed off without powerful equipments due to severe damages. Thus you need to prepare the equipments.

On the other hand, you have a number of magical spheres collected during the journey. Those spheres are not useful themselves, but they can be turned into special items by a spell of reaction. You can obtain some amount of money by selling those special items to shops in the town, then buy the equipments by that money.

The spell of reaction works as follows. Each sphere has a color, and either positive attribute or negative attribute. You choose one sphere with positive attribute and another with negative attribute, and you cast the spell to the two spheres. Then the spheres will make reaction to have a set of special items produced. Those spheres will disappear after the reaction. The set of items you will obtain solely depends on the colors of the two spheres. You can cast the spell as many as you want, but of course you cannot cast the spell to spheres that have disappeared. Also, not all pairs of colors of spheres make reaction.

It is natural that you want to obtain money as much as possible. So you should choose carefully the pairs of spheres before casting the spell. On the other hand, you should be an excellent programmer, so it should be an easy task to write a program that finds the best way using your computer.

Your task is now clear — write a program and get ready for the battle with the head monster!

▶ Input

The input is a sequence of datasets. Each dataset is formatted as follows:

> $N^- \ N^+$
> *Number_of_available_spheres*
> *Definition_of_items*
> *Definition_of_reactions*

The first line contains two integers N^- and N^+, which are the numbers of different colors of spheres with negative and positive attributes, respectively. The rest of the dataset is divided into three parts.

The first part describes the number of available spheres. This part has the following format:

$$K_1^- \ K_2^- \ \ldots \ K_{N^-}^-$$
$$K_1^+ \ K_2^+ \ \ldots \ K_{N^+}^+$$

K_i^- is the number of spheres of the i-th color with negative attribute, and K_i^+ is the number of spheres of the i-th color with positive attribute.

The second part contains the definition of items. This part is formatted as follows:

> M
> $A_1 \ P_1$
> $\ldots$
> $A_M \ P_M$

Here, M is the number of items that can be produced. Each of the following M lines contains a string A_i and an integer P_i, which are the name and the selling price of the i-th item respectively.

The last part gives the details of reactions. This part has the following format:

$$
\begin{array}{l}
L \\
I_1^- \ I_1^+ \ NJ_1 \ J_{1,1} \ \ldots \ J_{1,NJ_1} \\
\ldots \\
I_L^- \ I_L^+ \ NJ_L \ J_{L,1} \ \ldots \ J_{L,NJ_L}
\end{array}
$$

The first line contains an integer L, which is the number of pairs of colors of spheres that can make reaction. Each of the next L lines starts with two integers I_i^- and I_i^+, which denote the colors of negative and positive spheres respectively. The next integer NJ_i is the number of items produced by reaction between spheres I_i^- and I_i^+. The line is then followed by NJ_i strings, each of which is an item name.

You may assume all the following: $1 \le N^-, N^+ \le 100$; $1 \le K_i^-, K_i^+ \le 100$; $1 \le M \le 100$; $1 \le P_i \le 100$; $1 \le L \le 100$; $1 \le NJ_i \le 10$. You may also assume that an item name consists only of alphanumeric characters and the length of the name does not exceed ten.

The end of input is represented by a line with two zeros. This line is not part of any dataset.

► Output

For each dataset, print a line that contains the maximum possible total selling price.

► Sample Input

```
2 2
1 1
1 1
4
A 10
B 20
C 30
D 40
4
1 1 3 A A A
1 2 2 B C
2 1 1 D
2 2 3 A A B
2 2
1 2
2 1
3
Scroll 50
Bastard 100
Heal100 10
3
1 1 1 Scroll
2 1 1 Bastard
2 2 1 Heal100
0 0
```

► Output for the Sample Input

```
90
200
```

Problem E
Magical Island

Input: E.txt

Summer Camp (Day 4), Tokyo
15 Sep 2008

This is a story in the epoch of magic. A clan of magicians lived in an artificial island built by magic power.

One day, a crisis erupted on the island. An Empire ACM (Atlas Country of Magic) required unconditional surrender to them, otherwise an imperial force attacked by magical missiles to the island. However, they were so proud that they did not surrender to the ACM, and built a system to generate magical shield to protect the clan from the threat of magical missiles. In this system, a crystal with different elements was put on each corner of the island: the world consisted of four elements, namely Fire, Water, Air and Earth. Each crystal generated magical shield with the element of the crystal by receiving magicians' magic power; it shielded the island from magical missiles of the same element: any magical missile consists of one of the four elements. Magic shield covered a circular area; the crystal should be located on the center the circular area. The crystal required R^2 magic power to shield a circular area of radius R. However, there was one restriction. Magicians should send exactly the same amount of magic power to all crystals, otherwise the island was lost because of losing a balance between four elements.

They decided to live in an area which is shielded from any magical missile. Your job is to write a program to calculate minimum amount of magic power to secure enough area for them to live in.

► Input

The input consists of multiple datasets. Each dataset is a single line containing three integers W, H and S, separated by a single space. The line containing three zeros separated by a single space indicates the end of the input.

W and H are width and depth of the island, respectively. S is the area magicians needed to live in. You may assume that $0 < W, H \leq 100$ and $0 < S \leq W \times H$.

► Output

For each dataset, output a separate line containing the total minimum necessary magic power. The value may contain an error less than or equal to 0.001. You may print any number of digits after the decimal point.

► Sample Input

```
1 1 1
10 15 5
15 10 100
0 0 0
```

► Output for the Sample Input

```
8.000
409.479
861.420
```

Problem F
Ninja Legend

Input: F.txt

Summer Camp (Day 4), Tokyo
15 Sep 2008

Ninjas are professional spies in the Middle Age of Japan. They have been popular in movies and games as they have been described with extraordinary physical abilities and unrealistic abilities.

You are absorbed in one of ninja games, *Ninja Legend*. In this game, you control in an exceptionally talented ninja named *Master Ninja* and accomplish various missions. In one of the missions, the ninja intrudes into a mansion and aims to steal as many gold pieces as possible. It is not easy to be done because there are many pitfalls to evade him. Once he falls into a pitfall, all he can do will be just to wait for being caught by the owner of this mansion. This definitely means failure of the mission. Thus, you must control the ninja with a good strategy.

The inside of mansion is represented by a grid map as illustrated below. Master Ninja enters at and exits from the entrance. He can move in the four directions to the floor cells. He can pick up gold blocks when he moves to the cells where they are placed, but it is also allowed to pick up them in the later visit of those cells or even to give up them.

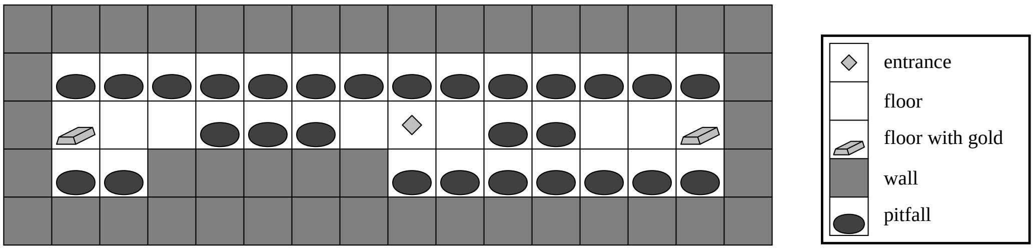

Figure 1: Example Map

The ninja has a couple of special abilities for dealing with the pitfalls. He has two moving modes: *normal mode* and *dash mode*. He is initially in the normal mode. He gets into the dash mode when he moves in the same direction by two cells in a row without picking up gold blocks. He gets back to the normal mode when he changes the direction, picks up a gold block, or runs on a wall as mentioned later. He is able to jump over a pitfall, or in the dash mode he can jump over two pitfalls in his dashing direction. In addition, when he is in the dash mode, he can *run on a wall* to pass over up to four pitfalls in his dashing direction. For his running, there must be consecutive wall cells adjacent to the passing cells including the departure and arrival floor cells. Note that he gets back to the normal mode when he runs on a wall as mentioned previously.

In the figure below, the left one shows how the ninja runs on a wall, and the right one illustrates a case in which he cannot skip over pitfalls by running on a wall.

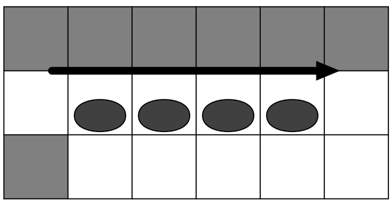

Figure 2: Running on a Wall

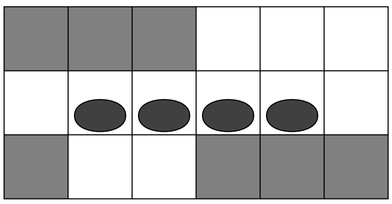

Figure 3: Non-consecutive Walls

You want to know the maximum number of gold blocks the ninja can get, and the minimum cost to get those gold blocks. So you have decided to write a program for it as a programmer. Here, move of the ninja from a cell to its adjacent cell is considered to take one unit of cost.

▶ Input

The input consists of multiple datasets.

The first line of each dataset contains two integers H ($3 \leq H \leq 60$) and W ($3 \leq W \leq 80$). H and W indicate the height and width of the mansion. The following H lines represent the map of the mansion. Each of these lines consists of W characters. Each character is one of the following: '`%`' (entrance), '`#`' (wall), '`.`' (floor), '`^`' (pitfall), '`*`' (floor with a gold block).

You can assume that the number of gold blocks are less than or equal to 15 and every map is surrounded by wall cells.

The end of the input indicated with a line containing two zeros.

▶ Output

For each dataset, output the maximum number of gold blocks and the minimum cost in one line. Output two zeros instead if the ninja cannot get any gold blocks. No other characters should be contained in the output.

▶ Sample Input

```
6 23
#######################
#%.^.^#################
#^^^^^######^^^^^^^^^^#
#^^^...^^^^...^^*^^..*#
#^^^^^^^^^^^^^^^^^^^^^#
#######################
5 16
################
#^^^^^^^^^^^^^^#
#*..^^^.%.^^..*#
#^^#####^^^^^^^#
################
0 0
```

▶ Output for the Sample Input

```
1 44
2 28
```

Problem G
Do It

Input: G.txt

Summer Camp (Day 4), Tokyo
15 Sep 2008

There is sometimes difficulty giving nice backgrounds to contest problems. So let this problem kept this simple — write a program that calculates the value of the following formula where the parameters N, R, A_i, ω_i, and φ_i are all given:

$$\int_0^R \left[\prod_{i=1}^{N} A_i \sin\left(\omega_i t + \varphi_i \cdot \frac{\pi}{180}\right) \right] dt$$

► Input

The input consists of a series of datasets each of which has the following format:

N R
A_1 ω_1 φ_1
A_2 ω_2 φ_2
...
A_N ω_N φ_N

The numbers in the input are all integers and satisfy the following conditions: $1 \le N \le 16$, $1 \le R \le 10^4$, $1 \le A_i \le 10$, $0 \le \omega_i \le 100$ and $0 \le \varphi_i \le 360$.

The end of input is indicated by $N = R = 0$. This is not part of any dataset and hence should not be processed.

► Output

For each dataset, print the result of the integration in a line. The result may be printed with an arbitrary number of fractional digits, but should have a relative or absolute error not greater than 10^{-8}.

► Sample Input

```
1 1
1 1 0
0 0
```

► Output for the Sample Input

```
0.45969769413186
```

Problem H
Robot Communication

Input: H.txt

Summer Camp (Day 4), Tokyo
15 Sep 2008

In the year 21xx, human beings are proliferating across the galaxy. Since the end of the last century, thousands of pioneer spaceships have been launched in order to discover new habitation planets.

The Presitener is one of those spaceships, heading toward the Andromeda galaxy. After a long, long cruise in the hyperspace, the crew have finally found a very hopeful candidate planet. The next thing to do is to investigate the planet whether it is really suitable for a new resident or not.

For that purpose, the ship is taking some unattended landers. The captain Juclean Dripac decided to drop them to the planet and collect data about it. But unfortunately, these robots are a bit old and not so clever that the operator has to program what to do on the planet beforehand of the landing. Many staffs including you are called for making this important plan.

The most complicated phase in the mission is to gather and integrate all the data collected independently by many robots. The robots need to establish all-to-all communication channels to exchange the data, once during the mission. That is, all the robots activate their communication channels all together at the predetermined time, and they exchange the data with each other at that time.

They use wireless channels to communicate with each other, so the distance between robots does not limit the connectivity. But the farther two robots goes, the more power they have to use for communication. Due to the limitation of the battery capacity, you want to save the transmission power as much as possible.

For a good thing, communication units of the robots also have the routing functionality, each robot only has to talk with the nearest robot. Suppose a graph whose vertices represent robots and edges represent communication channels established between them. If the graph is connected, all-to-all communication can be established.

Your task is to write the program to calculate the minimum total transmission power required for all-to-all communication among the robots. Each robot moves linearly on the planet surface. Each pair of robots which communicate each other must occupy one channel, but you can assume enough number of channels are available. The transmission power required for two robots is proportional to the distance between them, so the cost here is exactly the sum of the distances between each pair of robots which establish a communication channel.

You may also regard the planet surface as a two-dimensional surface, as it is huge enough. The time required for communicating data among robots are also negligible.

► Input

The input contains multiple datasets. Each dataset has the format below.

N T
x_1 y_1 vx_1 vy_1
...
x_N y_N vx_N vy_N

The first line of each dataset contains two integers; N is the number of robots used for gathering data ($2 \le N \le 16$), and T is the time limit of the mission ($1 \le T < 1000$).

Each of the following N lines describes the motion of a robot. (x_i, y_i) and (vx_i, vy_i) are the initial landing position and the velocity of the i-th robot, respectively ($|x_i|, |y_i| < 100000$, $|vx_i|, |vy_i| < 1000$).

The last dataset is followed by a line containing two zeros. This line is not a part of any dataset and should not be processed.

► Output

For each dataset, output in a line the minimum communication cost required for all-to-all communication. Your program may output an arbitrary number of digits after the decimal point. The absolute error should be less than or equal to 0.001.

► Sample Input

```
4 2
2 0 0 1
0 4 1 0
4 6 0 -1
6 2 -1 0
4 6
2 0 0 1
0 4 1 0
4 6 0 -1
6 2 -1 0
0 0
```

► Output for the Sample Input

```
6.00000000
4.24264069
```