IDI Open Programming Contest April 14th, 2018

Problem Set

A Boat Parts (Easy)
B Cross Country
C Cuboid Slicing Game
D Draft Time
E Gear Changing (Easy)
F Gruesome Cave
G Number Anagrams
H Picture Day
I Polygon Game
J Secret Santa Cycles
K When Planetoids Align
L Witchwood

Jury and Problem Writers

Torbjørn Morland (Head Judge)
Karl Johan Heimark
Jean Niklas L'orange
Ruben Spaans

Tips

- Tear the problem set apart and share the problems among you.
- Problems are not ordered by difficulty.
- Try solving the easy problems first. Two problems in this set are tagged with "(Easy)" to help point you in the right direction.
- If your solution fails on a problem, you can print your program and debug it on paper while you let someone else work on a different problem on the computer.
- If you need help, contact the judges.
- Look at the scoreboard if you are unsure which problem to work on next.

Rules

- Each team consists of one to three contestants.
- One computer is used per team.
- You may not cooperate with persons not on your team.
- You may print your programs on paper to debug them.
- What you may bring to the contest:
  - Any written material (Books, manuals, handwritten notes, printed notes, etc).
  - Pens, pencils, blank paper, stapler and other useful non-electronic office equipment.
  - NO material in electronic form (CDs, USB pen and so on).
  - NO electronic devices (PDAs and so on).
- The only electronic content you may consult during the content is that specified by the organiser (see the web-page). You may not copy source code from web pages, etc.
- Your programs should read from standard in and write to standard out. Writing to standard error will result in a failed submission. C programs should return 0 from `main()`.
- Your programs may not:
  - access the network,
  - read or write files on the system,
  - talk to other processes,
  - fork,
  - or similar stuff.
  - If you try, your program will hang or crash. If it hangs, it will take a couple of minutes before others will be able to run their programs. So please make an effort to not crack/break what we have spent our spare time preparing for you.
- Show common sense and good sportsmanship.

Boat Parts

Problem ID: boatparts

Photo by Can Mustafa Ozdemir

Boating season is over for this year, and Theseus has parked his boat on land. Of course, the boat looks nothing like it did as of the beginning of the season; it never does. You see, Theseus is constantly looking for ways to improve his boat.

At every day of the boating season, Theseus bought exactly one type of item at his local supply store, and replaced the existing part on his boat with it. Now, as the season has ended, Theseus wonders what day he replaced all the parts from the previous season.

Input

The first line of the input consists of two space-separated integers P and N, representing the number of parts the boat consists of, and the number of days in the boating season respectively. Then follows N lines, each line has a single word w_i, the type of boat part that Theseus bought on day i.

Output

Output the day Theseus ended up replacing the last existing part from the previous season, or `paradox avoided` if Theseus never ended up replacing all the different parts.

Limits

- $1 \leq P \leq N \leq 1\,000$.
- Each word w_i will consist only of the letters a–z and _ (underscore).
- Each word w_i will be between 1 and 20 characters long.
- The number of distinct w_is will be at most P.

Sample Input 1

```
3 5
left_oar
right_oar
left_oar
hull
right_oar
```

Sample Output 1

```
4
```

Sample Input 2

```
4 5
motor
hull
left_oar
hull
motor
```

Sample Output 2

```
paradox avoided
```

Cross Country

Problem ID: crosscountry

Charles Johnson is on his way to meet his very good friend Bernard Terrell. They are going on their bi-weekly skiing trip. Today Charles had forgot all about the Kvikk Lunsj and had to go back home to get it. This has left him very late, and to make things worse all the parking spots close to the meeting spot has been taken. Given the workout diary of Charles, could you help find the fastest path to the meeting spot?

Input

The first line of the input consists of three space-separated integers N, S, T, representing the number of intersections in the ski tracks networks, the index of the intersection where Charles parked his car and the index of the intersection where Charles agreed to meet Bernard. Then follows N lines. Each of the lines consists of N space-separated integers D_{ij}. The integer on line i and column j describes the time in minutes it takes for Charles to get from intersection i to intersection j.

Output

The output should be a single line consisting of the minimum number of minutes Charles will have to use to get to the meeting point to share some Kvikk Lunsj with Bernard.

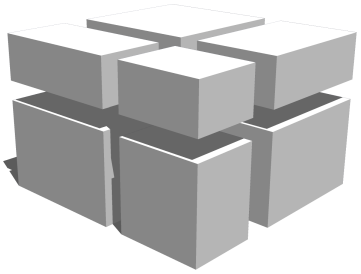
Limits

- $1 \leq N \leq 1\,000$
- $0 \leq S, T < N$
- $1 \leq D_{ij} < 10\,000$ for $i \neq j$
- $D_{ii} = 0$

Sample Input 1	Sample Output 1
4 1 3 0 1 3 14 2 0 4 22 3 10 0 7 13 8 2 0	11

Cuboid Slicing Game

Problem ID: cuboidslicinggame



Ruben and Albert are what you can call abnormally smart. They are also both very fond of mathematically inspired games. Their only problem is that most games are too easy for them, and they end up beating everyone who dares challenge them. Because of that, they're now mostly playing against each other. To make things interesting, they had a professor design a new game for them.

This new game was interesting at first. Nowadays, however, Albert often complains that it is impossible for him to win a particular round. After long discussions, they've now decided to take this a step further, and actually figure out who'd win if they both played optimally. They need you to write a computer program that does this for them.

A state in the game consists of one or more $x \times y \times z$ cuboids. A (legal) move is choosing a cuboid, then a value for each of the three axes (basically choosing three planes), and then cutting the cuboid along these (thus removing a $1 \times y \times z$, $x \times 1 \times z$ and a $x \times y \times 1$, all overlapping, cuboid). In effect you've created between 0 and 8 (inclusive) smaller cuboids. All three planes cut from the cuboid need to be on the cuboid (you can't cut away a hypothetical cuboid on the outside of the real one).

An example might be in order. You've chosen a $3 \times 5 \times 4$ cuboid, and are about to cut it. You now need to choose the three planes. This means you need an x between 1 and 3, a y between 1 and 5 and a z between 1 and 4. Say you choose 2, 1 and 3, respectively. The first cut would alone cut the cuboid into two $1 \times 5 \times 4$ cuboids, the second into a single $3 \times 4 \times 4$ cuboid, while the third would alone cut the cuboid into a $3 \times 5 \times 1$ and a $3 \times 5 \times 2$ cuboid. Put together these cuts produces 4 new smaller cuboids, of sizes $1 \times 4 \times 1$,$1 \times 4 \times 1$,$1 \times 4 \times 2$ and $1 \times 4 \times 2$. Note that cutting a cuboid with an axis of size 1 would remove it altogether.

The players take turns making a move. The winner is the player that removes the last cuboid.

Input

The first line of input is a line containing either `RUBEN` or `ALBERT`, the name of the player who starts that particular round.
Then follows a line containing N, the number of cuboids that particular game starts with.
N lines follow, each describing a cuboid. A cuboid description consists of three numbers, x, y and z, the size of that particular cuboid.

Output

Output the name of the player that wins the game (either `RUBEN` or `ALBERT`).

Limits

- $1 \leq N \leq 100$
- $1 \leq x, y, z \leq 30$

Sample Input 1

```
RUBEN
1
4 1 7
```

Sample Output 1

```
RUBEN
```

Sample Input 2

```
ALBERT
2
4 4 4
2 2 2
```

Sample Output 2

```
RUBEN
```

Sample Input 3	Sample Output 3
ALBERT 3 6 3 7 18 1 9 7 5 8	ALBERT

Draft Time
Problem ID: drafttime

The NFL draft is soon upon us and it is only a few weeks until the best college players will have their dreams come true by joining a professional (American) football team. As always your friend Bubba just won't stop talking about it.

The NFL draft is a process where N NFL teams select college players, one at a time, in M rounds from a pool of K total players. Bubba has checked with all his sources in the different teams and already knows how each team rates each of the college players. Knowing how the draft will end a few weeks early has left Bubba with some time to solve his favourite problem. Would it be possible be to make this draft a happy draft; a draft where all the teams and all the players are happy with the result?

Bubba thinks all teams and all players will be happy with the result if there exists no player P and team T such that *both* of the following are true:

- P is undrafted or would prefer to go to team T over the team he got drafted by
- T can draft more players or would prefer P over one of the players they have already drafted

Thankfully all the college players keep their SnapFace profiles up to date with a ranked list of their favourite NFL teams. Can you help Bubba create a happy draft?

Input

The first line of the input consists of three integers N, M and K representing the number of NFL teams, the number of rounds in this years draft, and the number of players in the draft.
Then follows N lines. Each line begins with a word, the name of the NFL team, followed by K words, the name of every player in the draft, rated from best to worst according to this team.
Then follows K lines. Each line begins with a word, the name of the player, followed by N words, the name of every NFL team in the draft, rated from best to worst according to this player.

Output

Output the result of a happy draft if one is possible and `Hello darkness my old friend!` if no happy draft can be made. A happy draft should be formatted on N lines. Each line begins with a word, the name of the NFL team, followed by M words, the names of the players drafted by this team. Any happy draft on this format will be accepted.

Limits

- $1 \leq N \leq 50$
- $1 \leq M \leq 100$
- $N \cdot M \leq K \leq 10\,000$
- All team names T and player names P are unique, and consists of between 1 and 20 characters in the range `a-z`.

Sample Input 1	Sample Output 1
3 2 6 browns martellus tyrod tom john teddy danny rams danny tyrod martellus john teddy tom giants teddy danny tyrod john tom martellus teddy giants browns rams danny giants browns rams tyrod giants rams browns john giants rams browns tom browns giants rams martellus giants browns rams	browns martellus tom rams tyrod john giants teddy danny

Gear Changing

Problem ID: gearchanging

Lucy is making her road bike ready for the season. All parts have been checked for wear and tear and washed clean and she is ready to put the beauty back together again. There is only one thing troubling her. During the winter she has been to several motivational talks by previous professional cyclists. They have all talked about the importance of correct cadence while cycling. The cadence is the frequency your feet are pedaling at when you are cycling. Lucy is now wondering if she should change her gears setup before she reassembles the bike and goes on her first ride for the year.

Bike gears typically consist of a set of different-sized gears on the crank (the thing connected to the pedals) and a set of different-sized gears on the back wheel. These are connected by the chain to drive the bike forward. Can you help her figure out if she can keep her current setup and still change to all gears without it affecting her cadence too much? Specifically she does not want her cadence to be increased by more than $P\%$ when changing to the next lighter gear. We assume that the bicycle will continue at the same speed immediately after the gear change.

Input

The first line of the input consists of three integers N, M and P representing the number of gears on the crank, the numbers of gears on the back wheel, and the maximum cadence change in percentage Lucy will tolerate when changing gears.
Then follows a line with N integers C_i each representing the number of teeth on each of the gears on the crank.
Then follows a line with M integers D_j each representing the number of teeth on each of the gears on the back wheel.

Output

Output a single line with the text `Ride on!` if Lucy can change through all the gears from heaviest to lightest in her current setup and `Time to change gears!` if she cannot.

Limits

- $1 \leq N, M \leq 100$
- $0 \leq P \leq 1\,000$
- $3 \leq C, D \leq 100\,000$

Sample Input 1

```
2 11 15
50 34
11 12 13 14 16 18 20 22 25 28 32
```

Sample Output 1

```
Ride on!
```

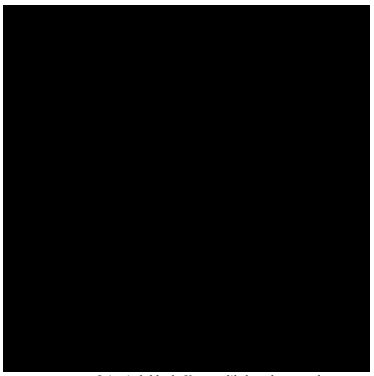
Sample Input 2

```
3 8 10
30 39 48
10 13 16 22 25 28 32 35
```

Sample Output 2

```
Time to change gears!
```

Gruesome Cave
Problem ID: gruesomecave



It is pitch black. You are likely to be eaten by a grue.

The archaeologist Joanina Iones has been hard at work deciphering the arcane glyphs in front of a secret cave entrance. The glyphs tell the story of the cave: It has existed long before the beginning of time, and with it, a single grue was brought to life. It has, of course, also a diamond in it, which is why Joanina is here.

Encountering a grue will always lead to a horrible death, but fortunately Joanina knows how grues work:

- Every hour, a grue moves to a random neighboring tile with uniform probability, except if a human is in the cave. While a human is in the cave it stays in one place.
- A grue does not like diamonds nor cave entrances, so it will never move to a cave entrance or a diamond tile.

The glyphs also describe the map of the cave, as well as where the diamond is located. If Joanina wants to retrieve the diamond, how likely is it she encounters the grue using the least risky path?

Input

The first line of the input consists of two space-separated integers L and W, indicating the length and width of the map, respectively.
Then follows L lines each consisting of a string of length W. Here, ' ' denotes an empty tile, '`#`' denotes a wall, '`E`' denotes the cave entrance and '`D`' denotes the diamond's location.

Output

Output the probability that Joanina will encounter the grue, assuming she chooses the least risky path. Your answer must have an absolute or relative error of at most 10^{-6}.

Limits

- $3 \leq L, W \leq 30$
- $2 \leq S \leq 70$, where S is the number of ' ' tiles.
- The border of the map can only contain '`E`' and '`#`' tiles.
- The map will only contain the characters ' ', '`#`', '`E`' and '`D`' and there will only be one '`E`' character and one '`D`' character per map.
- There will always be a way from the entrance to the diamond.
- All ' ' tiles form a connected graph.

Sample Input 1	Sample Output 1
5 6 ###### ## ### E D# ## ### ######	0.75

Number Anagrams

Problem ID: nanagrams

Tom is really interested in numbers, so he decided to become an accountant. To his great pleasure, he gets to see lots of numbers every day in his new job. Even though it is exciting to see all these numbers, the actual work happens to be boring. In order to combat boredom, he has created a game to help him have fun with all the interesting numbers.

From all the numbers he sees during a day, he picks one. Then he adds a digit, and rearranges the digits so that they form another number he has seen during the day. For example, 123 can be turned into 2 031 by adding a 0 and rearranging the digits. He keeps doing this with the new number until he has created a number that cannot be transformed into one he has seen using this method.

Letś say that he has seen the numbers 2, 32, 322, 243, and 1 234 during the day. If he starts with 2, he can produce 32 next. Then he can choose between 322 and 243. If he chooses 322, the game is over, since he cannot create new numbers from there. If he chooses 243 instead, he can have one more turn, creating 1 234.

Tom likes long numbers, so he tries to make the numbers with the largest number of digits possible. In the example above, he would choose to make 234 as his second move, so that he would be able to make 1 234.

Help Tom find out how many ways he can make numbers with the largest number of digits possible, for a given list of M starting numbers. The results can be very large, so you should output them modulo 1 000 000 007. Note that Tom is only interested in the length of the number, so if several numbers are longest, count the ways to make all of them. Note that transforming 55 to 555 only counts as one way of making 555, not three. If Tom sees the same number twice in a day, it still only counts as one number.

Since Tom sees a lot of numbers every day, we will use a Pseudo Random Number Generator to generate the numbers. The i^{th} number is generated using the following formula: $X_i = (a \cdot X_{i-1} + b) \bmod c$. $X_1 = d$.

Input

The first line of the input consists of two space-separated integers N and M, indicating how many numbers to generate, and how many starting numbers you need to process.
The second line of the input consists of four space-separated integers a, b, c, d, describing the pseudo random number generator.
The next M lines each contain a single integer S_i, describing a starting number that you should process.

Output

For each of the M starting numbers, output a line with a single integer, representing the number of ways he can make the longest possible numbers for that starting number.

Example

The numbers that would be generated in the example are $1, 5, 17, 53, 61, 85, 57, 73, 21, 65$.

Limits

- $1 \leq N \leq 100\,000$
- $1 \leq M \leq 1\,000$
- $1 \leq a, b, d < c \leq 100\,000$
- All starting numbers S_i will be numbers that Tom has seen during the day.

Sample Input 1	Sample Output 1
10 2 3 2 100 1 1 5	3 4

Picture Day

Problem ID: pictureday

The school for applied astrology has decided to take pictures of all their students. With moon phases being what they are, combined with Saturn's position relative to Venus, they are forced to focus on cost.

In order to minimise cost, they have asked three photographers for their price and process. Photographer A wants to take G_a group photos costing a per photo. Photographer B wants to take G_b group photos costing b per photo. Photographer C wants to take photos with one student in each photo, costing c per photo.

Each student will be in exactly one of A's groups and exactly one of B's groups.

The three photographers are pretty flexible, so they will allow the school to choose some pictures from each if they want to.

What is the lowest price they have to pay so that each student will be in at least one picture?

Input

The first line of the input consists of a single integer N, the number of students in the school.
The second line consists of two space-separated integers, G_a, G_b, representing the number of groups for photographers A and B.
The third line consists of the three space-separated integers a, b, c, representing the prices of the tree photographers.
The next G_a lines represent photographer A's groups, and consist of an integer G_{ai}, representing the number of students in that group, followed by G_{ai} space-separated integers S_j, representing the student numbers in that group.
The next G_b lines represent photographer B's groups, and consist of an integer G_{bi}, representing the number of students in that group, followed by G_{bi} space-separated integers S_j, representing the student numbers in that group.

Output

The output should be a single integer representing the cost of photographing all students at least once.

Limits

- $1 \leq N \leq 1\,000$
- $1 \leq G_a, G_b \leq 50$
- $1 \leq G_a, G_b \leq N$
- $1 \leq G_{ai}, G_{bi} \leq N$
- $1 \leq a, b, c \leq 1\,000$
- $1 \leq S_j \leq N$

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

Polygon Game

Problem ID: polygongame

Jack and Jill are playing a game they call Polygon Game. In this game they start with a convex polygon and then take turns dividing the polygon into smaller parts. They divide the polygon by drawing a straight line from one point on the border of the original polygon to another point on the border of the original polygon. The goal of the game is then to guess which of the resulting polygons has the largest area.

Input

The first line of the input consists of two integers N, M: The number of vertices in the original polygon, and the number of lines drawn by Jack and Jill.
Then follows N lines, each consisting of two integers X_i and Y_i, representing the coordinates of the i^{th} of the vertex in original polygon given in a counter-clockwise order.
Then follows M lines, each consisting four real numbers X_1, Y_1, X_2, and X_2, representing the two endpoints of a line drawn by Jack and Jill.

Output

The output should be a single line consisting of the area of the largest polygon created. Your answer must have an absolute or relative error of at most 10^{-6}.

Limits

- $3 \leq N \leq 25$
- $2 \leq M \leq 50$
- M will be an even number.
- $0 \leq X_i, Y_i \leq 1\,000$
- The polygon's vertices are distinct.
- All lines will start and end on the border of the original polygon with an absolute error of at most 10^{-6}, and all lines will divide the original polygon into two parts.
- Real numbers in the input will have at most 20 digits after the decimal point.

Sample Input 1	Sample Output 1
4 2 0 0 1 0 1 1 0 1 0 0 1 1 0.5 0 0.5 1	0.375

Secret Santa Cycles

Problem ID: secretsantacycles

Tina is organizing Secret Santa at work. Secret Santa is a game where everyone is assigned to give a gift to another person, in secret. On the last day before Christmas, everyone opens their present.

The opening ceremony is organized in the following way: Tina picks a random person who gets to open their present. Then the person who gave that present gets to open theirs, and so on, until they get to a person who already opened theirs. At this point, Tina has to pick another random person, and continue the process from there.

Tina doesn't like picking a random person, so she very much prefers that everyone form one big cycle. You have been given the task to find out if this is the case, so she can mentally prepare for how many random people she has to pick.

To your great horror, you find out that a secret Grinch has messed with the process so that each person A has to give a present to a random person B, instead of each person B receiving a present from a random person A. Specifically this means that some people might get no gifts, and some people might get multiple gifts.

This mistake obviously needs to be fixed. What is the smallest number of people that have to be assigned a new person to give a gift, so that everyone will get a gift, and Tina only has to pick one random person in the opening ceremony?

Input

The first line of the input consists of a single integer N, the number of people involved.
The following N lines each consists of a single integer G_i, the id of the person they should give a gift to.

Output

Output the number of people who have to be assigned a new person as their secret Santa target.

Limits

- $1 \leq N \leq 10\,000$
- $1 \leq G_i \leq N$

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

When Planetoids Align

Problem ID: whenplanetoidsalign

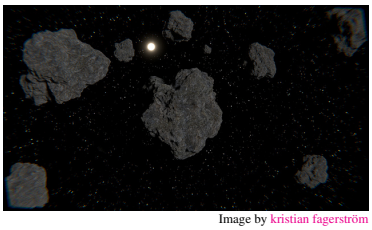


Image by kristian fagerström

After the Twilight Wars, both Venus, Earth and Mars broke into small pieces we now call planetoids. Fortunately humanity survived, and has managed to repopulate the remains of what was once was the Planetarian Trinity. To keep the planetoids united, we formed the InterPlantoidary Programming Competition (IPPC).

To compete in the IPPC, you need to qualify through one of the regional finals. A region is defined as all planetoids whose distance from Sol is within a specified interval (A_i, B_i). A planetoid can be part of multiple regions.

Unfortunately, the regional finals seldom happen, as they require low InterPlaNet delays between the planetoids. In fact, they only happen when all planetoids in the region are aligned along the *Mean Axis* (empty regions are always aligned along the *Mean Axis*).

Given that all planetoids are aligned along the *Mean Axis* today, when is the next Old Earth day a region could hold a regional final?

Input

The first line of the input consists of two space-separated integers P and R, representing the number of planetoids and regions.

The next P lines each consist of a number r_i and an integer o_i, representing the i^{th} planetoid's orbital radius and orbital period in Old Earth days. The orbital period represents the number of old earth days between each time the planetoid is aligned along the *Mean Axis*.

The next R lines consist of two numbers A_i and B_i, describing the interval that defines the i^{th} region.

Output

For each region, output a line with the number of Old Earth days until the next time a regional final can be arranged in this region.

Limits

- $1 \leq P, R \leq 100\,000$
- $1 \leq o_i \leq 1\,000\,000$
- $0.5 < r_i < 3.0$
- $0.5 \leq A_i < B_i \leq 3.0$
- No r_i will be closer to any A_j or B_j than 10^{-8}.
- There are less than 10^{18} Old Earth days until the next time all planetoids are aligned along the *Mean Axis*.
- Real numbers in the input will have at most 20 digits after the decimal point.

Sample Input 1	Sample Output 1
7 4 1.0 365 1.02 400 1.20 301 0.8 250 0.81 224 1.52 686 1.40 1843 0.9 1.3 1.211119 1.6 0.79 1.1 0.7 0.9	8789200 1264298 2044000 28000

Witchwood
Problem ID: witchwood


Photo by Mike Lewinski

Maaba is playing the game *Lumberjack*, where the goal is to build log huts. Her current customer wants a hut made completely out of *witchwood*, which protects the hut from curses, hexes and magic spells – or at least that's what the customer claims.

However, witchwood is not easy to handle: Some witch trees are highly volatile, and logs made from those trees will spontaneously combust when placed on the hut location. This will destroy both the original log and all logs nearby. Unfortunately, it is not possible to see the difference before the log is placed on the hut location, but it is known that a log from location i will have probability P_i of combusting.

To avoid spending too much time, Maaba has devised a simple way to build the hut:

- She uses T_i seconds to retrieve and place a single witchwood log from location i.
- If the hut logs combust, she waits K seconds to wait for the fire to stop, then collect the ashes. Afterwards, she starts over from scratch.
- If nothing happens, she continues with the next log.

As all locations have an infinite supply of witch trees, Maaba knows she will be done eventually. However, she wonders how long it would take her on average, if she retrieves logs from the right location at the right time. Could you help her?

Input

The first line of the input consists of three integers N, M, K: The number of logs required to build the hut, the number of locations with witchwood, and the time it takes for witchwood to stop burning.
Then follows M lines, each consisting of an integer T_i, and a real number P_i, representing the time it takes to retrieve a log from i and the probability that the log will spontaneously combust when placed.

Output

The output should be a single real number representing the expected time in seconds Maaba will use to build a log hut out of only witchwood. Your answer must have an absolute or relative error of at most 10^{-6}.

Limits

- $1 \leq N, M \leq 1000$
- $0 \leq K \leq 1000$
- $1 \leq T_i \leq 1000$
- $0 \leq P_i < 1$
- The result will always be less than 10^{30}.
- Real numbers in the input will have at most 20 digits after the decimal point.

Sample Explanation

In the provided example, the optimal choice is to retrieve a log from 2 when she has placed 0 logs, a log from 0 when she has placed 1 log, and a log from 1 when she has placed 2 logs.

Sample Input 1	Sample Output 1
3 3 1 8 0.5 20 0.3 1 0.8	79.0