Universidad Mayor de San Andrés

ACM International Collegiate Programming Contest

2014

Competencia Interna de Segunda División

16 de agosto de 2014

Las páginas están numeradas desde el 1 al 7.

Índice

Problema A
LCA (Lowest Common Ancestor)

El ancestro común más bajo (LCA) es un concepto dentro de la teoría de grafos y Ciencias de la Computación. Sea T un árbol con raíz y n nodos. El ancestro común más bajo entre dos nodos v y w se define como el nodo más bajo en T que tiene a v y w como descendientes (donde se permite a un nodo ser descendiente de él mismo). El LCA de v y w en T es el ancestro compartido de v y w que está localizado más lejos de la raíz. Considere un árbol binario completo infinita (cada nodo tiene dos hijos, excepto los nodos hoja) definida como sigue. Para un nodo etiquetado v su hijo izquierdo se clasificará **2 * v** y su hijo derecho será etiquetado **2 * v + 1**. La raíz está etiquetado como 1. Se le dará varios casos de entrada cada caso consta de dos números v y w, $1 \leq v, w \leq (2^{63}) - 1$. Para cada consulta, hay que imprimir el LCA entre los nodos v y w del árbol binario.

Ejemplos de entrada	**Ejemplos de salida**
8 5 122 124 45 654	2 15 5

Problema B
Mercado Mágico

Alice trabaja en un súper mercado Mágico, que tiene N productos para vender y la cantidad de estos productos es infinita, pero tiene una regla poco usual a la hora que un cliente realice su compra, los gerentes decidieron que la forma de comprar será la siguiente: Un cliente solo puede elegir del único estante de productos un determinado rango, es decir, del producto i hasta el producto j. Esto para facilitar el cálculo del precio y no pasar cada producto uno a uno por el lector de código de barras. Aparte de esta forma poco usual de realizar compras, los gerentes cambian el precio de cada producto constantemente, es decir que en un momento dado, el total del precio de un rango (i, j) puede variar en el futuro. Un día Alice se cayo de la motocicleta y se olvidó sumar, es por eso que ella desea un programa que realice las siguientes operaciones:

- Obtener el precio total de un rango (i,j)
- Actualizar el precio del producto "i" a un valor "v"

Input

La entrada comienza con un número T($1 \leq T \leq 100$) que representa el número de casos de prueba, para cada caso la entrada consta de dos números N ($1 \leq N \leq 100000$) M ($1 \leq M \leq 100000$) que representan el número de productos en el mercado y el número de operaciones a realizar. La siguiente línea contiene N elementos que son los precios iniciales "v" $1 \leq v \leq 100$ de cada producto "i". Luego siguen M líneas con el siguiente formato: P i j, donde P es el comando para obtener el precio total del rango (i,j). o A i v, donde A es el comando para actualizar el producto "i" a un valor "v".

Output

Por cada comando P, mostrar el precio total para el rango (i,j) dado.

Ejemplos de entrada	**Ejemplos de salida**
1 4 9 37 14 23 78 P 2 2 A 2 74 A 1 75 A 2 52 P 2 3 P 1 2 P 2 2 A 4 67 A 3 32	14 75 127 52

Problema C
Defense Of The Ancients

Defense of the Ancients (DOTA) por sus siglas en ingles, es un juego que nunca pasara de moda, se juega entre 2 equipos de N personas, cada jugador elige a un héroe con 4 habilidades. DOTA causa un gran impacto en las personas por que algunas partidas son mas interesantes que otras, una partida es mas interesante que otra si existen mas batallas entre héroes, un héroe puede batallar contra otro solo si comparten al menos una habilidad. Por ejemplo si un héroe tiene habilidades "XTVS" y otro tiene "XCAS" entonces pueden tener una batalla, un héroe pierde y **muere para toda la partida** y el otro acaba tan herido que **no puede enfrentarse a otro**. Necesitamos saber si una partida va a ser interesante, pero para eso nos tienes que decir cuantas batallas se pueden realizar como máximo.

Input

La entrada consiste en un numero T casos de prueba ($1 \leq T \leq 1000000$). Seguido por T casos de prueba, cada caso de entrada contiene un numero N ($1 \leq N \leq 6$) el numero de jugadores por equipo, seguido por N cadenas que representan las 4 habilidades (caracteres de la "A" - "Z") de cada héroe del primer equipo y por ultimo N cadenas con las habilidades del segundo equipo.

Output

Imprimir una linea por cada caso de prueba que indica el máximo numero de batallas en la partida.

Ejemplos de entrada	**Ejemplos de salida**
1 3 ABCD XYTS ABCW BACD OPCW POIU	2

Problema D
El juego de los polígonos

Dos jugadores participan en el juego de los polígonos. Un polígono convexo con n vértices dividido por n-3 diagonales en n-2 triángulos es necesario. Estas diagonales solo pueden cruzarse en vértices del polígono. Uno de los triángulos es negro y las restantes son de color blanco. Los jugadores avanzan en los giros alternos. Cada jugador, cuando llegue su turno, corta solo un triángulo a partir del polígono. Los jugadores están autorizados a cortar triángulos a lo largo de las diagonales dadas. El ganador es el jugador que corta el triángulo negro. Su tarea es verificar si el jugador q tiene el primer turno es el ganador. NOTA: Nosotros llamamos un polígono convexo si un segmento que une dos puntos cualesquiera del polígono se encuentra en el polígono.

Input

La primera línea de la entrada estándar contiene un entero n, $4 \leq n \leq 50000$. Este es el número de vértices en el polígono. Los vértices del polígono se numeran, las agujas del reloj, a partir 0 de n-1. Las siguientes n-2 líneas contienen descripciones de los triángulos en el polígono. En cada línea hay tres números enteros no negativos a, b, c separados por espacios que son los números de los vértices del i-esimo triángulo. El primer triángulo en una secuencia es negro.

Output

La salida estándar debe tener una línea con la palabra:

- SI, si el jugador que comienza el juego es el que gana,
- N0, si no gana.

Ejemplos de entrada	**Ejemplos de salida**
6 0 1 2 2 3 4 4 2 0 0 5 4	SI

Problema E
Habilidad Mental

Existen muchos tests de habilidad mental, uno de ellos fue ideado por un estudiante de la UMSA, consiste en una pantalla llena de puntos de varios colores, debes encontrar mentalmente los dos puntos del mismo color que esten mas cerca el uno del otro. Obviamente a mas colores, mas se tarda en encontrar todos estos puntos, ademas es un buen ejercicio para poder separar lo que se ve y ejercitar el cerebro. Alex es un estudiante muy presumido, el sabe que no le va muy bien con los tests de habilidad mental, pero realmente quiere tener un buen resultado y no le importa como. Encontro una forma de dado la pantalla con todos los puntos tener todas las coordenadas de cada punto con su respectivo color. Lamentablemente Alex no es tan bueno programando como lo es presumiendo, por tanto solicita tu ayuda. A cambio promete ser un poco menos presumido contigo.

Input

La entrada comienza con un numero N tal que $1 < N \leq 10000$ que representa cuantos puntos hay en la pantalla, luego siguien N lineas cada una con 2 numeros enteros y un String representando las coordenadas x, y del punto y el color respectivamente tal que $0 < x, y \leq 1000$. Se garantiza que no siempre habran mas de 2 puntos por color y que todos los puntos son unicos.

Output

Debes imprimir cada color existente con la minima distancia entre los puntos de ese color, el numero debe estar aproximado con 2 digitos y el orden de salida debe estar ordenado alfabeticamente en funcion al color.

Ejemplos de entrada	**Ejemplos de salida**
12 2 2 Verde 2 6 Rojo 4 1 Azul 4 4 Verde 6 2 Verde 6 6 Rojo 6 9 Rojo 8 4 Azul 8 6 Azul	Azul 2.00 Rojo 2.83 Verde 3.00

Problema F
Tunquña

Bob ve a su hermano menor Jack jungando Tunquña. El esta facinando con lo interesante del juego, y decidio jugarlo.

Se pintan una secuencia de cuadrados en el suelo con la ayuda de una tiza, y se le asigna un numero a cada cuadrado(1, 2, 3, 4, ...). Bob esta frente a esos cuadrados. A partir de aqui, el lanza una piedra al primer cuadrado, entonces Bob se mueve a ese cuadrado y recoge la piedra, entonces el vuelve a lanzar la piedra esta vez 2 cuadrados adelante, se mueve a ese cuadrado y recoge la piedra, vuelve a lanzar la piedra pero esta vez 3 cuadrados adelante y asi sucesivamente. ¿Cual es el objetivo del juego?. El objetivo es comprobar si es posible llegar al N-esimo cuadrado con el procedimiento descrito. Bob es un poco perezoso. El jugara solo si esta seguro que puede llegar al N-esimo cuadrado. Ayuda a Bob a decidir si jugara o no.

Input

La primera linea contiene un numero entero T ($1 \leq T \leq 10^5$) que indica el numero de veces que Bob jugara el juego. Cada una de las T siguientes lineas contiene un unico entero N ($1 \leq N \leq 10^{18}$)que denota el N-esimo cuadrado.

Output

La salida se compone de varias lineas(una linea por cada juego), siguiendo los siguientes criterios: Si Bob es capaz de llegar al N-esimo cuadrado, entonces imprima "Go On Bob"(sin comillas) seguido del numero de movimientos necesarios para llegar al N-esimo cuadrado, ambos separados por un espacio. Si Bob no es capaz de llegar al N-esimo cuadrado, imprima "Better Luck Next Time"(sin comillas).

Explicacion de los casos

En el primer juego Bob puede saltar al cuadrado 1 luego el solo puede saltar al cuadrado 3, entonces no hay forma de llegar al cuadrado 2. En el segundo juego como se explico en el primer juego, el primero pude saltar al primer cuadrado, lanzar nuevamente la piedra entonces llegara al 3er cuadrado.

Ejemplos de entrada	**Ejemplos de salida**
2 2 3	Better Luck Next Time Go On Bob 2

Problema G
Fiboprimos

Juan es una persona muy curiosa, y desde que conoce los fiboncacci, ha quedado maravillado por las muchas propiedades que tiene. Cada fibonacci puede ser descompuesto en sus factores primos, si solo tomamos en cuenta los numeros de fibonacci, ciertos de ellos aparecen con un factor primo que ningun otro anterior fibonacci tenia, a estos Juan les llamo FiboPrimos, por ejemplo la siguiente tabla muestra los factores que no aparecen anteriormente de los primeros 9 fibonacci. En la tabla por ejemplo los factores primos de 34 son 2 y 17, el 17 no aparece como factor primo de ningun numero fibonacci anterior a 34 por tanto se trata de un FiboPrimo.

	Fib(1)	Fib(2)	Fib(3)	Fib(4)	Fib(5)	Fib(6)	Fib(7)	Fib(8)	Fib(9)
Valor Fibonacci	1	2	3	5	8	13	21	34	55
Fiboprimo	-	2	3	5	-	13	7	17	11

Juan es tan habilidoso que se dio cuenta de que no importa que tan grande el fibonacci, de tener un FiboPrimo, este siempre sera unico. A pesar de ello a el le cuesta mucho calcular el i-esimo FiboPrimo a mano, asi que pide tu ayuda.

Input

La entrada comienza con un numero natural $N \leq 100000$, que es el numero de consultas. Luego vendran N lineas cada una con un entero T tal que $0 \leq T \leq 100000$.

Output

Por cada entero T se pide imprimir una salida con el FiboPrimo de Fibo(T), de no tener Fibo(T) un FiboPrimo imprime, NO FIBOPRIMO

Ejemplos de entrada	**Ejemplos de salida**
5 2 7 8 9 5 11	2 7 17 11 NO FIBOPRIMO NO FIBOPRIMO