PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACION

Diseño y Análisis de Algoritmos - IIC2283
Tarea 4
Entrega: 23 de diciembre (hasta las 11:59pm)

La solución de cada problema debe ser un programa en Python 3. Estos programas deben ser entregados a través del Siding. El formato de entrega será informado en el foro del curso en el Siding. En este foro además puede hacer preguntas sobre los problemas de la tarea. Si quiere usar alguna librería en sus soluciones debe preguntar en el foro del curso si esta librería está permitida. El foro es el canal de comunicación oficial para todas las tareas.

Problema 1

En esta pregunta usted debe implementar el test de primalidad incluido en el capítulo "Algoritmos en teoría de números". Es decir, su programa debe recibir un número n y devolver `YES` si es primo y `NO` si es compuesto.

Importante: En esta pregunta se puede usar las librerías `random` y `math`. Excepcionalmente, el código en esta pregunta será revisado a mano.

Formato

El input consistirá de una línea con el número n.

Su output debe ser una sola línea con el string `YES` si n es primo y `NO` si n es compuesto.

Límites

$2 \leq n \leq 10^{50}$

Tiempo de ejecución

5 segundos

Complejidad esperada

$O(\log n^3)$. Para este problema se espera un algoritmo **aleatorizado** con una probabilidad de error menor o igual a $\frac{1}{2^{20}}$.

Ejemplos

Input 1

```
641
```

Output 1

```
YES
```

Input 2

```
3200564
```

Output 2

```
NO
```

Problema 2

Como se menciona en el capítulo "Algoritmos en teoría de números", la identidad de Bézout dice que para cada $a, b \in \mathbb{N}$ tales que $a \neq 0$ o $b \neq 0$, existen $s, t \in \mathbb{Z}$ tales que:

$$s \cdot a + t \cdot b = m$$

donde m es el máximo común divisor (MCD) de a y b. Naturalmente, este teorema se puede extender de la siguiente forma: Para cada $a, b \in \mathbb{N}$ tales que $a \neq 0$ o $b \neq 0$, existen $s, t \in \mathbb{Z}$ tales que $s \cdot a + t \cdot b = m$, donde $\text{MCD}(a, b)$ **divide a** m.

En esta pregunta usted debe hacer un programa que reciba s, t y m, y entregue un par de números naturales a y b tal que $a \geq 1$, $b \geq 1$ y $s \cdot a + t \cdot b = m$. No es necesario que $\text{MCD}(a, b) = m$.

Como condición adicional, este par debe ser lexicográficamente mínimo. Es decir, si existe un par (a', b') distinto que cumple con lo anterior, tiene que cumplirse que $a' > a$, o bien $a' = a$ y $b' > b$. En otras palabras, el número a pedido es el mínimo posible, y si hay más de un número b válido para este valor a, se pide el mínimo de ellos.

El input va a cumplir con que $s \geq 2$, $t \leq -2$ y m es múltiplo de $\text{MCD}(s, t)$.

Formato

El input consiste de una sola línea con los dígitos s, t y m separados por espacios.

Su output debe ser una sola línea con los números a y b separados por un espacio.

Límites

$2 \leq s \leq 10^{12}$
$-10^{12} \leq t \leq -2$
$1 \leq m \leq 10^{12}$
m es múltiplo de $\text{MCD}(s, t)$.

Tiempo de ejecución

2 segundos

Complejidad esperada

$O(\log(|s| + |t|))$

Ejemplos

Input 1

```
2 -2 2
```

Output 1

```
2 1
```

Input 2

```
3 -10 3
```

Output 2

```
11 3
```

Nota: `1 0` no es una solución válida.

Input 3

```
37519548 -81935448 71983128
```

Output 3

```
4207392 1926631
```

Problema 3

Hay N ciudades y M agencias de vuelo. Las M agencias están presentes en todas las ciudades. Cada agencia ofrece un único vuelo en cada ciudad. Formalmente, la agencia i en la ciudad j ofrece el vuelo $j \to D_{i,j}$ ($D_{i,j} \in [1, N]$). Notar que un vuelo podría partir y terminar en la misma ciudad (imagine una ciudad grande con muchos aeropuertos). Ahora suponga que hay K personas, tales que la persona k-ésima está parada actualmente en la ciudad c_k y planea volar exclusivamente con la agencia a_k (no le gustan las otras). Además, estas personas siguen estrictamente el siguiente comportamiento: apenas se bajan de un avión, inmediatamente toman el único vuelo disponible de la agencia que les gusta en esa ciudad.

Dadas estas condiciones, ¿existe alguna ciudad tal que las K personas pasen todas por ella **simultáneamente**? Si existe, ¿cuál es la primera ciudad en la que eso ocurre y en cuánto tiempo? Puede asumir que un vuelo siempre toma una unidad de tiempo.

Formato

El input consta de varias líneas. La primera línea consiste en 3 enteros N, M y K (la cantidad de ciudades, agencias de vuelo y personas respectivamente). Luego vienen M líneas, cada una con N enteros. La i-ésima línea describe los N vuelos ofrecidos por la i-ésima agencia, donde el j-ésimo número es un entero $D_{i,j}$ indicando un vuelo de la ciudad j a la ciudad $D_{i,j}$. Finalmente vienen K líneas. La k-ésima línea consiste en un par de enteros c_k y a_k, indicando que la k-ésima persona está inicialmente parada en la ciudad c_k y sólo hará vuelos con la agencia a_k.

Su output debe ser una única línea con 2 enteros, X y T, indicando que todas las personas se juntan por primera vez en la ciudad X luego de transcurridas T unidades de tiempo, o bien retornar el caracter "*" si es que no existe una ciudad en la que todos pasen por ella simultáneamente.

Límites

$1 \leq N, M \leq 100$
$1 \leq K \leq 10$
$1 \leq D_{i,j} \leq N$ para cada $i \in [1, M]$, $j \in [1, N]$
$1 \leq c_k \leq N$ para cada $k \in [1, K]$
$1 \leq a_k \leq M$ para cada $k \in [1, K]$

Tiempo de ejecución

2 segundos

Complejidad esperada

$O(M + NK^2 log(N))$

Ejemplos

Input 1

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

Output 1

```
1 2
```

Input 2

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

Output 2

```
*
```

Problema 4

María descubrió el mundo de las aplicaciones para celular y ahora que está en búsqueda de pareja su interés está especialmente centrado en *Binder, la app para encontrar tu pareja ideal*. La app es bien sencilla, cada persona al registrarse debe ingresar 2 enteros no-negativos: su nivel de belleza y su coeficiente intelectual. Así, la i-ésima persona en el sistema tiene una belleza de B_i y un coeficiente intelectual de CI_i. Luego, al momento de buscar parejas *Binder* le solicita al usuario que defina una función de utilidad $F(i) = W_B \cdot B_i + W_{CI} \cdot CI_i$, donde W_B y W_{CI} deben ser **enteros positivos** ingresados por el usuario que reflejen la importancia que

le asigna a cada dimensión respectivamente. De esta manera, *Binder* procede a rankear a las personas registradas en el sistema en base a la función de utilidad definida por el usuario. En caso de haber empates, *Binder* desempata aleatoriamente, de modo que entre personas empatadas cualquier orden es posible.

María se pone a jugar con los valores W_B y W_{CI} y se da cuenta que el ranking retornado por *Binder* a veces cambia mucho, poco o se mantiene igual. Ante esto María se pregunta cuántos posibles rankings diferentes pueden haber para un conjunto fijo de N personas, pero sus conocimientos matemáticos y de teoría de números son muy limitados. ¿Podrías ayudar a María a resolver este complicado enigma?

Formato

El input consta de varias líneas. La primera línea consiste en un entero N indicando la cantidad de personas. Le siguen N líneas, la i-ésima línea con 2 enteros B_i y CI_i indicando la belleza y el coeficiente intelectual de la i-ésima persona respectivamente.

Su output debe consistir en un único entero correspondiente a la cantidad de maneras distintas posibles de rankear las N personas usando la app de *Binder*. Dado que este número puede ser muy grande, se pide el **resto que queda al dividirlo por** $10^9 + 7$.

Límites

$1 \leq N \leq 1000$
$0 \leq B_i, CI_i \leq 10^4$ para $i \in [1, N]$

Tiempo de ejecución

3 segundos

Complejidad esperada

$O(N^2 \times \log(10^9 + 7))$

Ejemplos

Input 1

```
3
0 2
1 2
2 1
```

Output 1

```
3
```

Input 2

```
4
1 0
1 3
```

```
2 2
1 3
```

Output 2

```
6
```

Input 3

```
4
0 0
3 1
0 0
0 0
```

Output 3

```
6
```