The 2019 ICPC Caribbean National Contests

Real Contest Problem Set

Document composed by 12 pages (not including this cover)

Problem set developers

Marcelo Fornet (NEAR, Cuba)

Carlos Joa (INTEC, República Dominicana)

Carlos Toribio (Google, República Dominicana)

Elio Govea (UPR, Cuba)

José Carlos Gutiérrez (Universität Bremen, Cuba)

Rafael Fernández (Cuba)

Ariel Cruz (UH, Cuba)

Norge Vizcay (Cuba)

Daniel Otero (Universität Bremen, Cuba)

Leandro Castillo (Dropbox, USA)

Roberto Abreu (República Dominicana)

Aurora Gil (Cuba)

Reynaldo Gil (Datys, Cuba)

Dovier Ripoll (UCI, Cuba)

October 5th, 2019

Problem A. Guess the number

Randy está jugando un juego simple llamado "Adivina el Número": tenemos un número entero V cuyo valor está entre 1 y N. El objetivo del juego es adivinar ese número V. En cada ronda, Randy elige un número entero entre 1 y N. Si ese número coincide con V, el juego se detiene y Randy gana un premio. De lo contrario, el juego continua y le pide a Randy por otro número, y así sucesivamente.

En cada ronda, Randy utiliza la siguiente estrategia para adivinar el número:

1. El selecciona aleatoriamente un número entero x entre 1 y N.
2. Si x nunca fue usado en una ronda anterior, x sería el número a adivinar en esta ronda. Esto le cuesta \$1 a Randy.
3. De lo contrario, él elige el número entero no usado anteriormente más pequeño que exceda a x (y no exceda N). Esto tambien le cuesta \$1 a Randy.
4. Si dicho número no existe, he descarta el número x y vuelve al paso 1. Decartar el número x no tiene costo.

Pregunta: En promedio, cuánto dinero Randy tiene que pagar antes de ganarse el premio?

Input

La entrada consiste en una sola línea conteniendo dos enteros N ($1 \leq N \leq 20$) y V ($1 \leq V \leq N$).

Output

Imprime una sola línea con la respuesta a la pregunta propuesta en el enunciado. Tu respuesta se considera correcta si su diferencia con la respuesta esperada no excede 10^{-6}.

Example

standard input	standard output
2 2	1.5
3 2	1.944444444
10 6	5.305939329

Problem B. Which permutation

A Juan le encantan las permutaciones. Las ama tanto que, cada vez que encuentra una secuencia de números, él genera el conjunto de todas las posibles permutaciones de esa secuencia. Obviamente, el termina con un conjunto inmenso de permutaciones. No obstante, Juan desea ordenar este conjunto usando el algoritmo de Bucket Sort. Para lograr esto, Juan necesita saber, dada una permutación P, cuantas permutaciones son lexicográficamente más pequeñas que P.
Una vez que él aprenda a computar esto eficientemente, utilizará procesamiento en paralelo y será capaz de ordenar el gigantesco conjunto de permutaciones de manera casi instantánea. Por lo tanto, él te ha encargado completar esta importante tarea: dado una secuencia P de enteros (no necesariamente distintos), calcula cuantas permutaciones de esa secuencia existen que son lexicográficamente más pequeñas P.

Una permutation de una secuencia S es cualquier reordenamiento de los elementos de S.

Si P y Q son dos permutaciones de la misma secuencia de números, decimos que P es lexicográficamente más pequeña que Q si $P_i < Q_i$ donde i es la primera posición en la cual las permutaciones P y Q difieren. Por ejemplo, la permutación $(3, 2, 9, 2, 10)$ es lexicográficamente más pequeña que $(3, 2, 10, 2, 9)$.

En el primer caso de prueba, las seis permutaciones que son lexicográficamente más pequeñas que $(3, 1, 2, 3)$ son: $(1, 2, 3, 3)$, $(1, 3, 2, 3)$, $(1, 3, 3, 2)$, $(2, 1, 3, 3)$, $(2, 3, 1, 3)$ y $(2, 3, 3, 1)$.

Input

La primera línea contiene un entero N ($1 \leq N \leq 200,000$). La próxima línea contiene N números enteros positivos P_i ($1 \leq P_i \leq 10^9$) que representan la secuencia P.

Output

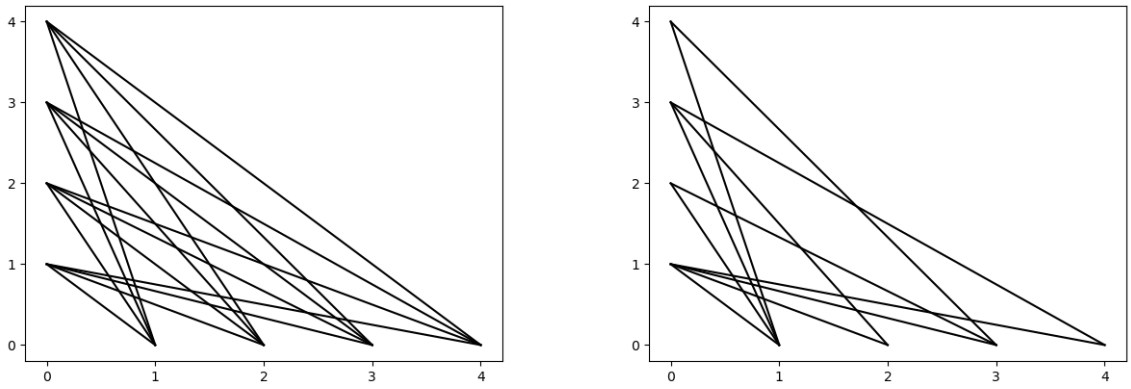
Imprime la cantidad de permutaciones de P que son lexicográficamente más pequeñas que P, módulo el primo 998244353.

Example

standard input	standard output
4 3 1 2 3	6
13 13 1 2 3 4 5 6 7 8 9 10 11 12	756797435

Problem C. Straight Lines

Teniendo todos los puntos con coordenadas enteras positivas no mayores que N sobre los ejes de coordenadas x, y disponibles, se quieren trazar rectas que pasen por dos de estos (siempre de ejes distintos) de tal manera que no existan rectas paralelas. El objetivo es trazar tantas como sea posible y calcular de cuántas formas se puede lograr esto.



Las imagenes corresponden al ejemplo donde $N = 4$:

1. La imagen de la izquierda muestra todas las rectas que pudieran ser dibujadas sin respetar la condicion de que no hayan paralelas.
2. La imagen de la derecha muestra una de las formas de trazar la mayor cantidad de rectas sin que hayan paralelas.

Input

Una línea con un entero positivo N ($1 \leq N \leq 10^9$).

Output

Dos enteros c y w, la mayor cantidad de rectas y la cantidad de formas de hacer esto respectivamente. Como la cantidad de formas puede ser muy grande, se pide calcular su resto de la división por el número primo 998244353

Example

standard input	standard output
1	1 1
4	11 16

Problem D. Bridge

Fito tiene una caja nueva de cubos con colores. Como está tan contento con sus nuevos cubos, Fito desea construir un puente situándolos todos en una fila, pero no de forma arbitraria. Para que la fila sea elegante, no deben haber dos cubos consecutivos del mismo color. Como si esto fuera poco, Fito ya decidió el color que deben tener los cubos en los extremos del puente. Ayuda a Fito a determinar una distribución de los cubos para construir el puente.

Input

Primero, una línea con un entero t que denota la cantidad de casos de prueba. Cada caso de prueba está formado por dos líneas. La primera tiene cuatro enteros n, k, p, q ($1 \leq n, k \leq 10^5$, $1 \leq p, q \leq k$) que denotan la cantidad de cubos, el total de colores y el color del primer cubo en la fila y el color del último cubo en la fila, respectivamente. La siguiente línea contiene n enteros a_i ($1 \leq a_i \leq k$) que denota el color de cada cubo. Se garantiza que la cantidad de cubos entre todos los casos de prueba no excederá 10^5.

Output

Por cada caso de prueba, imprime `"No"` (sin las comillas) si no hay ninguna forma de distribuir los cubos para construir el puente de acuerdo a las reglas de Fito, o `"Yes"` (sin las comillas) en caso contrario. En caso de que exista solución, imprime n enteros (los colores) en cualquier orden que constituya una respuesta válida. Cualquier respuesta válida será aceptada. **Nota que TODOS los cubos deben ser utilizados.**

Example

standard input	standard output
3 1 1 1 1 1 8 3 3 1 2 2 2 2 3 3 1 1 8 3 3 1 3 3 3 2 2 2 1 1	Yes 1 No Yes 3 2 3 2 3 1 2 1

Problem E. Xela

Xela es un nuevo lenguaje de programación diseñado para calcular operaciones algebraicas con ciclos anidados extremadamente rápido. Tu tarea es construir un intérprete de dicho lenguaje. Especificaciones

del lenguaje.

1. Hay 26 variables en *Xela*. Todas son letras minúsculas del inglés de la `a` a la `z`. Inicialmente todas las variables empiezan en 0.
2. Todas las operaciones algebraicas se hacen modulo el número primo 998244353.
3. Todos los escalares del código están en el intervalo $[0, 998244353)$.
4. *Asignación*: Instrucción que asigna a una variable un escalar o el valor de otra variable. Ejemplo $a = 42$ (Asigna 42 a la variable a) o $z = x$ (Asigna el valor de la variable x a la variable z).
5. *Adición*: Instrucción que suma a una variable un escalar o el valor de otra variable. Ejemplo $a \mathrel{+}= 31415$ (Suma el valor 31415 a la variable a) o $y \mathrel{+}= a$ (Suma el valor de la variable a en la variable y).
6. *Sustracción*: Instrucción que resta de una variable un escalar o el valor de otra variable. Example $a \mathrel{-}= 1$ (Resta 1 de la variable a) o $v \mathrel{-}= u$ (Resta el valor de la variable u de la variable v).
7. *Multiplicación*: Instrucción que multiplica a una variable el valor de un escalar *(multiplicar por otra variable no está permitido)*. Ejemplo $a \mathrel{*}= 2$ (Multiplica a por 2 y guarda el valor en a).
8. *Ciclos*: Bloque de código que contienen un parámetro repeat, que es un escalar en el rango $[1, 998244353)$ *(no está permitido usar variables como parámetro de repeat)*. El código contenido dentro de un bloque es ejecutado *repeat* veces. Todo el código dentro de un bloque es ejecutado *repeat* número de veces. Dentro de cada bloque hay siempre al menos una Instrucción. Múltiples instrucciones dentro del mismo bloque tienen la misma indentación.
9. Todas las líneas en el programa son instrucciones válidas o ciclos.

Dado un código escrito en *Xela* calcula el último valor de cada variable.

Input

Un programa de *Xela* válido con a lo sumo 100 líneas.

Output

Para cada variable con valor distinto de 0, imprime su valor en una línea separada con el formato "*variable* = *valor*" omitiendo las comillas. Las variables deben listarse en orden alfabético. Si todos los valores son iguales a 0 al final del programa, entonces imprima `"INITIAL STATE"` sin las comillas.

Example

standard input	standard output

a = 1
b = 1
loop 10
    c = a
    a += b
    b = c
    c = 0
	a = 144
	b = 89

a = 0
loop 100000000
    c = 1
    d = a
    a -= c
    loop 100000000
        c *= 2
        d += c
    d *= 3
    d -= 1
    a += d
	a = 916538251
	c = 113003797
	d = 108226118

x = 17 loop 23 x *= 2 x *= 7 x += 1	INITIAL STATE

Problem F. Dates

Rosita escribió en una libreta la fecha (día y mes) de k días consecutivos y guardó esta libreta en un cajón. Al cabo de unos años encuentra su libreta, pero lamentablemente los meses que escribió eran ilegibles. Rosita desea saber los posibles meses de la última fecha que escribió en esta libreta.

Input

Una línea con un entero k ($1 \leq k \leq 300$) denotando cuántas fechas fueron anotadas. La siguiente línea contiene k números enteros: los días de cada fecha en el mismo orden en que fueron anotados. Está garantizado que los días corresponden a una secuencia consecutiva valida de fechas.

Output

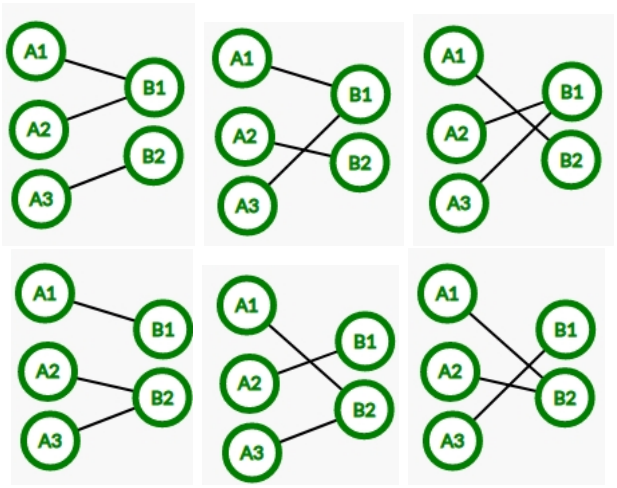
Imprima la cantidad de meses posibles de la última fecha que Rosita escribió en su libreta. En las siguientes líneas imprima los posibles meses, uno por línea, en el orden del calendario.

Example

standard input	standard output
10 1 2 3 4 5 6 7 8 9 10	12 January February March April May June July August September October November December
2 31 1	7 January February April June August September November
11 20 21 22 23 24 25 26 27 28 29 1	1 March

Problem G. How many

Dado dos conjuntos A y B que poseen n y m elementos respectivamente con $n \geq m$, diga de cuántas maneras es posible emparejar todos los elementos de forma tal que cada elemento de A esté emparejado con exactamente un elemento de B y cada elemento de B esté emparejado con al menos un elemento de A. Debajo estan todas las posibles asignaciones para el primer caso de ejemplo.



Input

La única línea de entrada consiste en dos enteros n ($1 \leq n \leq 10^{18}$) y m ($1 \leq m \leq 10^5$) separados por un espacio, la cantidad de elementos de A y B respectivamente.

Output

Imprime la cantidad de formas en las que se pueden emparejar los elementos de los conjuntos como se pide en el problema. Como este número puede ser muy grande, imprima el valor del resto de la división de dicho número con el número primo 998244353.

Example

standard input	standard output
3 2	6
10 3	55980
314159265358979323 84626	683366796

Problem H. Colors I

La diferencia entre este problema y "Colors II" esta en negrita.

Hay n puntos en el plano. Cada uno de los puntos tiene asignado un color, pero todos los puntos no tienen el mismo color. Determine la mayor distancia (Euclideana) entre algún par de puntos de diferente color. **Todos estos puntos están en una única recta.**

Input

La primera línea contiene un entero n ($2 \leq n \leq 250000$). Cada una de las siguientes n líneas que siguen describen un punto. La descripción de cada punto consiste en 3 enteros x_i, y_i, c_i ($-10^9 \leq x_i, y_i \leq 10^9, 1 \leq c_i \leq n$) separados por espacios; los números x_i, y_i son las coordenadas del i-ésimo punto, y c_i es el id del color del punto.

Hay al menos dos colores diferentes y no hay dos puntos con la misma coordenada.

Se garantiza que existe una línea recta, tal que todos los puntos están sobre esta línea.

Output

La primera y única línea de la salida debe ser un entero que represente el cuadrado de la máxima distancia entre dos pares de puntos con color diferentes.

Example

standard input	standard output
3 0 0 1 1 1 2 2 2 1	2

Problem I. Colors II

Hay n puntos en el plano. Cada uno de los puntos tiene asignado un color, pero todos los puntos no tienen el mismo color. Determine la mayor distancia (Euclideana) entre algún par de puntos de diferente color.

Input

La primera línea contiene un entero n ($2 \leq n \leq 250000$). Cada una de las siguientes n líneas que siguen describen un punto. La descripción de cada punto consiste en 3 enteros x_i, y_i, c_i ($-10^9 \leq x_i, y_i \leq 10^9, 1 \leq c_i \leq n$) separados por espacios; los números x_i, y_i son las coordenadas del i-ésimo punto, y c_i es el id del color del punto.

Hay al menos dos colores diferentes y no hay dos puntos con la misma coordenada.

Output

La primera y única línea de la salida debe ser un entero que represente el cuadrado de la máxima distancia entre dos pares de puntos con colores diferentes.

Example

standard input	standard output
3 0 0 1 1 1 2 2 2 1	2
4 0 0 1 0 1 2 1 0 3 1 1 4	2

Problem J. Anagram

Dos cadenas son anagramas entre sí si las letras de una cadena se pueden reorganizar para formar la otra cadena. Dada una cadena s, encuentre el número de pares de subcadenas de la cadena s que son anagramas entre sí.

Por ejemplo $s = wow$ la lista de todos los pares anagramáticos es $[w, w], [wo, ow]$ en las posiciones $([0..0], [2..2])$ y $([0..1], [1..2])$ respectivamente.

Input

Una sola línea que contiene una cadena s para analizar. La cadena s solo contiene caracteres del alfabeto ingles del rango $[a..z]$ y $|s| \leq 100$.

Output

Imprima el número de pares anagramáticos que existen en la cadena dada.

Example

standard input	standard output
abba	4
abcd	0

Problem K. White and black

Son dados n conjuntos con k números diferentes cada uno. Todos los pares de conjuntos son distintos (difieren en al menos un elemento). Cada conjunto se debe colorear de blanco o negro de forma tal que sea posible resolver el siguiente acertijo:

Una persona en secreto elige uno de los conjuntos y revela de forma arbitraria su color y $k-1$ números del mismo. Otra persona con estos datos debe ser capaz de inferir cual es el número que falta. La segunda persona conoce el color y contenido de cada conjunto inicial.

El problema consiste en determinar si existe alguna forma de colorear los conjuntos de forma tal que, independientemente de la acción de la primera persona, la segunda persona siempre pueda determinar el número restante.

Input

En la primera línea hay dos enteros n y k ($1 \leq n \leq 100, 1 \leq k \leq 10$), la cantidad de conjuntos y la cantidad de números por conjunto.

A continuación n líneas cada una con k números v_i ($1 \leq v_i \leq 100$) que denotan los elementos de cada conjunto.

Output

En la primera línea, en caso de que existe solución, imprime la palabra `"Yes"` (sin la comillas); en caso contrario, imprime la palabra `"No"` (sin las comillas).

Si hay solución, en la segunda línea imprima una cadena con n caracteres donde el i-ésimo caracter sea 'W' si el conjunto i se pinta de blanco y 'B' si se pinta de negro. Cualquier solución válida será aceptada.

Example

standard input	standard output
2 3 1 2 3 2 3 4	Yes WB
2 3 1 2 3 3 4 5	Yes WW
3 3 1 2 3 2 3 4 3 4 1	No