Practice Problem A

Candy Compress

Time limit: 2 seconds

You are developing a mobile game called *Candy Compress*. In this game, there are several colored candies lined up from left to right. There are 26 possible colors. At any point in time, the player can choose to add a candy at any position or to remove a subset of neighboring candies to get points depending on the colors of the removed candies.

To develop the game, you need to implement the following data structure. Initially, the data structure loads a 1-indexed string of n characters, which represents the colors of the initial candies. The string consists of only uppercase Latin characters (`A`–`Z`). After loading the string, there are q operations that you need to support. Each operation is either one of the following:

Operation 1: Insert the uppercase Latin character c to the string so that the character is in the i-th position in the new string. In particular, $i = 1$ means inserting character c at the beginning of the string. It is guaranteed that $1 \leq i \leq m + 1$, where m is the length of the string just before this operation.

Operation 2: Remove the characters of the string from the l-th to the r-th position, inclusive. It is guaranteed that $1 \leq l \leq r \leq m$, where m is the length of the string just before this operation.

For each Operation 2, your data structure needs to determine the characters that are removed, so that the game can calculate the number of points to be given to the player. In other words, you need to determine the content of the string from the l-th position to the r-th position just before the operation.

Input

The first line of input contains two integers n and q ($1 \leq n \leq 300\,000$; $1 \leq q \leq 300\,000$). The second line contains a string consisting of n uppercase Latin characters representing the initial string to be loaded by the data structure. Each of the next q lines represents an operation with either one of the following formats:

1. `1` c i represents an Operation 1. It is guaranteed that c is an uppercase Latin character and $1 \leq i \leq m + 1$, where m is the length of the string just before this operation.
2. `2` l r represents an Operation 2. It is guaranteed that $1 \leq l \leq r \leq m$ where m is the length of the string just before this operation.

The operations are given in the order they are to be performed. It is guaranteed that there is at least one Operation 2.

Output

For each Operation 2 in order, output one line containing the characters that are removed by the operation.

Sample Input #1	Sample Output #1
3 5 XPA 1 P 3 1 P 5 2 2 5 1 Y 2 2 1 2	PPAP XY

Explanation for the sample input/output #1

The first and second operations modify the string in the data structure as follows: `XPA` $\rightarrow$ `XP`**`P`**`A` $\rightarrow$ `XPPA`**`P`**. The third operation removes `PPAP` from the string, leaving the string `X` in the data structure. The fourth operation inserts the character `Y` to the end of the string. The last operation removes `XY` from the string, leaving an empty string in the data structure.

Sample Input #2	Sample Output #2
27 7 ICPCASIAPACIFICCHAMPIONSHIP 2 5 8 2 5 11 1 A 5 1 P 6 1 A 7 1 C 8 2 1 8	ASIA PACIFIC ICPCAPAC

Practice Problem B
Combination Lock
Time limit: 2 seconds

Your house is protected by a combination lock containing n rotating discs, numbered from 1 to n. On a typical combination lock, each rotating disc has 10 symbols, represented by integers between 0 to 9, inclusive. Since you are a mathematician, your combination lock is not typical. Instead, each rotating disc on your combination lock may have a different number of symbols. In particular, rotating disc i has $b_i - a_i + 1$ symbols, represented by integers between a_i to b_i, inclusive.

The combination lock is unlocked when each rotating disc displays one integer, and any pair of two integers displayed by the rotating disc are coprime. Two integers are coprime if they do not have any common positive factors other than 1.

You want to unlock the combination lock, so you want to determine what integer to be displayed on each combination lock to satisfy the requirement above. It is possible that your combination lock was sabotaged when you were gone so it might be impossible to unlock your combination lock.

Input

The first line of input contains one integer n ($2 \leq n \leq 50$). Each of the next n lines contains two integers. The i-th line contains a_i and b_i ($1 \leq a_i \leq b_i \leq 50$).

Output

Output one line containing n integers, where the i-th integer represents the integer to be displayed by rotating disc i to unlock the combination lock. If there are multiple solutions, you can output any of them. If there is no solution, output just the integer `-1`.

Sample Input #1	Sample Output #1
4 3 9 2 8 1 4 2 10	3 7 1 10

Explanation for the sample input/output #1

Other solutions such as `9 5 2 7` are also accepted.

Sample Input #2	Sample Output #2
3 5 6 5 6 2 6	-1

This page is intentionally left blank.

Practice Problem C

Online Assignment

Time limit: 2 seconds

You have to finish an assignment by the end of today. You can submit the assignment online and, even if your submission is rejected, you still have several more submission chances.

The assignment consists of five questions, each of which asks whether or not a given statement is true. For each submission, you have to submit answers for all five questions. If your answers are correct for all the questions, your assignment is completed. However, if any of your answers are wrong, your submission is rejected, and the number of correct answers will be told.

All the questions are so difficult that you cannot give confident answers to any of them. Your task here is to write a program that tries to submit answers repeatedly until the assignment is completed. Note that you can submit answers only a limited number of times.

Interaction

For each submission, your program should output exactly five characters without any delimiters followed by a newline to the standard output. Each character should be either `t` or `f`. The i-th character being `t` means the statement of the i-th question is judged to be true. Similarly, the character `f` means it is judged to be false.

After the output, you will receive a feedback from the standard input: either "`rejected` C" or "`completed`" followed by a newline. When you receive `rejected` C, it means that at least one of your answers is wrong, and C is the number of correct answers. In this case, your program should submit another answer. When you receive `completed`, your assignment is completed. Your program should terminate without extra output.

Your program can submit answers at most 12 times. If your 12-th submission is still rejected, your program will be judged as "Wrong Answer".

Notes on interactive judging:

- *The evaluation is non-adversarial, meaning that the correct answers to the questions are chosen in advance rather than in response to your submissions.*
- *Do not forget to flush output buffers after writing. See the "Judging Details" document for details.*
- *You are provided with a command-line tool for local testing, together with input files corresponding to the sample interactions. You can download these files from DOMjudge. The tool has comments at the top to explain its use.*

Read	Sample Interaction #1	Write
		tffft
rejected 4		
		fffft
rejected 3		
		ttfft
completed		

Read	Sample Interaction #2	Write
		fffff
completed		

Practice Problem D
Squeeze the Cylinders
Time limit: 2 seconds

Laid on the flat ground in the stockyard are n heavy metal cylinders with possibly different diameters but with the same length. Their ends are aligned and their axes are oriented to exactly the same direction.

We'd like to minimize the area occupied. The cylinders are too heavy to lift up, although rolling them is not too difficult. So, we decided to push the cylinders with two high walls from both sides.

Your task is to compute the minimum possible distance between the two walls when cylinders are squeezed as much as possible. Cylinders and walls may touch one another. They cannot be lifted up from the ground, and thus their order cannot be altered.

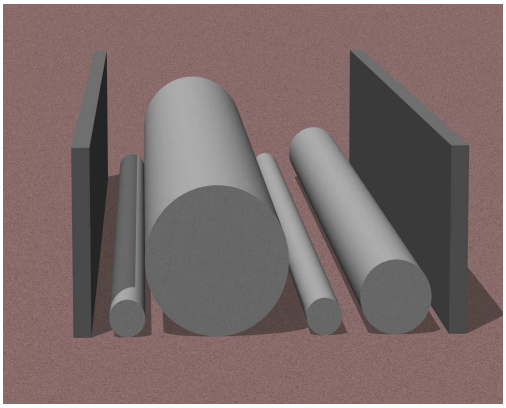

Figure D.1: Cylinders between two walls.

Input

The first line of input contains one integer n ($1 \leq n \leq 500$). The second line contains n positive integers not more than 10 000, representing the radii of cylinders from left to right.

Output

Output the distance between the two walls when they fully squeeze up the cylinders. The relative error of the output must be within 10^{-9}.

Sample Input #1	Sample Output #1
2 10 10	40.000000000

Sample Input #2	Sample Output #2
2 4 12	29.856406461

Sample Input #3	Sample Output #3
5 1 10 1 10 1	40.000000000

Explanation for the sample input/output #1

Figure D.2 illustrates possible positions of the cylinders to achieve the minimum possible distance between the two walls in the sample inputs.

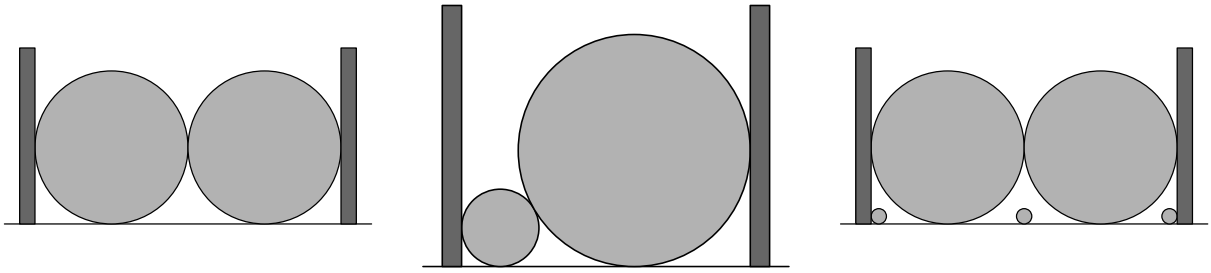

Figure D.2: Illustrations of the sample inputs (from left to right).

(Note: This task has been adapted from ICPC Asia Tsukuba Regional Contest 2015 with an approval from the contest judge)

Practice Problem E
Candy Compress
Time limit: 2 seconds

You are developing a mobile game called *Candy Compress*. In this game, there are several colored candies lined up from left to right. There are 26 possible colors. At any point in time, the player can choose to add a candy at any position or to remove a subset of neighboring candies to get points depending on the colors of the removed candies.

To develop the game, you need to implement the following data structure. Initially, the data structure loads a 1-indexed string of n characters, which represents the colors of the initial candies. The string consists of only uppercase Latin characters (`A`–`Z`). After loading the string, there are q operations that you need to support. Each operation is either one of the following:

Operation 1: Insert the uppercase Latin character c to the string so that the character is in the i-th position in the new string. In particular, $i = 1$ means inserting character c at the beginning of the string. It is guaranteed that $1 \leq i \leq m + 1$, where m is the length of the string just before this operation.

Operation 2: Remove the characters of the string from the l-th to the r-th position, inclusive. It is guaranteed that $1 \leq l \leq r \leq m$, where m is the length of the string just before this operation.

For each Operation 2, your data structure needs to determine the characters that are removed, so that the game can calculate the number of points to be given to the player. In other words, you need to determine the content of the string from the l-th position to the r-th position just before the operation.

Input

The first line of input contains two integers n and q ($1 \leq n \leq 300\,000$; $1 \leq q \leq 300\,000$). The second line contains a string consisting of n uppercase Latin characters representing the initial string to be loaded by the data structure. Each of the next q lines represents an operation with either one of the following formats:

1. `1` c i represents an Operation 1. It is guaranteed that c is an uppercase Latin character and $1 \leq i \leq m + 1$, where m is the length of the string just before this operation.
2. `2` l r represents an Operation 2. It is guaranteed that $1 \leq l \leq r \leq m$ where m is the length of the string just before this operation.

The operations are given in the order they are to be performed. It is guaranteed that there is at least one Operation 2.

Output

For each Operation 2 in order, output one line containing the characters that are removed by the operation.

Sample Input #1	Sample Output #1
3 5 XPA 1 P 3 1 P 5 2 2 5 1 Y 2 2 1 2	PPAP XY

Explanation for the sample input/output #1

The first and second operations modify the string in the data structure as follows: `XPA` $\rightarrow$ `XP`**`P`**`A` $\rightarrow$ `XPPA`**`P`**. The third operation removes `PPAP` from the string, leaving the string `X` in the data structure. The fourth operation inserts the character `Y` to the end of the string. The last operation removes `XY` from the string, leaving an empty string in the data structure.

Sample Input #2	Sample Output #2
27 7 ICPCASIAPACIFICCHAMPIONSHIP 2 5 8 2 5 11 1 A 5 1 P 6 1 A 7 1 C 8 2 1 8	ASIA PACIFIC ICPCAPAC

Practice Problem F
Combination Lock
Time limit: 2 seconds

Your house is protected by a combination lock containing n rotating discs, numbered from 1 to n. On a typical combination lock, each rotating disc has 10 symbols, represented by integers between 0 to 9, inclusive. Since you are a mathematician, your combination lock is not typical. Instead, each rotating disc on your combination lock may have a different number of symbols. In particular, rotating disc i has $b_i - a_i + 1$ symbols, represented by integers between a_i to b_i, inclusive.

The combination lock is unlocked when each rotating disc displays one integer, and any pair of two integers displayed by the rotating disc are coprime. Two integers are coprime if they do not have any common positive factors other than 1.

You want to unlock the combination lock, so you want to determine what integer to be displayed on each combination lock to satisfy the requirement above. It is possible that your combination lock was sabotaged when you were gone so it might be impossible to unlock your combination lock.

Input

The first line of input contains one integer n ($2 \leq n \leq 50$). Each of the next n lines contains two integers. The i-th line contains a_i and b_i ($1 \leq a_i \leq b_i \leq 50$).

Output

Output one line containing n integers, where the i-th integer represents the integer to be displayed by rotating disc i to unlock the combination lock. If there are multiple solutions, you can output any of them. If there is no solution, output just the integer `-1`.

Sample Input #1	Sample Output #1
4 3 9 2 8 1 4 2 10	3 7 1 10

Explanation for the sample input/output #1

Other solutions such as `9 5 2 7` are also accepted.

Sample Input #2	Sample Output #2
3 5 6 5 6 2 6	-1

This page is intentionally left blank.

Practice Problem G

Online Assignment

Time limit: 2 seconds

You have to finish an assignment by the end of today. You can submit the assignment online and, even if your submission is rejected, you still have several more submission chances.

The assignment consists of five questions, each of which asks whether or not a given statement is true. For each submission, you have to submit answers for all five questions. If your answers are correct for all the questions, your assignment is completed. However, if any of your answers are wrong, your submission is rejected, and the number of correct answers will be told.

All the questions are so difficult that you cannot give confident answers to any of them. Your task here is to write a program that tries to submit answers repeatedly until the assignment is completed. Note that you can submit answers only a limited number of times.

Interaction

For each submission, your program should output exactly five characters without any delimiters followed by a newline to the standard output. Each character should be either `t` or `f`. The i-th character being `t` means the statement of the i-th question is judged to be true. Similarly, the character `f` means it is judged to be false.

After the output, you will receive a feedback from the standard input: either "`rejected` C" or "`completed`" followed by a newline. When you receive `rejected` C, it means that at least one of your answers is wrong, and C is the number of correct answers. In this case, your program should submit another answer. When you receive `completed`, your assignment is completed. Your program should terminate without extra output.

Your program can submit answers at most 12 times. If your 12-th submission is still rejected, your program will be judged as "Wrong Answer".

Notes on interactive judging:

- *The evaluation is non-adversarial, meaning that the correct answers to the questions are chosen in advance rather than in response to your submissions.*
- *Do not forget to flush output buffers after writing. See the "Judging Details" document for details.*
- *You are provided with a command-line tool for local testing, together with input files corresponding to the sample interactions. You can download these files from DOMjudge. The tool has comments at the top to explain its use.*

Read	Sample Interaction #1	Write
	tffft	
rejected 4		
	fffft	
rejected 3		
	ttfft	
completed		

Read	Sample Interaction #2	Write
	fffff	
completed		

Practice Problem H
Squeeze the Cylinders
Time limit: 2 seconds

Laid on the flat ground in the stockyard are n heavy metal cylinders with possibly different diameters but with the same length. Their ends are aligned and their axes are oriented to exactly the same direction.

We'd like to minimize the area occupied. The cylinders are too heavy to lift up, although rolling them is not too difficult. So, we decided to push the cylinders with two high walls from both sides.

Your task is to compute the minimum possible distance between the two walls when cylinders are squeezed as much as possible. Cylinders and walls may touch one another. They cannot be lifted up from the ground, and thus their order cannot be altered.

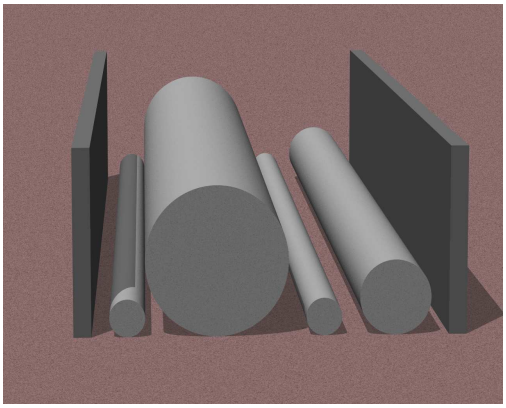

Figure H.1: Cylinders between two walls.

Input

The first line of input contains one integer n ($1 \leq n \leq 500$). The second line contains n positive integers not more than 10 000, representing the radii of cylinders from left to right.

Output

Output the distance between the two walls when they fully squeeze up the cylinders. The relative error of the output must be within 10^{-9}.

Sample Input #1	Sample Output #1
2 10 10	40.000000000

Sample Input #2	Sample Output #2
2 4 12	29.856406461

Sample Input #3	Sample Output #3
5 1 10 1 10 1	40.000000000

Explanation for the sample input/output #1

Figure H.2 illustrates possible positions of the cylinders to achieve the minimum possible distance between the two walls in the sample inputs.

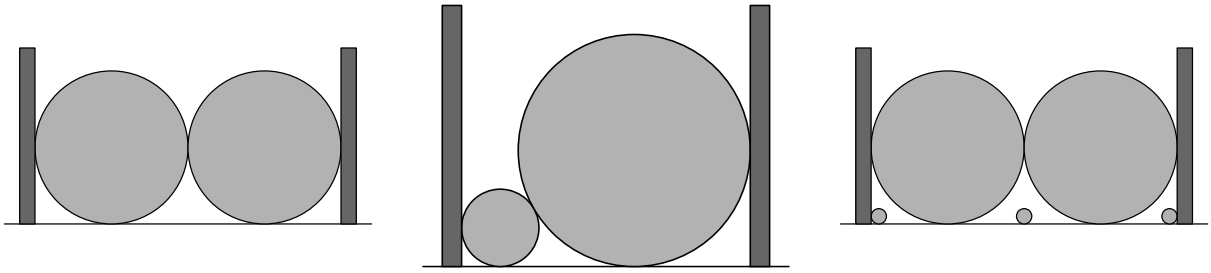

Figure H.2: Illustrations of the sample inputs (from left to right).

(Note: This task has been adapted from ICPC Asia Tsukuba Regional Contest 2015 with an approval from the contest judge)

Practice Problem I

Candy Compress

Time limit: 2 seconds

You are developing a mobile game called *Candy Compress*. In this game, there are several colored candies lined up from left to right. There are 26 possible colors. At any point in time, the player can choose to add a candy at any position or to remove a subset of neighboring candies to get points depending on the colors of the removed candies.

To develop the game, you need to implement the following data structure. Initially, the data structure loads a 1-indexed string of n characters, which represents the colors of the initial candies. The string consists of only uppercase Latin characters (`A`–`Z`). After loading the string, there are q operations that you need to support. Each operation is either one of the following:

Operation 1: Insert the uppercase Latin character c to the string so that the character is in the i-th position in the new string. In particular, $i = 1$ means inserting character c at the beginning of the string. It is guaranteed that $1 \leq i \leq m + 1$, where m is the length of the string just before this operation.

Operation 2: Remove the characters of the string from the l-th to the r-th position, inclusive. It is guaranteed that $1 \leq l \leq r \leq m$, where m is the length of the string just before this operation.

For each Operation 2, your data structure needs to determine the characters that are removed, so that the game can calculate the number of points to be given to the player. In other words, you need to determine the content of the string from the l-th position to the r-th position just before the operation.

Input

The first line of input contains two integers n and q ($1 \leq n \leq 300\,000$; $1 \leq q \leq 300\,000$). The second line contains a string consisting of n uppercase Latin characters representing the initial string to be loaded by the data structure. Each of the next q lines represents an operation with either one of the following formats:

1. `1` c i represents an Operation 1. It is guaranteed that c is an uppercase Latin character and $1 \leq i \leq m + 1$, where m is the length of the string just before this operation.
2. `2` l r represents an Operation 2. It is guaranteed that $1 \leq l \leq r \leq m$ where m is the length of the string just before this operation.

The operations are given in the order they are to be performed. It is guaranteed that there is at least one Operation 2.

Output

For each Operation 2 in order, output one line containing the characters that are removed by the operation.

Sample Input #1	Sample Output #1
3 5 XPA 1 P 3 1 P 5 2 2 5 1 Y 2 2 1 2	PPAP XY

Explanation for the sample input/output #1

The first and second operations modify the string in the data structure as follows: `XPA` $\rightarrow$ `XP`**`P`**`A` $\rightarrow$ `XPPA`**`P`**. The third operation removes `PPAP` from the string, leaving the string `X` in the data structure. The fourth operation inserts the character `Y` to the end of the string. The last operation removes `XY` from the string, leaving an empty string in the data structure.

Sample Input #2	Sample Output #2
27 7 ICPCASIAPACIFICCHAMPIONSHIP 2 5 8 2 5 11 1 A 5 1 P 6 1 A 7 1 C 8 2 1 8	ASIA PACIFIC ICPCAPAC

Practice Problem J
Combination Lock
Time limit: 2 seconds

Your house is protected by a combination lock containing n rotating discs, numbered from 1 to n. On a typical combination lock, each rotating disc has 10 symbols, represented by integers between 0 to 9, inclusive. Since you are a mathematician, your combination lock is not typical. Instead, each rotating disc on your combination lock may have a different number of symbols. In particular, rotating disc i has $b_i - a_i + 1$ symbols, represented by integers between a_i to b_i, inclusive.

The combination lock is unlocked when each rotating disc displays one integer, and any pair of two integers displayed by the rotating disc are coprime. Two integers are coprime if they do not have any common positive factors other than 1.

You want to unlock the combination lock, so you want to determine what integer to be displayed on each combination lock to satisfy the requirement above. It is possible that your combination lock was sabotaged when you were gone so it might be impossible to unlock your combination lock.

Input

The first line of input contains one integer n ($2 \leq n \leq 50$). Each of the next n lines contains two integers. The i-th line contains a_i and b_i ($1 \leq a_i \leq b_i \leq 50$).

Output

Output one line containing n integers, where the i-th integer represents the integer to be displayed by rotating disc i to unlock the combination lock. If there are multiple solutions, you can output any of them. If there is no solution, output just the integer `-1`.

Sample Input #1	Sample Output #1
4 3 9 2 8 1 4 2 10	3 7 1 10

Explanation for the sample input/output #1

Other solutions such as `9 5 2 7` are also accepted.

Sample Input #2	Sample Output #2
3 5 6 5 6 2 6	-1

This page is intentionally left blank.

Practice Problem K
Online Assignment
Time limit: 2 seconds

You have to finish an assignment by the end of today. You can submit the assignment online and, even if your submission is rejected, you still have several more submission chances.

The assignment consists of five questions, each of which asks whether or not a given statement is true. For each submission, you have to submit answers for all five questions. If your answers are correct for all the questions, your assignment is completed. However, if any of your answers are wrong, your submission is rejected, and the number of correct answers will be told.

All the questions are so difficult that you cannot give confident answers to any of them. Your task here is to write a program that tries to submit answers repeatedly until the assignment is completed. Note that you can submit answers only a limited number of times.

Interaction

For each submission, your program should output exactly five characters without any delimiters followed by a newline to the standard output. Each character should be either `t` or `f`. The i-th character being `t` means the statement of the i-th question is judged to be true. Similarly, the character `f` means it is judged to be false.

After the output, you will receive a feedback from the standard input: either "`rejected` C" or "`completed`" followed by a newline. When you receive `rejected` C, it means that at least one of your answers is wrong, and C is the number of correct answers. In this case, your program should submit another answer. When you receive `completed`, your assignment is completed. Your program should terminate without extra output.

Your program can submit answers at most 12 times. If your 12-th submission is still rejected, your program will be judged as "Wrong Answer".

Notes on interactive judging:

- *The evaluation is non-adversarial, meaning that the correct answers to the questions are chosen in advance rather than in response to your submissions.*
- *Do not forget to flush output buffers after writing. See the "Judging Details" document for details.*
- *You are provided with a command-line tool for local testing, together with input files corresponding to the sample interactions. You can download these files from DOMjudge. The tool has comments at the top to explain its use.*

Read	Sample Interaction #1	Write
		tffft
rejected 4		
		fffft
rejected 3		
		ttfft
completed		

Read	Sample Interaction #2	Write
		fffff
completed		

Practice Problem L
Squeeze the Cylinders
Time limit: 2 seconds

Laid on the flat ground in the stockyard are n heavy metal cylinders with possibly different diameters but with the same length. Their ends are aligned and their axes are oriented to exactly the same direction.

We'd like to minimize the area occupied. The cylinders are too heavy to lift up, although rolling them is not too difficult. So, we decided to push the cylinders with two high walls from both sides.

Your task is to compute the minimum possible distance between the two walls when cylinders are squeezed as much as possible. Cylinders and walls may touch one another. They cannot be lifted up from the ground, and thus their order cannot be altered.

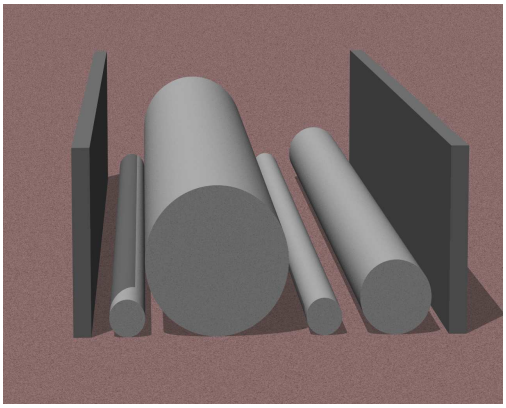

Figure L.1: Cylinders between two walls.

Input

The first line of input contains one integer n ($1 \leq n \leq 500$). The second line contains n positive integers not more than 10 000, representing the radii of cylinders from left to right.

Output

Output the distance between the two walls when they fully squeeze up the cylinders. The relative error of the output must be within 10^{-9}.

Sample Input #1	Sample Output #1
2 10 10	40.000000000

Sample Input #2	Sample Output #2
2 4 12	29.856406461

Sample Input #3	Sample Output #3
5 1 10 1 10 1	40.000000000

Explanation for the sample input/output #1

Figure L.2 illustrates possible positions of the cylinders to achieve the minimum possible distance between the two walls in the sample inputs.

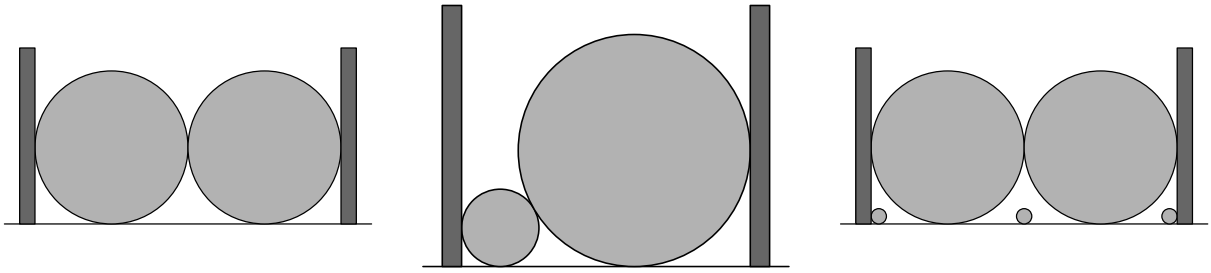

Figure L.2: Illustrations of the sample inputs (from left to right).

(Note: This task has been adapted from ICPC Asia Tsukuba Regional Contest 2015 with an approval from the contest judge)