icpc.foundation

ICPC North America Regionals

icpc international collegiate programming contest

JET BRAINS
icpc global sponsor
programming tools

upsilon pi epsilon
honor society

aws educate

icpc EdTech
gold sponsor

The 2021 ICPC South Central USA Regional Contest

Problem A
Average Character
Time Limit: 1 Second(s)

Have you ever wondered what the average ASCII character of any given string is? No? Never? Really? Well, is it a character in the string or something else?

Would you do this calculation by hand with an ASCII table? Probably not! All modern programming languages include functions for converting an ASCII character to an integer, and to convert an integer to an ASCII character. Of course, these functions often also handle unicode characters as well, but that is not part of this problem.

Given a string of ASCII characters, compute the average character. If the average character lies between two integer ASCII values, return the smaller one.

Input

The single line of input contains a single string s ($1 \leq |s| \leq 100$), which consists of ASCII text. All of the characters of s will be printable ASCII, between ASCII 32 (space: `' '`) and ASCII 126 (tilde: `'~'`). It will **NOT** contain any control characters such as carriage returns, line feeds, tabs, etc. It is **NOT** guaranteed to begin, end, or even contain a non-space character.

Output

Output a single ASCII character, which is the average of all of the ASCII characters in s.

Sample Input 1	Sample Output 1
`ABCDE`	`C`

Sample Input 2	Sample Output 2
`AbCdE`	`O`

Sample Input 3	Sample Output 3
`aBcDe`	`V`

This page is intentionally left blank.

Problem B
Black and White
Time Limit: 1 Second(s)

Black and White is a Chinese children's game played in rounds. During each round, the children who are playing all put their hands in either face-up ("White") or face-down ("Black"). If all the children but one make the same choice, then the "odd one out" sits out for the rest of the game. Play continues until there are only two children left.

Each child independently chooses whether to put their hand face-up with their own fixed probability. What is the expected number of rounds that such a game will last?

Input

The first line contains a single integer n ($2 \leq n \leq 20$), which is the number of children.

Each of the next n lines contains a single real number p ($0.1 \leq p \leq 0.9$). These are the probabilities for each child that they will put their hand in face-up. The probabilities will have at most three digits after the decimal point.

Output

Output a single real number, which is the expected number of rounds. The result must be accurate to within an absolute or relative error of 10^{-6}.

Sample Input 1

```
3
0.5
0.5
0.5
```

Sample Output 1

```
1.3333333
```

Sample Input 2

```
3
0.3
0.3
0.3
```

Sample Output 2

```
1.5873015
```

Sample Input 3

```
5
0.1
0.3
0.5
0.7
0.9
```

Sample Output 3

```
7.4752846
```

Problem C

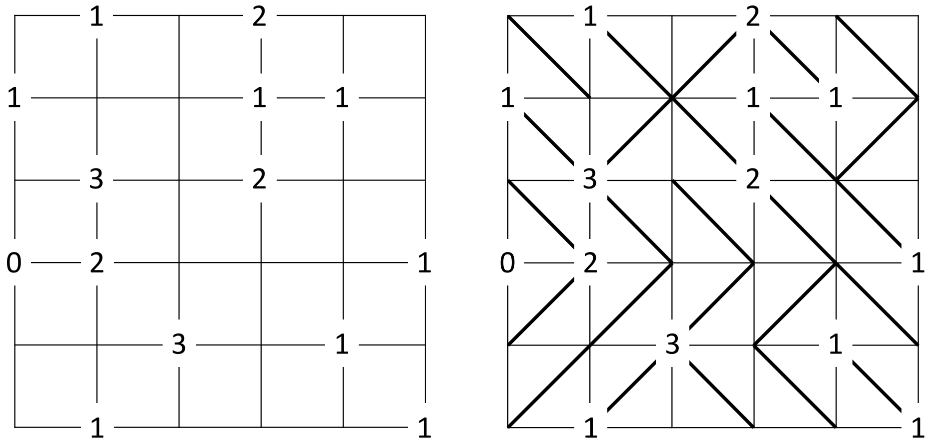
Diagonals

Time Limit: 10 Second(s)

Diagonals is a pencil puzzle which is played on a square grid. The player must draw a diagonal line corner to corner in every cell in the grid, either top left to bottom right, or bottom left to top right. There are two constraints:

- Some intersections of gridlines have a number from 0 to 4 inclusive on them, which is the exact number of diagonals that must touch that point.
- No set of diagonals may form a loop of any size or shape.

The following is a 5×5 example, with its unique solution:



Given the numbers at the intersections of a grid, solve the puzzle.

Input

The first line of input contains an integer n ($1 \leq n \leq 8$), which is the size of the grid.

Each of the next $n+1$ lines contains a string s ($|s| = n+1, s \in \{0,1,2,3,4,+\}^*$). These are the intersections of the grid, with `+' indicating that there is no number at that intersection.

The input data will be such that the puzzle has exactly one solution.

Output

Output exactly n lines, each with exactly n characters, representing the solution to the puzzle. Each character must be either `'/'` or `'\'`.

- Note that Sample 1 corresponds to the example in the problem description.

Sample Input 1

```
5
+1+2++
1++11+
+3+2++
02+++1
++3+1+
+1+++1
```

Sample Output 1

```
\\/\\
\/\\/
\\\\\
////\
//\\\
```

Sample Input 2

```
3
++++
+1+1
+31+
+0+0
```

Sample Output 2

```
/\/
///
/\/
```

Sample Input 3

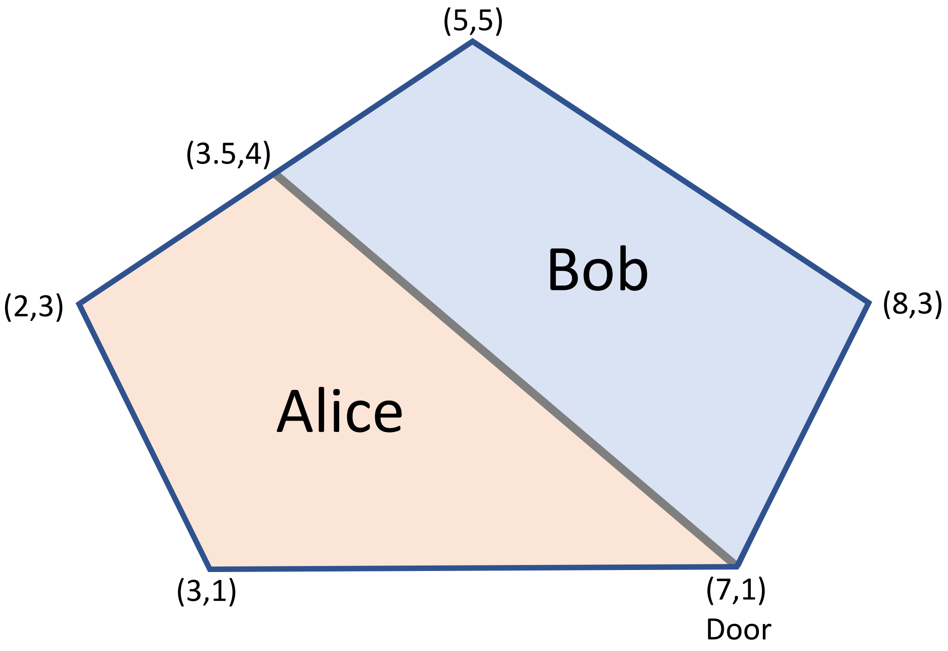
```
4
+++++
+3++2
++3++
+3+3+
++2+0
```

Sample Output 3

```
\//\
\\//
\\\/
/\//
```

Problem D
Dorm Room Divide
Time Limit: 1 Second(s)

Bob and Alice are roommates at the International College of Polygonal Chambers (ICPC). To avoid conflict, they've agreed to divide their dorm room in half—as closely as possible. However, the room is shaped so irregularly that they need your help!



Each dorm room is a convex polygon, with a single entrance. You need to figure out how to divide this room in half (by area) using a single straight line starting at the door, and terminating on a wall or corner of the room.

Input

The first line of input contains a single integer n ($3 \leq n \leq 2 \cdot 10^5$), which is the number of vertices describing the convex polygon.

Each of the next n lines contains two space-separated integers x and y ($-10^7 \leq x, y \leq 10^7$). These are the coordinates of the vertices of the convex polygon, in counterclockwise order. All points will be distinct.

The door is considered to be a single point located at the first vertex given in the input.

Output

Output two space-separated real numbers, which are the x and y coordinates of the other endpoint of the dividing line, such that the area of the room is divided in half. Each coordinate value must be accurate to within an absolute or relative error of 10^{-6}. Output x first, then y.

- Note that Sample 1 corresponds to the example in the problem description.

Sample Input 1

```
5
7 1
8 3
5 5
2 3
3 1
```

Sample Output 1

```
3.5 4
```

Sample Input 2

```
3
2 2
10 3
6 8
```

Sample Output 2

```
8 5.5
```

Problem E
Double Password
Time Limit: 1 Second(s)

A computer at ICPC headquarters is protected by a four-digit password—in order to log in, you normally need to guess the four digits exactly. However, the programmer who implemented the password check left a backdoor in the computer—there is a second four-digit password. If the programmer enters a four-digit sequence, and for each digit position the digit entered matches at least one of the two passwords in that same position, then that four-digit sequence will log the programmer into the computer.

Given the two passwords, count the number of distinct four-digit sequences that can be entered to log into the computer.

Input

The input consists of exactly two lines. Each of the two lines contains a string s ($|s| = 4, s \in \{\texttt{0}-\texttt{9}\}^*$). These are the two passwords.

Output

Output a single integer, which is the number of distinct four-digit sequences that will log the programmer into the system.

Sample Input 1	Sample Output 1
1111 1234	8

Sample Input 2	Sample Output 2
2718 2718	1

This page is intentionally left blank.

Problem F
Overdraft
Time Limit: 1 Second(s)

Banks often charge overdraft fees if you attempt to withdraw more money from your account than is available in your current balance. Given a sequence of deposits and withdrawals (and assuming each deposit and withdrawal is immediately reflected in your balance), determine the minimum (non-negative) starting balance you need to ensure that you will not be charged any overdraft fees over the course of the sequence.

Input

The first line of input contains a single integer n ($1 \leq n \leq 1{,}000$), which is the number of transactions.

Each of the next n lines contains a single integer t ($-10^6 \leq t \leq 10^6$, $t \neq 0$). These are the transactions, in the order that they occur. A positive number represents a deposit, a negative number represents a withdrawal. No two transactions occur simultaneously.

Output

Output a single non-negative integer, which is the minimum non-negative balance you must start with in your account in order to avoid any overdraft fees.

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

This page is intentionally left blank.

Problem G
Rise and Fall
Time Limit: 1 Second(s)

A number is said to *Rise and Fall* if the decimal representation can be broken up into two parts (possibly empty) where the first part has digits in nondecreasing order and the second part has digits in nonincreasing order.

Compute the largest number less than or equal to an input number that rises and falls.

Input

The first line of input contains an integer t ($1 \le t \le 10^5$), which is the number of test cases.

Each of the next t lines contains a single integer n ($1 \le n < 10^{100,000}$). Each is a single test case.

- Note: that is not a typo. The integer can be up to 10^5 digits long.

The sum of the lengths of all input test cases will not exceed 10^5.

Output

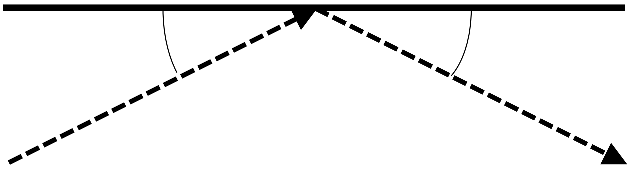
For each test case, output a single line with a single integer, which is the largest number less than or equal to the n for that test case that rises and falls.

Sample Input 1	Sample Output 1
2 29041 56577	29000 56555

This page is intentionally left blank.

Problem H
Square Bounce
Time Limit: 5 Second(s)

Given a square in the plane with corners at $(-1, -1)$, $(-1, 1)$, $(1, 1)$ and $(1, -1)$, we fire a ray from point $(-1, 0)$ into the interior of the square on a path with a given slope. The ray bounces off of the sides of the square with an angle of reflection which is the same as the angle of incidence to the side at the point of intersection.



After a number of bounces, the ray intersects one of the square's sides again at some point which has rational coordinates. Find those rational coordinates in reduced form.

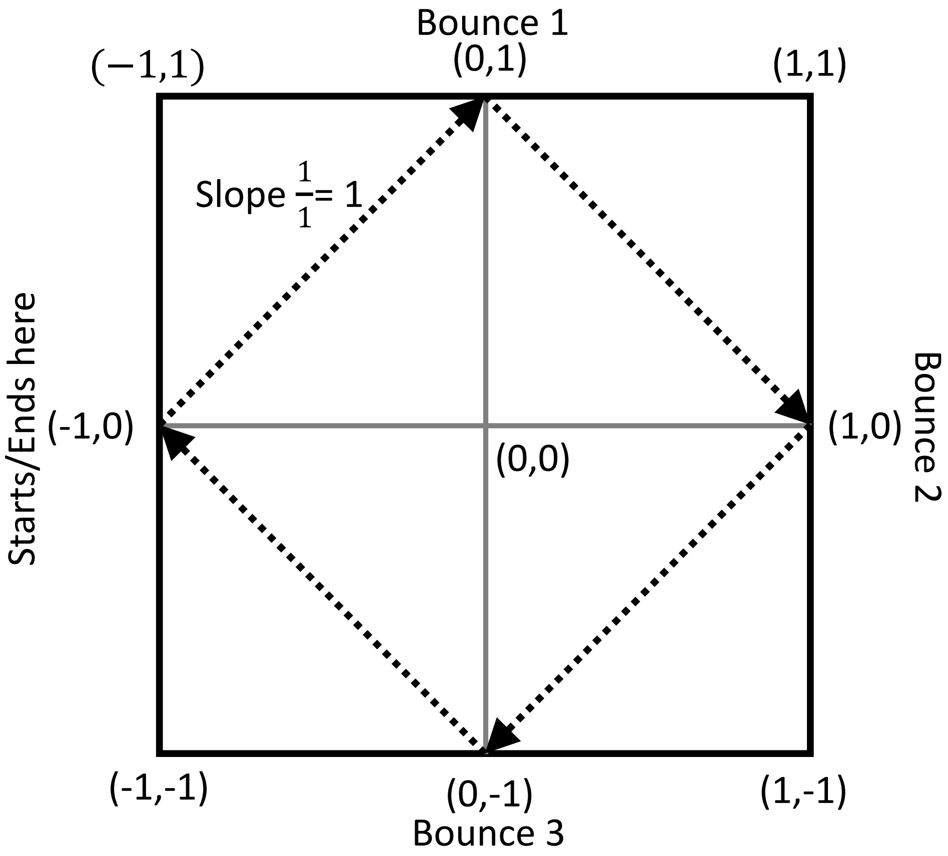
Input

The single line of input contains three integers a, b and n ($1 \le a, b, n \le 10^6$, $\gcd(a, b) = 1$), where the slope of the ray's initial path is a/b, and there are n bounces. Note that a and b are relatively prime. The slope will be chosen so that the ray never bounces at a corner of the square.

Output

Output a single line with four space-separated integers p, q, s and t, where $(p/q, s/t)$ is the final point where the ray hits a side of the square, p/q and s/t are in reduced form, and the denominators (q and t) are positive. If one of the coordinates has value 0, output it as `0 1`.

The following is a picture of the first sample:



Sample Input 1

```
1 1 3
```

Sample Output 1

```
-1 1 0 1
```

Sample Input 2

```
1 7 4
```

Sample Output 2

```
-1 1 6 7
```

Sample Input 3

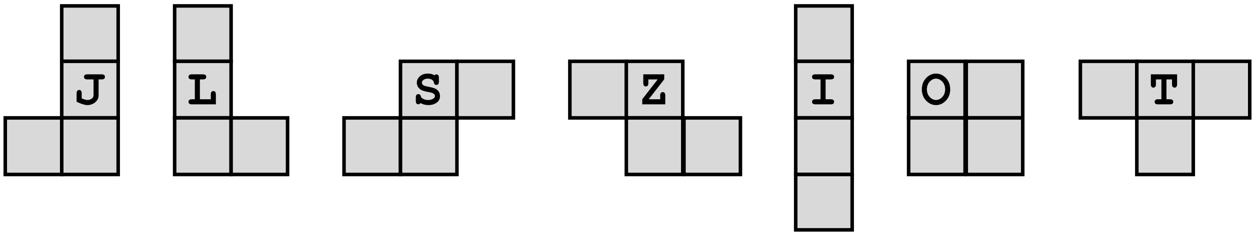
```
355 113 123456
```

Sample Output 3

```
1 1 -58 113
```

Problem I
Tetris Generation
Time Limit: 1 Second(s)

The classic game Tetris involves arranging falling tetrominoes on a board. There are seven different tetrominoes, each named after a letter that resembles their shape: `J`, `L`, `S`, `Z`, `I`, `O`, and `T`.



In the original Tetris, the player would receive one tetromino at a time, and each tetromino would be chosen from among the seven possibilities independently and uniformly at random. This meant that any sequence of tetrominoes could appear in a game, such as numerous `I` tetrominoes in a row. Modern versions of Tetris remove these streaks by generating tetrominoes in groups of seven: The first seven tetrominoes in a game will be one of each of the seven different tetrominoes in a random order. The next seven tetrominoes will also be one of each of the seven different tetrominoes in a random order (possibly but not necessarily different from the ordering of the first seven). Same goes for the next seven, and so on and so forth. With this generator, it is still possible to get two of the same tetromino in a row (for example, the seventh and eighth tetrominoes in the game can be the same as each other), but it is not possible to get three of the same type in a row.

Given a sequence of tetrominoes, determine whether it is possible for a modern Tetris generator to produce that sequence at some point in a game.

Input

The first line of input contains an integer t ($1 \le t \le 10^5$), which is the number of test cases.

Each of the next t lines contains a single string s ($1 \le |s| \le 1{,}000$, $s \in \{\texttt{J}, \texttt{L}, \texttt{S}, \texttt{Z}, \texttt{I}, \texttt{O}, \texttt{T}\}^*$). This string represents a sequence of tetrominoes, and is a single test case.

The sum of the lengths of all input test cases will not exceed 10^5.

Output

For each test case, output a single line with a single integer, which is `1` if the sequence can be generated by a modern Tetris generator, and `0` otherwise.

Sample Input 1

```
2
JJTO
JJTT
```

Sample Output 1

```
1
0
```

Problem J
Tournament Seeding
Time Limit: 2 Second(s)

You are tasked with seeding a single-elimination tournament for a one-on-one game. The number of players who have registered for the tournament is exactly a power of two, and there will be exactly enough rounds in this tournament to decide a winner. Furthermore, each player has a unique numeric rating in the game known to you; when two players play against each other in a game, the player with the higher rating always wins. As the organizer of the tournament, you would like to make the tournament as exciting for players and spectators as possible. To do that, you wish the tournament to have the following properties:

- The top two (highest rated) players are present in the final round of the tournament, the top four players are present in the semi-final round of the tournament, the top eight players are present in the quarter-final round, and so on. This saves the highest rated games for last.
- Subject to the above, as many games as possible are "close." We define a game to be "close" if the difference between the two players' ratings is less than or equal to some threshold.

Given the number of rounds, the threshold for "close" games and the ratings of the players, what is the maximum number of "close" games that can happen subject to the above constraints?

Input

The first line of input contains two integers n ($1 \leq n \leq 18$) and k ($1 \leq k \leq 10^9$), where n is the number of rounds of the tournament, and k is the rating difference that makes a game "close."

Each of the next 2^n lines contains a single integer r ($1 \leq r \leq 10^9$) denoting the rating of each player. The ratings are guaranteed to be distinct.

Output

Output a single line with a single integer, which is the maximum number of "close" games possible in a tournament among these players satisfying the constraints described above.

Sample Input 1

```
2 2
9
1
6
4
```

Sample Output 1

```
1
```

Sample Input 2

```
2 5
9
1
6
4
```

Sample Output 2

```
3
```

Problem K
Tree Hopping
Time Limit: 1 Second(s)

You are given a tree and a permutation of its vertices. It can be proven that for any tree and any pair of source/destination nodes, there is some permutation of the nodes where the first node is the source, the last node is the destination, and the distance between adjacent nodes in the permutation is less than or equal to three.

Your job will be to write a verifier for this property. Given such a permutation and the tree, validate whether the distance between adjacent nodes in the permutation is less than or equal to three.

Input

The first line of input contains an integer t ($1 \leq t \leq 50{,}000$), which is the number of test cases.

In each test case, the first line of input contains an integer n ($2 \leq n \leq 100{,}000$), which is the number of nodes in the tree. The nodes are numbered from 1 to n.

Each of the next $n - 1$ lines contains a pair of integers a and b ($1 \leq a < b \leq n$), representing an edge in the tree between nodes a and b.

Each of the next n lines contains an integer p ($1 \leq p \leq n$, all values distinct). This is the permutation of the nodes.

The sum of the values of n over all test cases will not exceed $100{,}000$.

Output

For each test case, output a single line with a single integer, which is 1 if the given permutation satisfies the constraint that every pair of adjacent nodes in the permutation has distance less than or equal to three in the tree. Output 0 if the given permutation does not satisfy this constraint.

Sample Input 1

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

Sample Output 1

```
1
0
```

Problem L
Who Goes There?
Time Limit: 1 Second(s)

What happens when more teams want to go to an ICPC regional site than the site has capacity for? Who goes there?

One possible policy is the following: Every school is allowed to register as many teams as they wish. Accept every school's first team, then accept every school's second team (for schools with more than one team), then third, and so on, until all teams are accepted, or there isn't enough capacity for the next wave. Then, if there are extra spots available, the spots are given to schools, one by one, in the order that the schools registered.

Given the capacity of a site, the number of teams registered by each school and the order that they registered, determine how many teams from each school are accepted.

Input

The first line of input contains two integers n ($1 \leq n \leq 100$) and m ($1 \leq m \leq 100$), where n is the capacity of the site and m is the number of schools that wish to compete there.

Each of the next m lines contains an integer t ($1 \leq t \leq 100$), which is the number of teams that a school has registered. The schools are listed in the order that they registered.

Output

Output m lines, one for each school. Each line must contain a single integer indicating the number of teams accepted from that school. Output them in the same order as they appear in the input.

Sample Input 1

```
20 5
7
5
1
6
12
```

Sample Output 1

```
5
5
1
5
4
```

This page is intentionally left blank.