TJIOI 2024 Problem Set

Do not open this packet until instructed to do so.

Problems are written by:
Avnith Vijayram
Gabriel Xu
Johnny Liu
Daniel Qiu
Samarth Bhargav
Munir Kudrati-Plummer
Avni Garg
Vishal Nandakumar
Marina Lin
Lalit Boyapati
Dhruv Kalsi

Problems are tested by:
Danny Mittal
Kevin Shan
Patrick Zhang

Problem A. Cookie Eating I

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

Cookie Monster and Elmo are playing a game where they each take turns eating cookies. They have c cookies in total. On a player's turn, they can either eat a single cookie, or eat half of the remaining cookies. If the remaining cookies happens to be odd, eating half of the cookies is equivalent to eating $\frac{c+1}{2}$ cookies.

Cookie Monster goes first and the game ends when there are no cookies left. The person who eats the last cookie wins. Output who wins.

Input

The first line contains c ($1 \leq c \leq 10^{18}$) — the number of cookies.

Output

Output `Cookie Monster` if Cookie Monster wins. Otherwise output `Elmo`.

Examples

standard input	standard output
5	Cookie Monster
305843009213693952	Elmo

Note

In the first test case, Cookie Monster eats half of the cookies, turning c into 2. No matter what Elmo does now, there will be 1 cookie left. Then, Cookie Monster can simply eat the last cookie and win.

Problem B. Cookie Eating II

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

Cookie Monster and Elmo are playing yet another cookie-eating game. On his turn, Cookie Monster can eat n cookies while Elmo can eat m cookies. If there are less cookies remaining than the number of cookies the player is allowed to eat, they eat all of the cookies that are left.

The player who eats the last cookie is given the prestigious title of Cookie Connoisseur. If there are initially c cookies and Cookie Monster starts eating cookies first, who is the Cookie Connoisseur?

Input

The first line contains 3 integers n, m, and c ($1 \leq n, m, c \leq 10^9$).

Output

Output a single string — `Cookie Monster` if Cookie Monster will win and `Elmo` if Elmo will win and be named Cookie Connoisseur.

Examples

standard input	standard output
2 4 10	Elmo
1 2 3	Elmo
1 2 4	Cookie Monster
158260522 877914575 602436426	Elmo

Note

In the first test case, Cookie Monster will eat 2 cookies, Elmo will eat 4 cookies, and Cookie Monster will eat 2 cookies again. Since there are less than 4 cookies left, Elmo will eat all of them and become the Cookie Connoisseur.

Problem C. Cookie Eating III

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

Cookie Monster and his n friends are entering a cookie eating competition.

Cookie Monster is in charge of organizing the strategy, while his friends eat the cookies. His friends are seated in a line, and they each have a plate in front of them. His friends can only eat cookies on their own plates, and can eat up to 1 cookie per second. Cookie Monster does not eat any cookies himself.

Currently, his ith friend has a_i cookies on his plate. At this rate, Cookie Monster's team can finish in $max(a_1, a_2, ..., a_n)$ minutes.

However, the organizer has announced a twist for the competition. Cookie Monster's team will have to finish M additional cookies. It is Cookie Monster's job to assign these cookies to his friends.

Once every second, Cookie Monster can choose a contiguous range $[l, r]$ $(1 \leq l, r \leq n)$, and give 1 cookie to each friend in that range. This happens before the friends eat the cookies.

Cookie Monster wants to beat the world record, so he wants to find the minimum time that his team can finish. Please answer q independent queries, where for each query, you're provided m_i, the number of cookies additional cookies that have to be assigned.

Input

The first line contains integers n and q $(1 \leq n, q \leq 10^5)$ — the number of lines and the number of queries, respectively.

The second line contains $a_1, a_2, \ldots, a_n$ $(0 \leq a_i \leq 10^5)$ — the number of cookies that are currently in each plate.

The next q lines each contain 1 integer — the values of m_i $(0 \leq m_i \leq 10^9)$.

Output

Output q lines. The ith line should contain the answer to the ith query. That is, the minimum number of time to finish all cookies if m_i additional cookies have to be eaten.

Examples

standard input	standard output
5 5 1 2 5 1 4 1 5 9 10 100000000	5 5 5 6 20000004
3 3 1 1 1 0 2 4	1 2 3
10 10 1 7 0 0 5 3 7 4 3 4 0 20 21 22 100 500 1000000 10000000 100000000 500000000	7 7 7 8 15 55 100005 1000005 10000005 50000005

Note

For the first sample case, the following set of assignments works:

Query 1: Give 1 cookie at minute 1 to any friend besides friend 3. Suppose we give it to friend 2. Then the number of cookies each friend has is $[1, 3, 5, 1, 4]$. The following sequence show how many cookies each friend has as the competition progresses: $[1, 3, 5, 1, 4] \to [0, 2, 4, 0, 3] \to [0, 1, 3, 0, 2] \to [0, 0, 2, 0, 1] \to [0, 0, 1, 0, 0] \to [0, 0, 0, 0, 0]$

Query 2: Give cookies to friends in the ranges $[1, 2]$, $[4, 5]$, and $[1]$ in each of the first 3 minutes.

Query 3: Give cookies to friends in the ranges $[1, 2]$, $[4, 5]$, $[1, 2]$, $[1, 2]$, and $[1]$ in each of the first 5 minutes.

There may be other possible configurations to achieve the minimum time to finish all cookies besides the ones above.

Problem D. A + B = Cookie

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

Cookie Monster has $3n$ cookies, where the i-th cookie has i chocolate chips. However, instead of eating them all, he wants to try something he's never done before: go on a diet! Cookie Monster has come up with the following rules for his diet:

- He will eat exactly 3 cookies each day: two in the morning and one in the afternoon.
- To keep a balanced diet, on each day he should eat an equal number of chocolate chips in the morning and afternoon.

Cookie Monster does not want to go grocery shopping for more cookies, since he would be unable to resist the temptation of more cookies. Thus, he has asked you to come up with a diet plan where he eats all the $3n$ cookies he currently has over a period of n days.

Input

The first line of the input will contain t, the number of testcases ($1 \leq t \leq 4 \cdot 10^5$).

Each testcase will contain a single integer n ($1 \leq n \leq 4 \cdot 10^5$).

It is guaranteed that the sum of n over all testcases does not exceed $4 \cdot 10^5$.

Output

If it is impossible to create a diet plan that lasts n days, print "NO".

Otherwise, print "YES". On each of the next n lines, output values of a, b, and c – the cookies that Cookie Monster should eat on that day in the order that he should eat them.

Example

standard input	standard output
3 1 4 7	Yes 1 2 3 Yes 1 5 6 2 8 10 3 9 12 4 7 11 No

Problem E. Cookie Cutting

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

Cookie Monster has a x by y rectangular cookie cake. With his cookie cutter, he can cut the cake into smaller pieces. Each cut should be made in a straight line that is parallel to the sides of the rectangle, and should split the cake into 2 pieces.

The time it takes to make a cut is equal to the length of the cut. Find the minimum amount of time Cookie Monster needs to spend cutting in order to cut off a rectangular piece with dimensions a by b. The paper can be rotated, so a i by j rectangle is equivalent to a j by i rectangle.

Input

The first line will contain 4 integers x, y, a, and b ($1 \leq x, y, a, b \leq 10^9$).

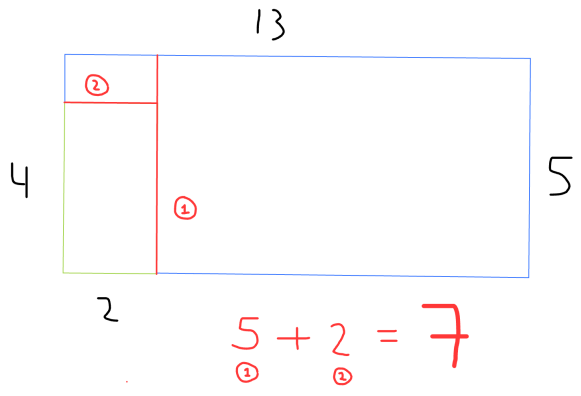
Output

If it is impossible to cut off a a by b rectangle from the original x by y rectangle, print -1. Otherwise, print the minimum amount of time Cookie Monster needs to spend cutting in order to do so.

Examples

standard input	standard output
13 5 4 2	7
5 7 6 5	5
5 7 6 6	-1
1000000000 1000000000 999999999 1	1000000001
1 3 3 1	0

Note



The first test case is illustrated above. The original 13 by 5 rectangle is drawn in blue, and the 4 by 2 rectangle to be cut out is in green. The red lines indicate the best choice of cuts to make, and cut 1 is made before cut 2.

In the second test case, only one cut is needed.

Problem F. Cookie Filling

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

Whoops! Cookie Monster messed up his cookie cake cutting.

His original cake could be represented by an n by m grid, with all squares being filled with cookie. However, after the cookie cutting accident, some locations on the grid have been cut out, and do not have cookie anymore. We can define $f_{ij} = 1$ if the j-th square on the i-th row is currently filled in with cake and $f_{ij} = 0$ otherwise.

Cookie Monster has extra rectangular pieces of cookie. To fill in the holes, he wants to figure out where the extra pieces could go.

Specifically, answer q independent queries. Each query consists of two parameters h and w representing a piece of cake with height h and width w. For each piece, find the number of locations where it can be placed in his cake such that it does not overlap with a square that already has cake. Pieces of cake cannot be rotated.

Input

The first line of the input will contain the integers n, m, and q ($1 \leq n \leq m \leq n \cdot m \leq 10^5$, $1 \leq q \leq 10^5$).

The next n lines will each contain m integers where the j-th integer on the i-th line is f_{ij} ($0 \leq f_{ij} \leq 1$).

The next q lines will each contain 2 integers h and w, the height and width of the cake in each query ($1 \leq h \leq n$, $1 \leq w \leq m$).

Output

Print q lines, the number of valid locations to place the cake in each query.

Examples

standard input	standard output
3 3 4 101 001 100 1 2 2 1 1 1 2 2	2 2 5 0
6 6 10 100010 110001 000011 000100 000100 100000 1 2 2 2 2 1 3 1 4 1 1 4 4 2 3 3 2 3 2 4	17 9 16 10 4 3 1 1 2 0

Note

In the first sample case, the 1 by 2 cookie cake piece can placed in 2 ways: on the locations $(2, 1), (2, 2)$ and $(3, 2), (3, 3)$. The 1 by 1 cookie cake piece can be placed in any of the 5 locations that aren't currently filled in with cake.

Problem G. Snacks

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

Cookie Monster currently has x chocolate chip cookies, y snickerdoodle cookies, and z macadamia cookies. He wants to eat as many good snacks as possible, where a good snack consists of exactly two cookies with different types.

However, Elmo stole and ate all of his macadamia cookies! Determine how many good snacks Cookie Monster can eat after having his macadamia cookies stolen.

Input

The first line will contain 3 integers x, y, and z ($1 \leq x, y, z \leq 10^9$).

Output

Output a single integer — the maximum number of good snacks Cookie Monster can eat.

Examples

standard input	standard output
1 2 2	1
10 10 100	10

Note

In the first sample test, after the 2 macadamia cookies are stolen, Cookie Monster has 1 chocolate chip cookie and 2 snickerdoodle cookies remaining, so he can eat 1 good snack.

Problem H. String Inside String

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

Cookie Monster's favorite word can be represented as a string B. He currently has a string A and wants to use it to create as many copies of string B as possible.

To do this, Cookie Monster will first rearrange the characters in A however he likes. Then, he will do the following: while string B is a prefix of A, keep removing that prefix. What is the maximum number of times he can remove the string B from A if he arranges the characters of A optimally?

Input

The first line of input contains the string A. The second line of input contains the string B. $(1 \leq |A|, |B| \leq 10^5)$, where $|A|$ indicates the length of string A. Strings A and B only contain lowercase alphabet characters.

Output

The maximum number of times Cookie Monster can remove string B from string A as a prefix.

Examples

standard input	standard output
avnith avni	1
xoinckseorzxoinckseorz cookie	2

Note

In this case, "avni"can fit inside of "avnith". There is no reordering required, as "avni"is a prefix of "avnith".

We can rearrange string A in the second test case to be “cookiecookiexnsrzxnsrz“, and we can remove “cookie“ as a prefix 2 times.

Problem I. Feeding Elmo

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

Elmo is picky eater, and has decided that he will only eat batches of m cookies where the number of chocolate chips on each cookie forms an arithmetic sequence[†] with common difference d.

Cookie Monster has n cookies, and the i-th cookie has c_i chocolate chips. Cookie Monster loves exotic cookies, so some of the cookies in his collection may have a negative number of chocolate chips.

Find the number of sequences that Cookie Monster can create to successfully feed Elmo by choosing m cookies. Of course, each cookie can only be used once. Two sequences are different if there is any position where the number of chocolate chips differs between the two cookies at that position in those sequences.

[†] An arithmetic sequence is a sequence of integers $b_1, b_2, \ldots, b_k$ which satisfy $b_2 - b_1 = b_3 - b_2 = \ldots = b_k - b_{k-1} = d$, where d is the common difference.

Input

The first line contains three integers n, d, and m ($1 \le m \le n \le 10^5, 1 \le d \le 10^9$).

The second line contains n space separated integers $c_1, c_2, \ldots, c_n$ ($-10^9 \le c_i \le 10^9$) — the number of chocolate chips on each cookie.

Output

Output a single integer — the number of different arithmetic sequences of chocolate chips that Cookie Monster can form to feed Elmo.

Examples

standard input	standard output
10 3 2 2 8 0 14 6 5 6 9 12 3	6
5 1 1 2 -1 0 3 5	5

Note

In the first example, the six sequences that can be formed are $\{0, 3\}$, $\{3, 6\}$, $\{6, 9\}$, $\{9, 12\}$, $\{2, 5\}$, and $\{5, 8\}$.

In the second example, the five sequences that can be formed are $\{2\}$, $\{-1\}$, $\{0\}$, $\{3\}$, and $\{5\}$.

Problem J. Tower Upgrades

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

There is a tree[†] with n nodes. A Cookie Monster with h health will travel through the tree, starting at node 1. Each node has a tower situated at it, initially with $p_i = 0$ power. The Cookie Monster takes p_i damage when visiting i which decreases its health by p_i. After taking damage, the Cookie Monster will arbitrarily choose to traverse an edge to an adjacent node which has not been visited yet.

If after the Cookie Monster takes damage at a node, it has positive health and has no valid moves (all edges at its current node lead to already visited nodes), it wins. You can perform the following operation as many times as you want: increase the p_i by 1 with a cost of c_i. Find the minimum cost needed to guarantee the Cookie Monster cannot win.

[†] A tree is an undirected graph in which there is exactly one simple path connecting any two vertices.

Input

The first line contains two integers n and h ($1 \le n \le 2 \cdot 10^5$, $1 \le h \le 10^6$).

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le 10^6$) — The cost to upgrade each tower.

The next $n - 1$ lines each contain two integers u and v ($1 \le u, v \le n$) — indicating an edge connecting nodes u and v.

It is guaranteed that the given graph forms a tree.

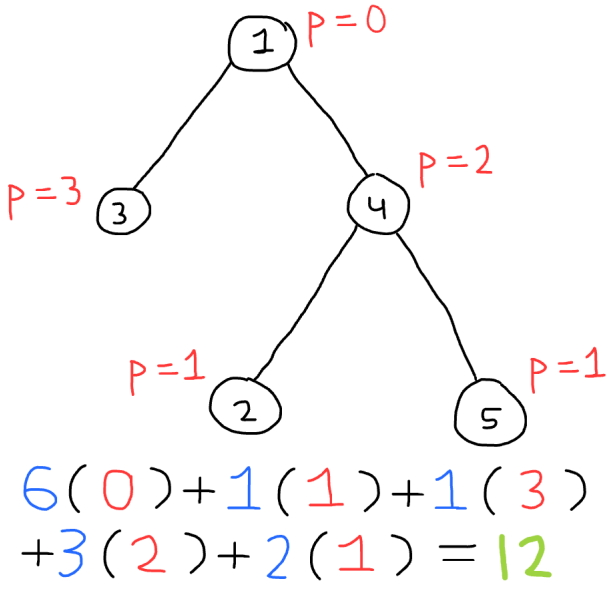
Output

Output a single integer — the minimum cost to upgrade the tower so that it is impossible for the Cookie Monster to win.

Examples

standard input	standard output
5 3 6 1 1 3 2 1 3 1 4 4 5 2 4	12
13 313494 58 96 47 34 79 41 72 48 60 8 62 58 77 9 2 13 4 4 9 4 12 11 2 5 7 8 5 6 4 2 1 2 3 1 10 10 7	18182652

Note



The diagram above illustrates the first test case. An optimal set of operations to perform is to upgrade tower 3 a total of 3 times, upgrade tower 4 twice, and upgrade towers 2 and 5 once each. This gives us a total cost of 12, which is the minimum possible. Note that there may be other valid sets of operations with the same minimum cost.

Problem K. Cookie Shuffle

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

Since Cookie Monster is quite clumsy and lacks dexterity, he struggles to execute the common cookie shuffling techniques. Instead, he has invented a rudimentary shuffling algorithm called the k-Stack Shuffle.

A deck is a stack of n cookies, where each of these cookies contains a unique number of chocolate chips in the range $[1, n]$. In other words, the number of chocolate chips in a deck of n cookies is a permutation of the integers from 1 to n. We will refer to the cookie with i chocolate chips as cookie i.

A k-Stack Shuffle with a deck of n cookies is as follows:

1. Initialize k empty stacks numbered from 1 to k.
2. Remove the cookie at the top of the deck and place it at the top of one of the k stacks. Repeat this step n times.
3. Take all the cookies in stack 1 and put them top of stack 2. Then, take the combined stack 1 and 2 and put it on top of stack 3. Repeat this process until you have one stack left. This stack is the new, shuffled deck.

Unfortunately, there have been several complaints involving Cookie Monster's new shuffling technique. In particular, the resulting shuffled deck is often very convenient for Cookie Monster, leading to accusations that he is not shuffling the deck randomly.

There are t particularly suspicious instances that suggest Cookie Monster may have been cheating. For each instance, the original stack was in sorted order (the cookies were labeled 1 to n from top to bottom) and the shuffled deck is given to you. Your task is to find the minimum possible k that could have been used in the k-Stack Shuffle. In other words, find the minimum k such that Cookie Monster could have used the k-Stack Shuffle technique to shuffle the sorted deck and obtain the given shuffled deck.

Input

The first line of the input will contain the integer t ($1 \leq t \leq 10^5$), the number of suspicious instances.

The first line of each test case will contain the integer n ($1 \leq n \leq 2 \cdot 10^5$), the size of the deck of cookies.

The second line of each test case will contain n integers $a_1 \dots a_n$ ($1 \leq a_i \leq n$, $a_i \neq a_j$ whenever $i \neq j$), the order of the cookies in the shuffled deck from top to bottom.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

Print t lines, the minimum k that Cookie Monster could have used in each instance.

Example

standard input	standard output
4 3 1 3 2 4 1 4 2 3 2 2 1 6 3 1 5 6 2 4	2 3 1 4

Note

In the second test case, Cookie Monster could have shuffled the deck with 3 stacks as follows.

1. Place cookie 1 in stack 1.
2. Place cookie 2 in stack 2.
3. Place cookie 3 in stack 3.
4. Place cookie 4 on top of stack 2.
5. Take stack 1 and place it on top of stack 2.
6. Take stack 2 and place it on top of stack 3.

It is impossible to perform this shuffle with fewer than 3 stacks.

Problem L. Windmill

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

Cookie Monster is solving a very cool IMO problem!! To better understand the problem, he wants to simulate it. Please help!!

You are given n points (x_i, y_i) ordered by x_i. Draw a vertical line through (x_2, y_2) and set (x_2, y_2) as the pivot. Then, repeat the following process indefinitely:

Rotate the line clockwise about the pivot until it intersects another point q. q becomes the pivot; the line now rotates about q.

Print the pivots in the order they are first encountered.

Input

The first line contains a single integer n ($3 \leq n \leq 10^5$).

Each of the next n lines contains two integers x_i and y_i ($-10^9 \leq x_i, y_i \leq 10^9$) — the coordinates of the i-th point.

It is guaranteed that $x_1 < x_2 < \ldots < x_n$ and all y_i are distinct. In addition, no 3 points are collinear.

Output

In the first line, output m — the number of distinct pivots encountered.

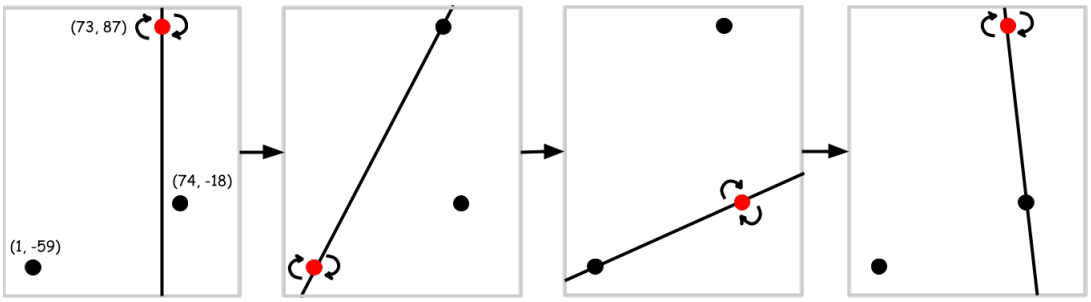
Output 2 integers on each the next m lines, where the j-th line contains integers x_j and y_j, the coordinates of the j-th encountered pivot.

Examples

standard input	standard output
3 1 -59 73 87 74 -18	3 73 87 1 -59 74 -18
4 -83 -40 -23 90 26 52 52 -68	4 -23 90 -83 -40 26 52 52 -68

Note

For sample case 1, the line rotates as follows:



Problem M. Telephone Fever Dream

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

Cookie Monster had a funny dream. His n cookies started talking! And the cookies were playing a game of telephone.

There are n cookies conveniently numbered $1, 2, \ldots, n$ that wish to participate. Each cookie i gets assigned a partner cookie c_i such that no two cookies have the same partner.

At time 0, every cookie i sends a unique message to its partner c_i. Every subsequent minute, each cookie i tells the next cookie c_i the message it just heard. This process continues indefinitely for every cookie until a cookie hears the original message it transmitted at time 0, at which point Cookie Monster will eat it.

Let's say that a cookie got eaten at time y. It turns out that a cookie will only be happy if it is eaten on some multiple of k minutes after time 0. In other words, if a cookie is eaten at time y, it will be happy if and only if k divides y.

Find the number of possible assignments $c_1, c_2, \ldots, c_n$ such that all the cookies are happy when they get eaten at the end. Since this number may be very large, output the result modulo $10^9 + 7$.

Input

The first line will contain two integers n and k ($1 \leq k \leq n \leq 10^5$).

Output

Output a single integer — the number of assignments $c_1, c_2, \ldots, c_n$ such that all cookies are happy modulo $10^9 + 7$.

Examples

standard input	standard output
4 2	9
8515 5	58417763
100000 5000	689458132

Note

In the first sample test case, the 9 valid assignments of $c_1, c_2, \ldots, c_n$ are listed below.

- $[2, 1, 4, 3]$
- $[3, 4, 1, 2]$
- $[4, 3, 2, 1]$
- $[2, 3, 4, 1]$
- $[2, 4, 1, 3]$
- $[3, 4, 2, 1]$
- $[3, 1, 4, 2]$
- $[4, 1, 2, 3]$
- $[4, 3, 1, 2]$

General Information

This packet contains the problem statements with the problems. Organizers are able to give clarifications but not help with any problem.

The only websites allowed are tjctgrader.org, ide.usaco.guide, desmos.com, and reference websites for the programming languages we provide. Note that GeeksForGeeks, Codeforces, OEIS, are not reference websites and are disallowed. In general, please follow common sense.

Please respect our grader and the TJHSST hardware. Malicious submissions and any attempts to mess with the TJHSST desktops will not be tolerated.

Each problem has a time limit of 2 seconds for C++, 3 seconds for Java, and 4 seconds for Python. Memory limit is 512 MB. Note that our grading servers are slower than USACO or Codeforces. Your solution should aim to use $<10^8$ operations given the largest constraints. Test cases are not guaranteed to be generated randomly.

When submitting problems, make sure you have selected the correct programming language. The only languages we accept are Python 3, Java, and C++ 14. Note that the versions of these languages on the machines vs. the grader might not be exactly the same.

You can view feedback for the first test case of each problem by clicking the link corresponding to your submission on the leftmost column of the "My Submissions" page. Further test cases are hidden and we won't be able to provide the inputs during the contest.

JAVA USERS PAY ATTENTION:

When submitting code, please add "package subcode;" and name your class "test".

Example:

```
package subcode;
import java.io.*;
import java.util.*;

public class test {
	public static void main(String[] args) throws IOException {
		Scanner s = new Scanner(System.in);
		int a = s.nextInt();
		int b = s.nextInt();
		System.out.println(a+b);
	}
}
```