Problem A. Arcane Accumulation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

A line of n wizards each has an integer magical power: positive for pure magic, negative for dark magic. The **magical potential** of a group of wizards is the sum of the magical powers of each wizard in that group.

To increase their potential, the wizards can use a special ability called the **Cloning Spell**, but it has strict rules:

- Any wizard can cast the spell, and it can be cast at most k times in total.
- When the i-th wizard (1-based index) casts the spell:
  - All wizards to their left (positions 1 through $i-1$) are **cloned in order**.
  - These clones are inserted **immediately before** the i-th wizard.
  - The original wizard and everyone after them are pushed to the right.
- The clone of a wizard possesses the exact same magical power.
- However, if any of the cloned wizards were themselves clones, their clones (also known as meta-clones) disappear immediately. Only wizards who were part of the original lineup and their direct clones survive. The lineup is adjusted accordingly — the remaining wizards shift left to fill any gaps left by disappearing clones.

The spell can be cast on any wizard (including ones who were cloned, as long as they are not clones of clones), and it can be used multiple times on the same wizard if desired.

Your task is to determine the **maximum possible magical potential** of the final lineup after performing at most k cloning spells optimally.

Input

The first line contains an integer t ($1 \le t \le 10^4$)— the number of test cases. Then t test cases follow.

The first line of each test case contains two space-separated integers n ($1 \le n \le 10^5$) and k ($0 \le k \le 10^9$) — the number of wizards and the maximum number of times the Cloning Spell can be cast.

The second line of each test case contains n integers: $a_1, a_2, \ldots, a_n$ ($-10^3 \le a_i \le 10^3$) — the magical powers of the initial wizards from left to right.

It is guaranteed that the sum of n across all test cases does not exceed 2×10^5.

Output

For each test case, print a single integer in a line — the **maximum possible magical potential** the wizards can achieve after casting the spell optimally.

Example

standard input	standard output
4 3 1 2 3 0 4 10 -2 -3 -1 -5 5 3 10 -2 -1 3 -4 5 1000000000 0 0 0 0 0	10 -11 36 0

Note

In the first test case, the rightmost wizard can cast the spell. After this, their magical potential will be $2+3+2+3+0=10$. It can be proved that the wizards cannot achieve more magical potential than this.

In the second test case, all the wizards possess dark magic, so casting the Cloning Spell will make the magical potential worse. So, it is optimal to not cast the spell even once.

Problem B. Balloon

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Shongshoptok, an ancient cult, developed their own cryptic language to communicate secretly among members.

In their language, they encrypt every sentence using a specific rule. Given any sentence in English, they reverse the order of all consonants in the sentence while keeping all vowels and spaces in their original positions.

For example, consider the sentence `NAVID ALVEE`. The consonants in the sentence are N, V, D, L, V. Reversing their order gives V, L, D, V, N. Replacing the original consonants with these reversed ones—while keeping all vowels and spaces untouched—results in the encrypted sentence `VALID AVNEE`.

A member of the *Shongshoptok* cult has sent your friend the following encrypted message:

`NILE LE A BAMVOOG`

Your friend is unable to figure out what it means, but he promises you a shiny reward if you can decrypt the message for him.

Write a program that prints the decrypted version of the message.

Note that this problem has no input. You are only required to output the decrypted version of the given sentence. Avoid printing any extra spaces or newlines. There is only one test case for this problem.

Input

This problem has no input

Output

Output exactly one line, the decrypted version of the sentence `NILE LE A BAMVOOG`. Avoid printing any extra spaces or newlines.

Note

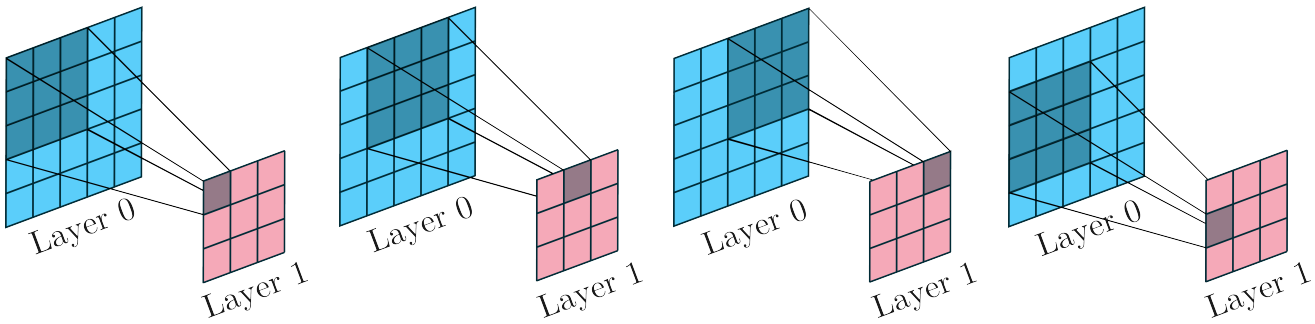
In English, the characters A, E, I, O, U are called vowels.

All other characters are called consonants.

Problem C. You Can't See Me

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

In digital image processing, an image is a 2D grid of pixels, each holding color or intensity values. A common operation on images is convolution, where a small square grid called a kernel slides across the image. At each step, the kernel looks at a small region of the image and uses it to calculate one pixel in the output.



The diagram above show how a 3×3 kernel slides across an image to produce the next layer (Layer 1) given an input image (Layer 0). Each output pixel in Layer 1 corresponds to one position of the kernel and is affected by the input region it covers.

Each convolutional layer applies this operation to the output of the previous layer, allowing the network to gradually combine information from larger regions of the input image. The size of the region in the original image that influences a pixel in the final layer is called its **receptive field** (or, how much a network can "*see*".)

You are given a network with n convolutional layers. Each layer uses a square kernel with an odd side length (e.g., 3, 5, 7, etc.).

Your task is to compute the size of the receptive field in the original input image for a pixel in the final layer. In other words, determine the side length of the square region in the input image that affects the value of a single output pixel in the n-th layer.

Input

The first line contains a single integer t ($1 \le t \le 2 \times 10^5$) — the number of test cases. Then t test cases follow.

The first line of each test case contains a single integer n ($1 \le n \le 2 \times 10^5$) — the number of layers in the network.

The second line of each test case contains n space-separated *odd* integers $k_1, k_2, \ldots, k_n$ ($1 \le k_i \le 10^9 - 1$) — the lengths of the kernels applied at each layer.

It is guaranteed that the sum of n over all test cases doesn't exceed 2×10^5.

Output

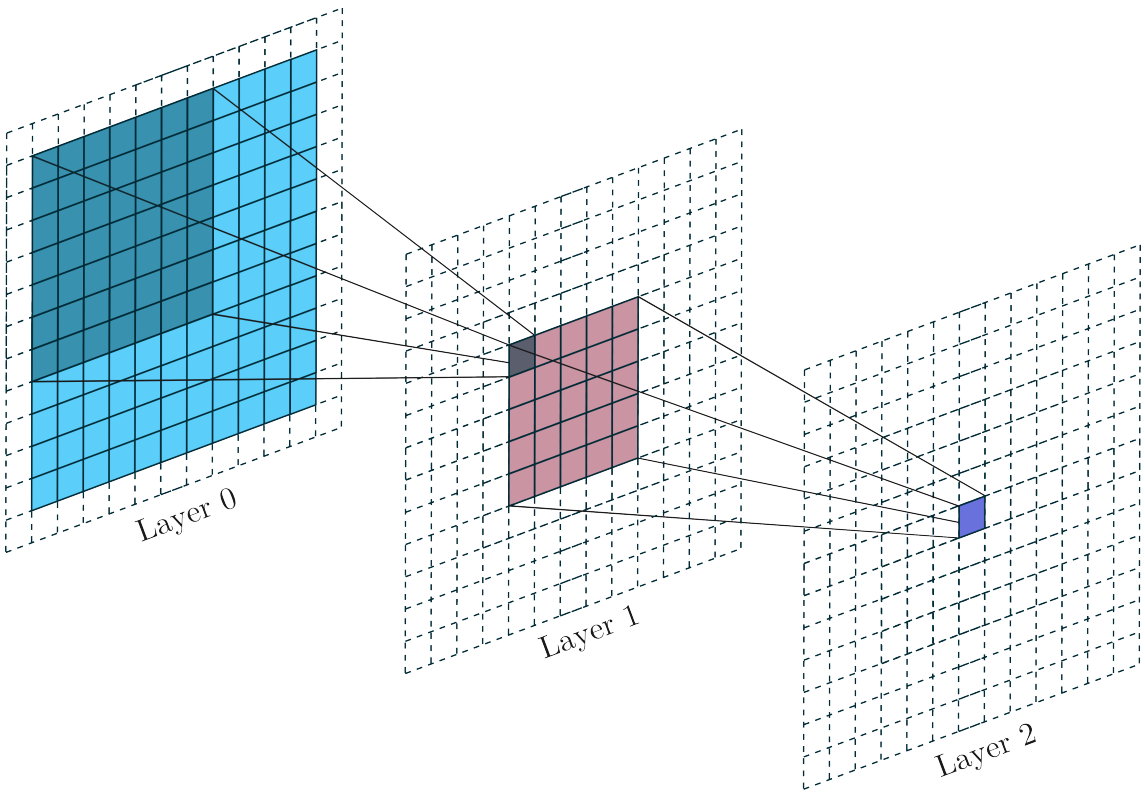
For each test case, output a single integer in a line — the length of the receptive field of a single pixel in the final layer.

Example

standard input	standard output
6 2 7 5 2 3 3 5 3 5 3 9 5 3 1 1 1 1 5 10 999999999 999999999 999999999 999999999	11 5 21 1 5 9999999981

Note

The network in the first test case is illustrated on the following diagram:



In the fourth test case, the network only uses 1×1 kernels. So, each pixel of the final layer is only affected by one pixel (the pixel with the same position) of the input image.

In the fifth test case, the network has only one layer. So, the receptive field is the same as the kernel size.

Problem D. Arise

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Sung Jinwoo, the Shadow Monarch, commands an army of fearsome shadow soldiers. In battle, he can raise the dead and summon any of his soldiers countless times using his mana. After closing a dungeon gate and defeating the boss, Jinwoo often reviews the battlefield log — a magical 2D grid that records traces of every being that participated in the battle.

Each shadow soldier bears a unique name, and when summoned, their name appears in a straight line in the battlefield log. A soldier's name may appear multiple times in the log, as they can be summoned repeatedly. Names are written in straight lines — horizontally, vertically, or diagonally, in any direction.

Now, given a list of Jinwoo's shadow soldiers and the battlefield log, your task is to determine how many **distinct soldiers** were summoned in the battle by checking if their names appear anywhere in the grid.

Input

The first line contains a single integer t ($1 \le t \le 10$) — the number of test cases.

The first line of each test case contains two integers n and m ($1 \le n, m \le 1000$) — the number of rows and columns in the battlefield log respectively.

The next n lines each contain a string of m uppercase English letters — representing the battlefield log.

The next line contains an integer q ($1 \le q \le 1000$) — the number of shadow soldiers.

The following q lines each contain a non-empty string of uppercase English letters — the names of each of Jinwoo's soldiers. It is guaranteed that the soldier names are unique.

It is guaranteed that the sum of $n \cdot m$ over all test cases does not exceed 1000 and the total number of characters across all soldier names in all test cases does not exceed 1000.

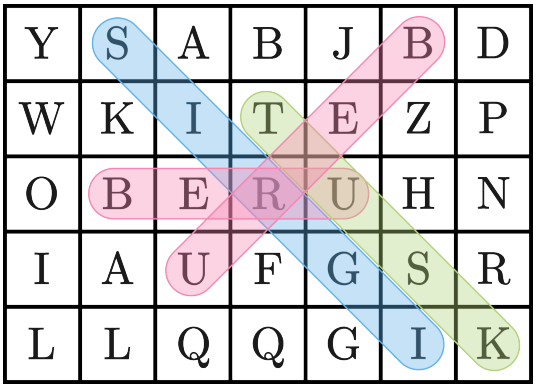
Output

For each query, output the number of soldiers that Jinwoo summoned during the battle.

Example

standard input	standard output
3 5 7 YSABJBD WKITEZP OBERUHN IAUFGSR LLQQGIK 5 BERU BELLION IGRIS KAMISH TUSK 3 4 ABCD EFGH PQRS 6 ABC BFQ DEF HGF PEF RFA 1 3 AAB 4 AA AB AAB BAA	3 4 4

Note



In the first test case, IGRIS, TUSK, and BERU were summoned. So, the answer is 3.

In the second test case, DEF and PEF do not appear in a straight line in any direction.

In the last scenario, all of the soldiers were summoned.

Problem E. Eid Salami

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Eid is here, and Zunaid is excited to visit his uncles and aunts to celebrate.

Zunaid has recently graduated from IUT with a brilliant result and landed a high-paying job. Naturally, his cousins are eager to receive salami from him. They are so enthusiastic that whenever he'll visit their house, they will immediately ask for salami before he even gets a chance to receive his salami from his uncle or aunt.

Zunaid has n uncles, and he can visit their houses in any order. At the i-th uncle's house, he must first give g_i taka to his cousins before receiving r_i taka from his uncle or aunt.

However, there's a problem- Zunaid is broke! His company has delayed his salary and bonus, leaving him with zero taka before Eid. Since he is the eldest cousin and known as a brilliant engineer, it would be embarrassing if he ever lacked the money to give salami when required.

To avoid this, Zunaid decides to take a loan from his friend before starting his journey. He will carefully plan his visits in an optimal way to minimize the loan he needs. Determine the minimum amount of loan Zunaid must take so that there exists at least one valid order in which he can visit all his uncles' houses without ever facing embarrassment.

Input

The first line contains a single integer t ($1 \le t \le 2 \times 10^5$) — the number of test cases. Then t test cases follow.

The first line of each test case contains a single integer n ($1 \le n \le 2 \times 10^5$) — the number of uncles Zunaid plans to visit.

The second line contains n space-separated integers $g_1, g_2, \ldots, g_n$ ($0 \le g_i \le 10^9$) — the amount of money in taka that Zunaid must give at each uncle's house.

The third line contains n space-separated integers $r_1, r_2, \ldots, r_n$ ($0 \le r_i \le 10^9$) — the amount of money in taka that Zunaid will receive at each uncle's house.

It is guaranteed that the sum of n over all test cases doesn't exceed 2×10^5.

Output

For each test case, print a single integer in a line — the minimum loan Zunaid needs to take before starting his journey.

Examples

standard input	standard output
6 2 200 100 250 220 3 100 0 300 200 200 400 3 100 300 0 350 400 50 4 100 200 300 400 50 1000 1500 2000 4 150 100 200 250 180 150 300 500 4 700 300 800 200 600 200 500 100	100 0 50 200 120 900
6 1 50 100 3 1000 2000 3000 1000 2000 3000 5 1000000000 1000000000 1000000000 1000000000 1000000000 0 0 0 0 0 5 0 0 0 0 0 1000000000 1000000000 1000000000 1000000000 1000000000 5 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 1000000000 5 0 0 0 0 0 0 0 0 0 0	50 3000 5000000000 0 1000000000 0

Note

In the first test case, Zunaid has two options: visit the first house first or visit the second house first. If Zunaid visits the first house first, he needs to take a loan of 200 taka. After he gives 200 taka salami to his cousin(s) in the first house, he will have 0 taka remaining. Then he will receive 250 taka from his uncle or aunt and have 250 taka with him. After that, he will visit the second house. He'll give 100 taka to his cousin(s) and have 150 taka with him. Then he will receive 220 taka and have 370 taka with him. On the other hand, if he visits the second house first, he needs to take a loan of 100 taka. He will go to the second house, give 100 taka salami and have 0 taka with him. Then he will receive 220 taka salami and have 220 taka with him. After that, he will go to the first house, give 200 taka and have 20 remaining. Then he will receive 250 and have 270 taka with him. Since Zunaid wants to minimize the loan, he will

visit the second house first, taking a loan of 100 taka.

In the second test case, Zunaid can visit his uncles' houses in the order $[2, 1, 3]$ without taking any loan.

In the third test case, the optimal order of visit is $[3, 1, 2]$, which requires a loan of 50 taka.

In the fourth test case, Zunaid needs a loan of 200 taka to visit the second house first. After that, he can visit the rest of the houses in any order.

Problem F. Inverted Queries

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Noor and Kabir are playing with an array of n numbers, A. Noor gives Kabir a new challenge.

For multiple queries, Noor will give Kabir a range in the form of $[l, r]$. Kabir must find the **maximum absolute difference** between any two **different** elements **outside** the subarray $A_l, A_{l+1}, \ldots, A_r$.

Unlike other challenges Noor gave him, Kabir finds this one tricky. Thus, he turns to you for help.

Note that, two elements are considered different if and only if they have different positions in the array. They might not necessarily be of different values.

Input

The first line contains a single integer t ($1 \le t \le 10^5$) — the number of test cases. Then t test cases follow.

The first line of each test case contains two integers n and q ($2 \le n \le 10^5$, $1 \le q \le 10^5$) — the size of the array and the number of queries.

The second line of each test case contains n integers $A_1, A_2, \ldots, A_n$ ($-10^9 \le A_i \le 10^9$) — the elements of the array.

Each of the next q lines contains two integers l and r ($1 \le l \le r \le n$) — the boundaries of the range to exclude.

It is guaranteed that the sum of n and the sum of q over all test cases does not exceed 3×10^5.

Output

For each query, output a single integer in a line — the maximum absolute difference between any two different elements not in the range $[l, r]$. If there are fewer than 2 elements outside the range, print -1.

Example

standard input	standard output
1 6 4 5 1 8 4 0 5 3 4 1 6 2 2 2 5	5 -1 8 0

Note

There is 1 test case. The array is $A = [5, 1, 8, 4, 0, 5]$ with $n = 6$, and there are $q = 4$ queries.

- **Query 1:** $[3,\ 4]$ — Exclude A_3 to A_4 ($[8,\ 4]$), remaining: $[5, 1, 0, 5]$. Max absolute difference is $|5 - 0| = 5$.

- **Query 2:** $[1,\ 6]$ — Entire array excluded. No elements remain, so the answer is -1.

Problem G. Trouble in the North

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The land of 404 North is ruled by four lords — Shabab, Hamim, Rafi, and Abdullah. Tensions among the lords have been rising, and the land is in disarray.

Each lord is too proud to reveal their own influence on the land. Instead, when asked, they will each only report a dissatisfaction value. You surmise that the dissatisfaction value reported by each lord is equal to the average influence of the other three lords. You, the chosen one, are the only person who can restore harmony in 404 North.

Your task, as the mediator of the land, is to figure out the actual influence value of each lord.

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of scenarios you must resolve.

Each of the next t lines contains four space-separated integers a, b, c, and d ($0 \le a, b, c, d \le 10^8$) — the dissatisfaction values reported by the four lords: Shabab, Hamim, Rafi, and Abdullah, in that order.

Output

For each scenario, output a single line with four space-separated integers — the actual influence values of Shabab, Hamim, Rafi, and Abdullah, in the same order.

Example

standard input	standard output
4 10 10 10 10 3 3 3 0 2 2 2 5 7 6 5 4	10 10 10 10 0 0 0 9 5 5 5 -4 1 4 7 10

Note

In the first scenario, each lord has an influence of 10.

In the third scenario, Abdullah's influence is negative. This means he has lost all influence and is controlled by the other lords.

Problem H. Hex Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

This is an interactive problem.

You are playing a turn-based game on a **regular hexagonal board**. The board has a side length l, and has vertices at the following points on a rectangular 2D coordinate sytem: $(0, 0)$, $\left(\frac{\sqrt{3}l}{2}, \frac{l}{2}\right)$, $\left(\frac{\sqrt{3}l}{2}, \frac{3l}{2}\right)$, $(0, 2l)$, $\left(-\frac{\sqrt{3}l}{2}, \frac{3l}{2}\right)$, $\left(-\frac{\sqrt{3}l}{2}, \frac{l}{2}\right)$.

Each player takes turns placing a **circular disk** of radius r on the board. The disk's center must be placed on a **lattice point**, i.e., a point (x, y) where both x and y are integers.

A move is **invalid** if:

- Any portion of the disk lies outside the hexagonal board, or
- The disk **strictly overlaps** with any previously placed disk.

The players alternate turns, with **you playing first**. The first player to make an invalid move **loses** the game.

Your task is to interact with the judge and play the game such that you **guarantee a win** regardless of your opponent's strategy.

Input

The first line contains a single integer t ($1 \le t \le 100$) — the number of games to be played.

Each test consists of two space-separated integers in a line l ($2 \le l \le 50$) and r ($1 \le r \le \frac{l}{2}$) — the side length of the hexagon and the radius of each disk.

Interaction Protocol

The interaction for each test case begins by reading the integers l and r.

After that, the game begins, and you move first.

For each of your moves, output two integers x and y in a line of the form: `P x y`

where `P` is a flag character indicating your move, and (x, y) are the coordinates of the center of the circular disk you wish to place. The point (x, y) must be a lattice point.

After each of your moves, the judge will respond with: `S r`

where `S` is a flag character indicating a status message, and r is:

- `1` if your move was valid.
- `-1` if your move was invalid. In this case, you lose and the game ends.

If your move was valid, the judge will then:

- Play its move and print: `J x y`

  where `J` indicates a judge move, and (x, y) is the center of the judge's disk.
- Then print: `S r`

  where r is:

- `1` if the judge's move was valid.
- `-1` if the judge's move was invalid. In this case, you win and the game ends.

You must continue playing until either you or the judge makes an invalid move. It can be proven that **the game must end** after a finite number of turns.

Upon receiving confirmation that you lost a game, your program should immediately terminate to receive the verdict `Wrong Answer`. Otherwise, it may receive a verdict that can be anything but `Accepted`.

Note that the flags are **case sensitive**.

After printing a command, do not forget to output the end of the line and flush the output. Otherwise, you will get `Idleness Limit Exceeded`. To do this, you can use:

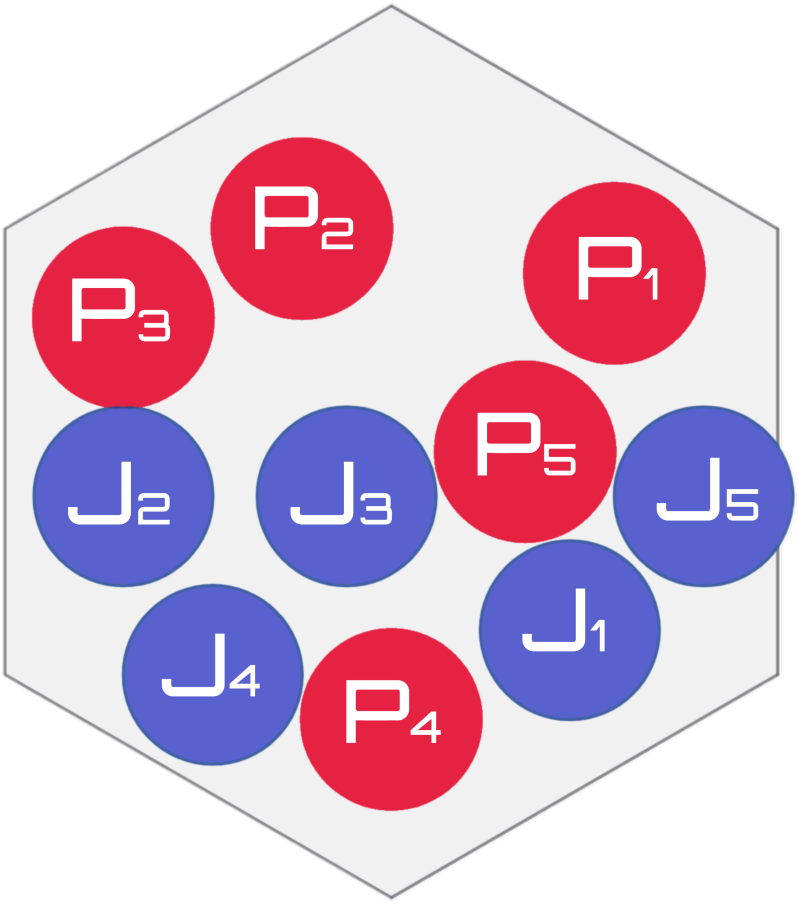
- `fflush(stdout)` in C;
- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `sys.stdout.flush()` in Python;
- `std::io::stdout().flush()` in Rust;
- Appropriate `flush` operation in any language.

Example

standard input	standard output
2 10 2 S 1 J 4 6 S 1 S 1 J -6 9 S 1 S 1 J -1 9 S 1 S 1 J -4 5 S 1 S 1 J 7 9 S -1 8 4 S 1 J 0 10 S -1	 P 5 14 P -2 15 P -6 13 P 0 4 P 3 10 P 2 6

Note

The game in the first test case is illustrated below. Here, the i-th disk placed by the player is marked P_i and the i-th disk placed by the judge is marked J_i.



It can be proven that any valid first move guarantees victory in the second game.

Note that the empty lines in the example input and output are for the sake of clarity, and may not occur in the real interaction.